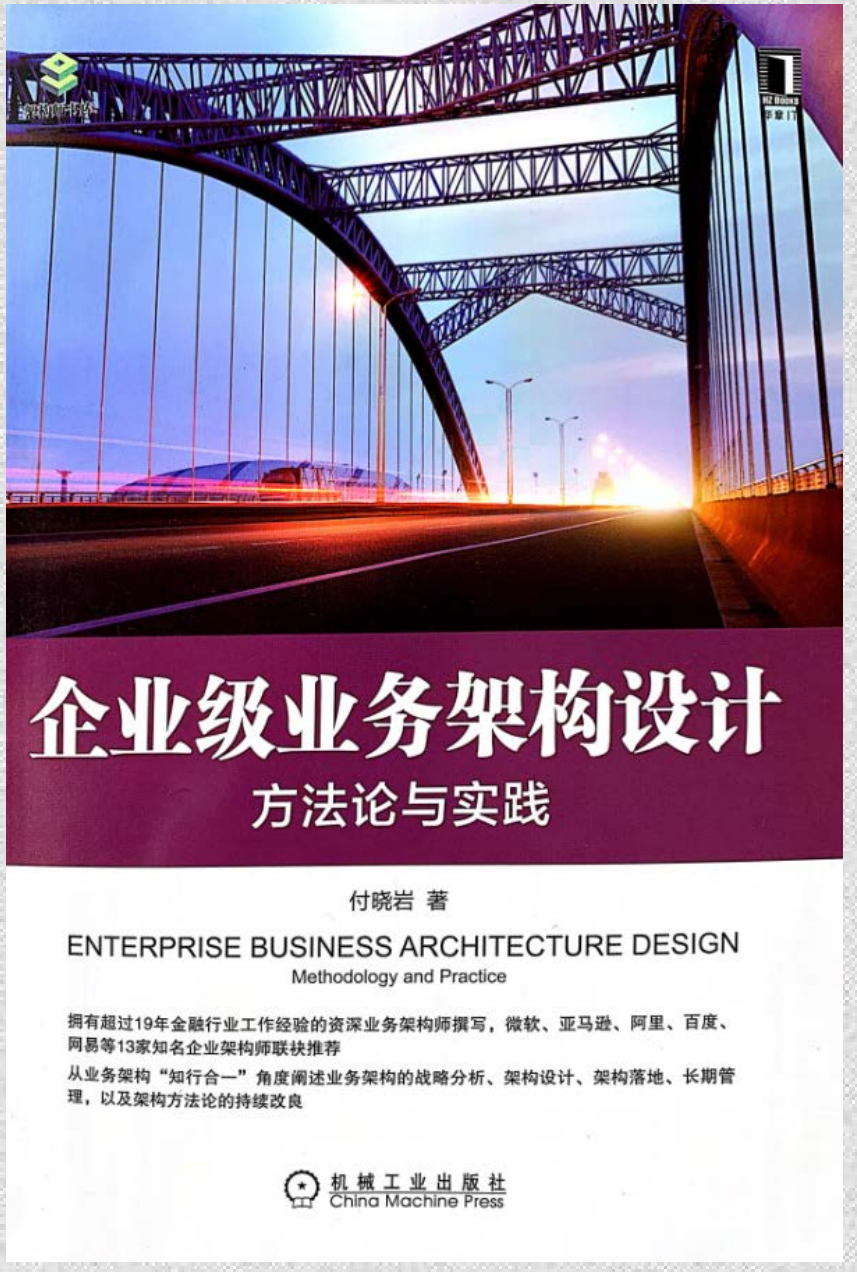


【读书笔记】

企业级业务架构设计

方法论与实践

一部跨越百年的经营战略理念集结之作

作者简介

付晓岩

资深的企业级业务架构师，有20年的金融行业工作经验；在软件过程、系统设计与分析、架构设计方面有深厚的理论知识和丰富的实践经验；

主要著作：

《企业级业务架构设计：方法论与实践》、《银行数字化转型》、《聚合架构：面向数字生态的构件化企业架构》

职业经历：

曾就职于IBM副合伙人，全球企业咨询服务部大中华区金融核心锐变团队业务发展和交付总监。后就职建信金融科技有限责任公司，参与了建行多个系统的业务架构设计，形成了一套方法论；InfoQ中文站专栏作家，发表《中台之上》系列文章，累计阅读量超过10万。维护着个人微信公众号：晓谈岩说，与各行业读者广泛交流，持续提升方法的普适性。现任阿里云新金融事业部资深行业解决方案总监、资深企业级业务架构师。

推荐理由

作者权威

作者在金融行业有超过20年的工作经验

知行合一

从“知线”和“行线”两条主线阐述完整的企业级业务架构

体系完善

业务架构基础、业务架构设计、业务架构落地、架构方法改良、架构与中台5个维度

授人以渔

相比具体的操作，更多的是业务架构的经验和感悟，引发共鸣

业界好评

微软、亚马逊、阿里、百度、网易等13家知名企业架构师联袂推荐

- 书中既有对业务架构设计理论的清晰介绍，又有大量“接地气”的案例分析。

——陈耿微软全球黑带技术专家

- 除了系统的业务架构设计理论外，还有基于大规模企业级项目的业务架构设计方法论。

——孙玄转转公司首席架构师/技术委员会主席

- 企业级应用开发中业务架构不可或缺，业内对业务架构的论述很少，本书填补了这一空白。

——曹洪伟百度DuerOS首席布道师

- 本书是作者多年业务架构的经验，既有由浅入深的理论讲解，又有贴合实际的案例实践。

——李鑫天弘基金（余额宝）移动平台技术总监兼首席架构师

- 从企业战略分析到与IT架构的衔接，从业务模型的打造到架构的管理方法，本书让我茅塞顿开。

——王天庆马蜂窝基础平台架构师

- 作者以金融行业的业务架构作为实例，从“行线”和“知线”两个维度系统讲解了业务架构的方方面面。

——茹炳晟DellEMC中国研发集团资深架构师

- 本书提出了业务架构“知行合一”的观点，让读者充分认识到业务架构的重要性，以及应该如何正确地进行业务架构设计。

——张逸领域驱动设计布道师

- 作者以“行线”和“知线”为指引，全面介绍了业务架构的设计、落地以及架构的持续改良。

——郑然百度主任研发架构师

- 作者分享业务架构设计的原理和落地方法，并阐述了如何通过业务架构推动技术架构和中台体系的建设。

——刘超网易杭州研究院云计算技术部首席架构师

- 作者把他在金融领域超过19年的经验和思考娓娓道来，逻辑清晰完整，案例引人思考。

——黄帅AWS 认证云架构师

本书核心内容

No.1 业务架构基础

- 经典架构设计理论
- 业务架构发展理念
- 业务架构与技术架构
- 常见业务模型及其建模原则

No.2 业务架构设计

- 设计的起点：战略分析、对标分析、组织结构调整
- 设计的过程：价值链分析、行为分析、数据分析、组件分析
- 设计的难点：标准化方法、避免过度整合

No.3 业务架构落地

- 从业务架构模型到方案
- 业务架构方案的实施
- 转型后的长期应用机制
- 业务架构的“笨重”与敏捷
- 企业级业务架构的5个难点

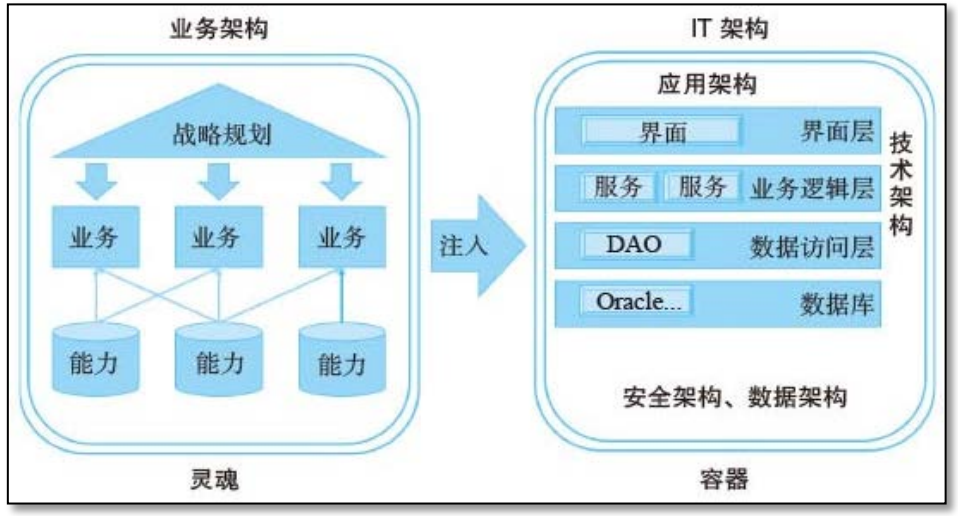
No.4 架构方法改良

- 面向架构的设计
- 轻量级架构管理工具
- 构件模型与产品创新

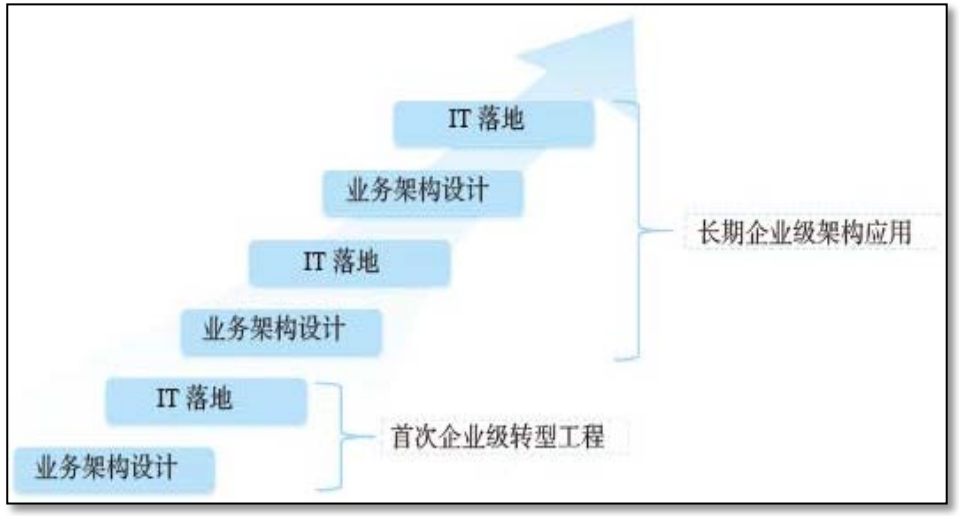
No.5 业务架构与中台

- 阿里中台
- 企业文化与中台
- 业务架构与中台

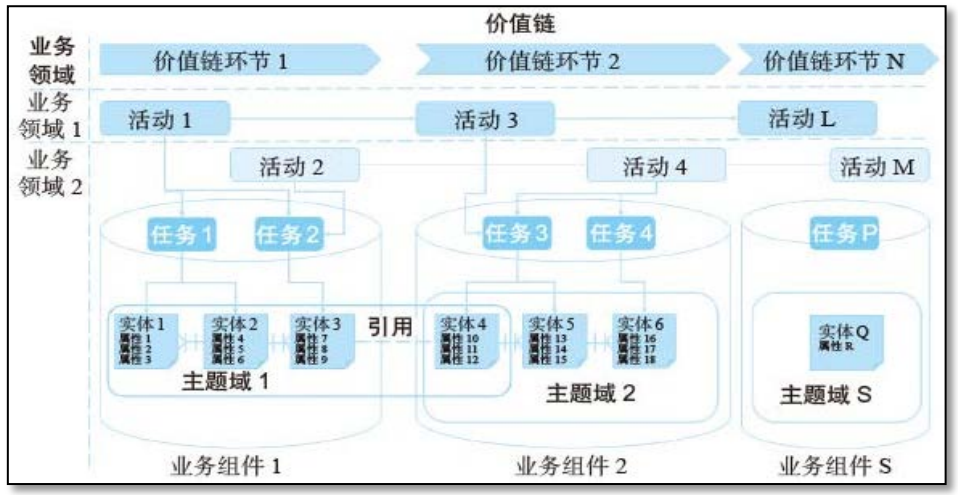
精彩摘要



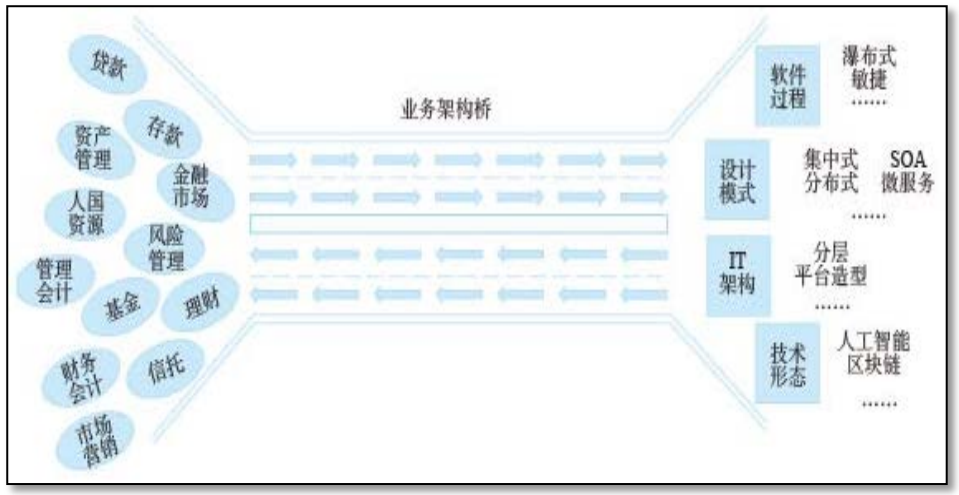
No.1 业务架构与IT架构的关系



No.2 交替上升的业务架构设计与IT实现

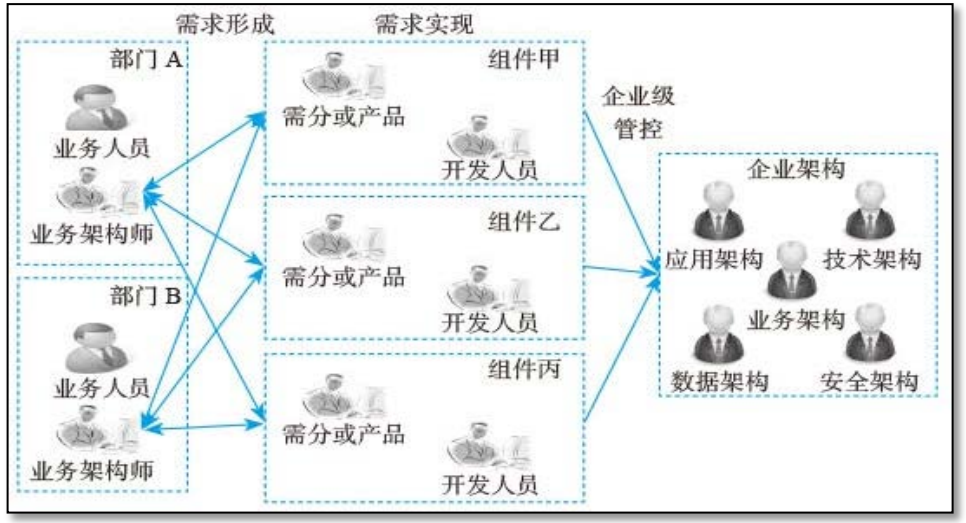


No.3 业务架构整体逻辑关系图

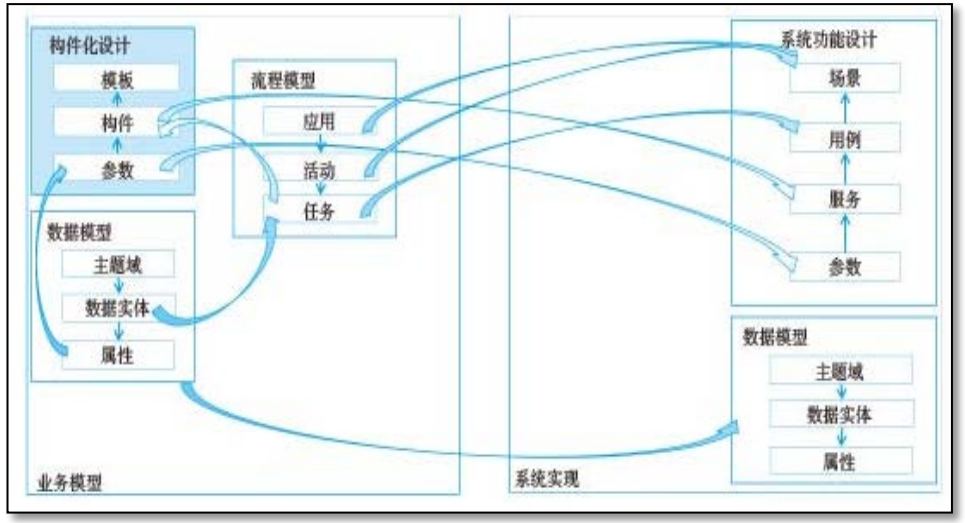


No.4 业务架构桥

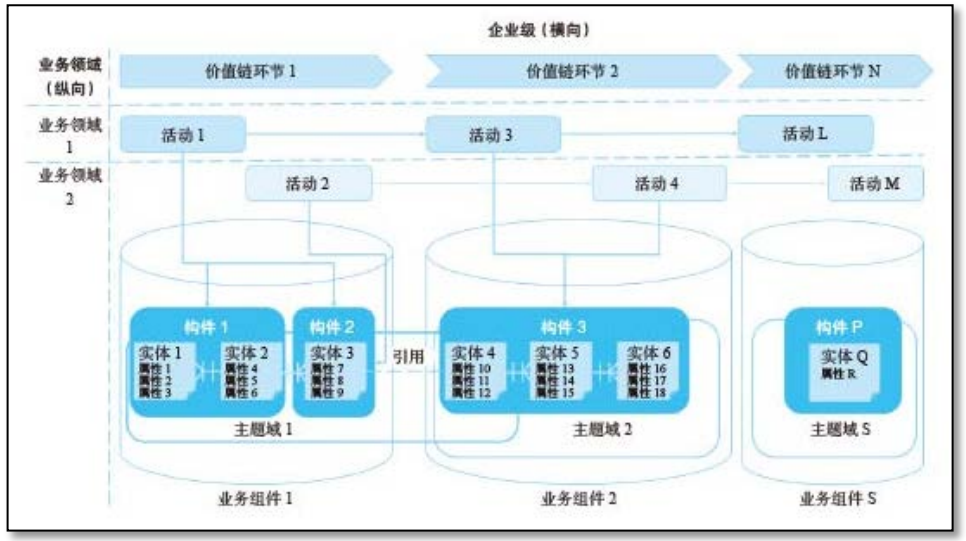
精彩摘要



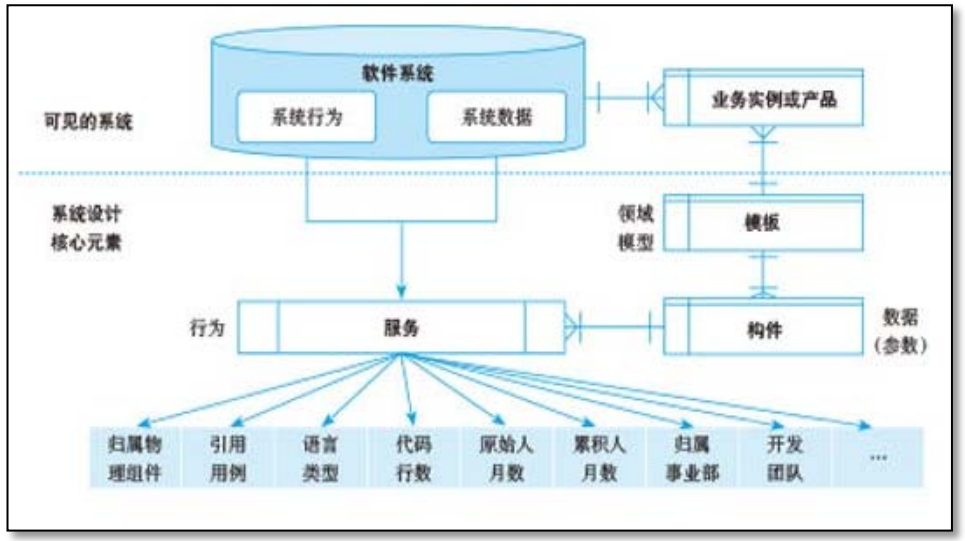
No.5 业务架构师工作组织形式示意图



No.6 构件模型与业务模型、系统实现的关系示意图



No.7 基于构件化设计的业务架构整体逻辑关系示意图



No.8 轻量级架构管理工具逻辑图

目录

目录

企业级业务架构的主线

业务架构的“知行合一”

行线：覆盖了企业级业务架构设计、实现及后期管理的完整过程。

知线：架构方法论的持续改良。

架构师有责任和义务持续改进、宣传架构的设计方法，推动架构理念在企业以及社会范围内的磨砺、传播，实现架构的“知行合一”。

第一部分：业务架构基础篇

第1章　业务架构的发展历程

- 1.1　Zachman模型
- 1.2　TOGAF
- 1.3　FEA和DODAF
- 1.4　沉吟至今
- 1.5　业务架构的定义

第2章　业务架构的作用及与IT架构的关系

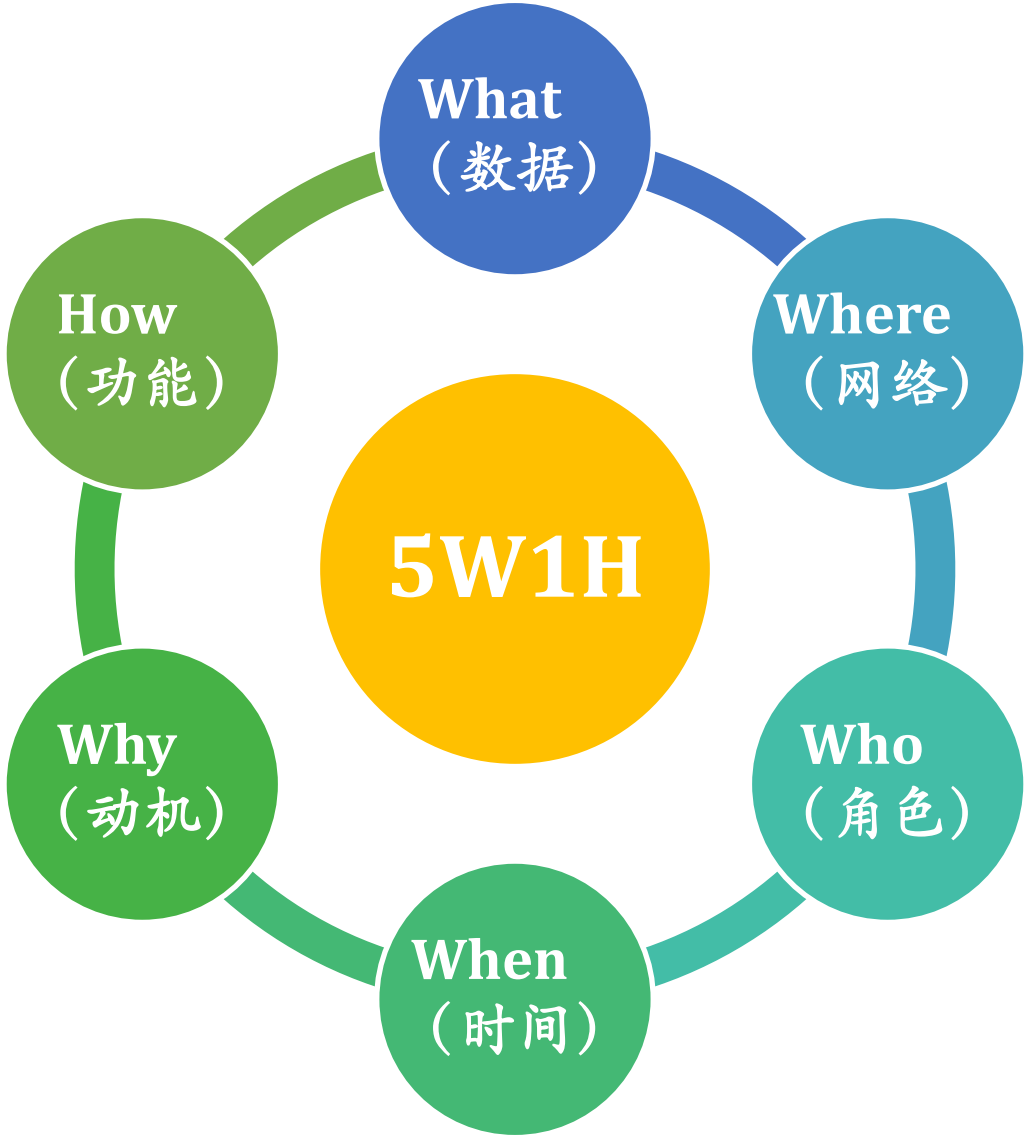
- 2.1　业务架构的作用
- 2.2　业务架构与IT架构的关系

第3章　架构伴侣：业务模型

- 3.1　模型与业务模型
- 3.2　常见的建模方法
- 3.3　建模原则与模型思维的应用

第1章 业务架构的发展历程

1.1 Zachman模型

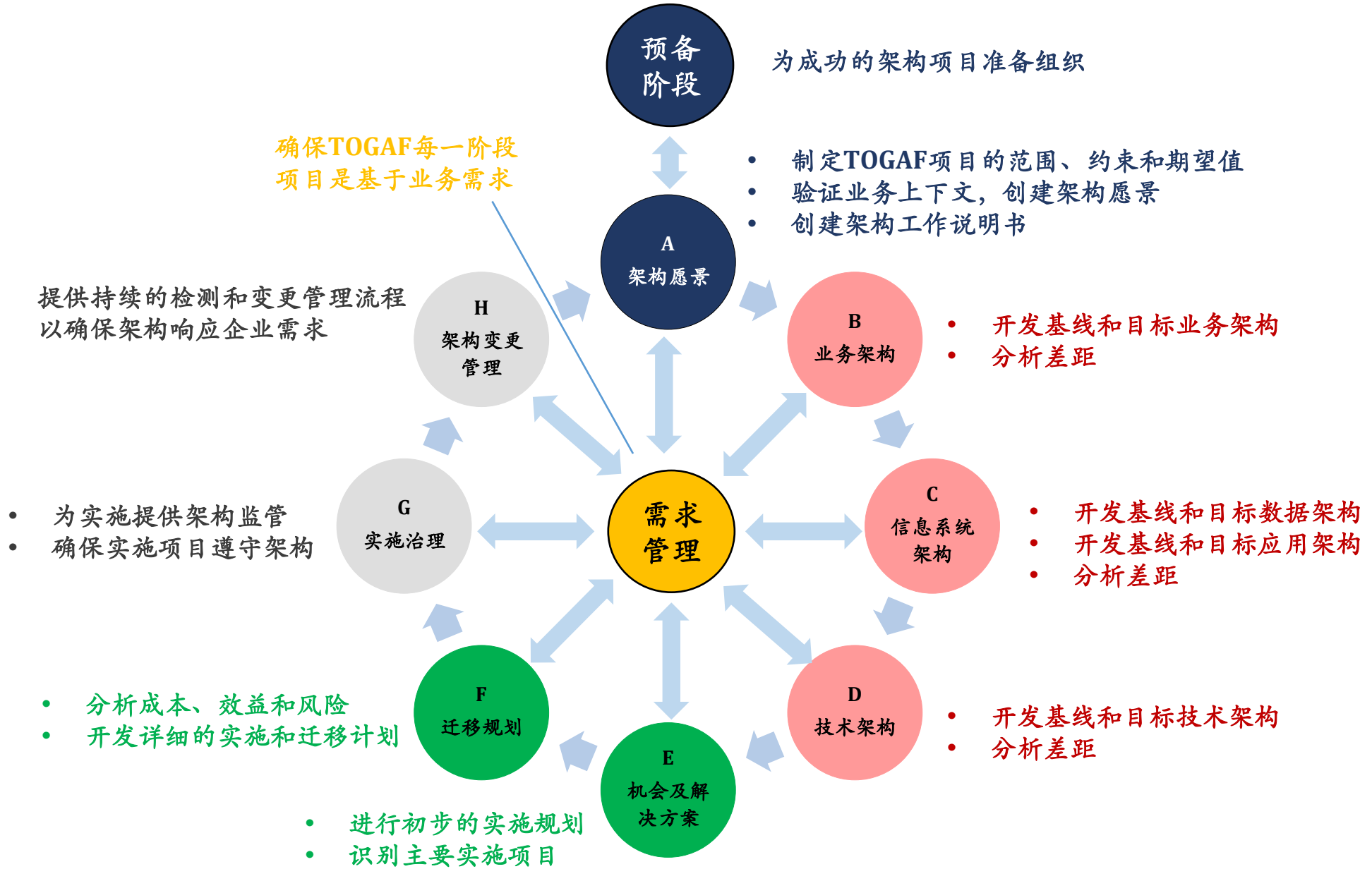


1.1 Zachman模型

Zachman模型

	数据（什么？）	功能（怎样？）	网络（哪里？）	角色（谁？）	时间（何时？）	动机（为何？）
目标范围	列出对业务至关重要的元素	列出业务执行的流程	列出与业务运营有关的地域分布要求	列出对业务重要的组织部门	列出对业务重要的事件及时间周期	列出企业目标、战略
业务模型	实体关系图（包括M:M关系、N-ary关系，归因关系）	业务流程模型（物理数据流程图）	物流网络（节点和链接）	基于角色的组织层次图，包括相关技能规定、安全保障问题	业务主进度表	业务计划
信息系统模型	数据模型（聚合体、完全规格化）	关键数据流程图、应用架构	分布系统架构	人机界面架构（角色、数据、入口）	相依关系图、数据实体生命历程（流程结构）	业务标准模型
技术模型	数据架构（数据库中的表格列表及属性）遗产数据图	系统设计：结构图、伪代码	系统架构（硬件、软件类型）	用户界面（系统如何工作）、安全设计	“控制流”图（控制结构）	业务标准设计
详细展现	数据设计（反向规格化）（物理存储器设计）	详细程序设计	网络架构	屏显、安全机构（不同种类数据源的开放设定）	时间、周期定义	程序逻辑的角色说明
功能系统	转化后的数据	可执行程序	通信设备	受训的人员	企业业务	强制标准

1.2 TOGAF模型


预备阶段
为成功的架构项目准备组织
确保TOGAF每一阶段项目是基于业务需求
A 架构愿景
制定TOGAF项目的范围、约束和期望值
验证业务上下文，创建架构愿景
创建架构工作说明书
B 业务架构
开发基线和目标业务架构
分析差距
C 信息系统架构
开发基线和目标数据架构
开发基线和目标应用架构
分析差距
D 技术架构
开发基线和目标技术架构
分析差距
E 机会及解决方案
进行初步的实施规划
识别主要实施项目
F 迁移规划
分析成本、效益和风险
开发详细的实施和迁移计划
G 实施治理
为实施提供架构监管
确保实施项目遵守架构
H 架构变更管理
提供持续的检测和变更管理流程以确保架构响应企业需求
需求管理

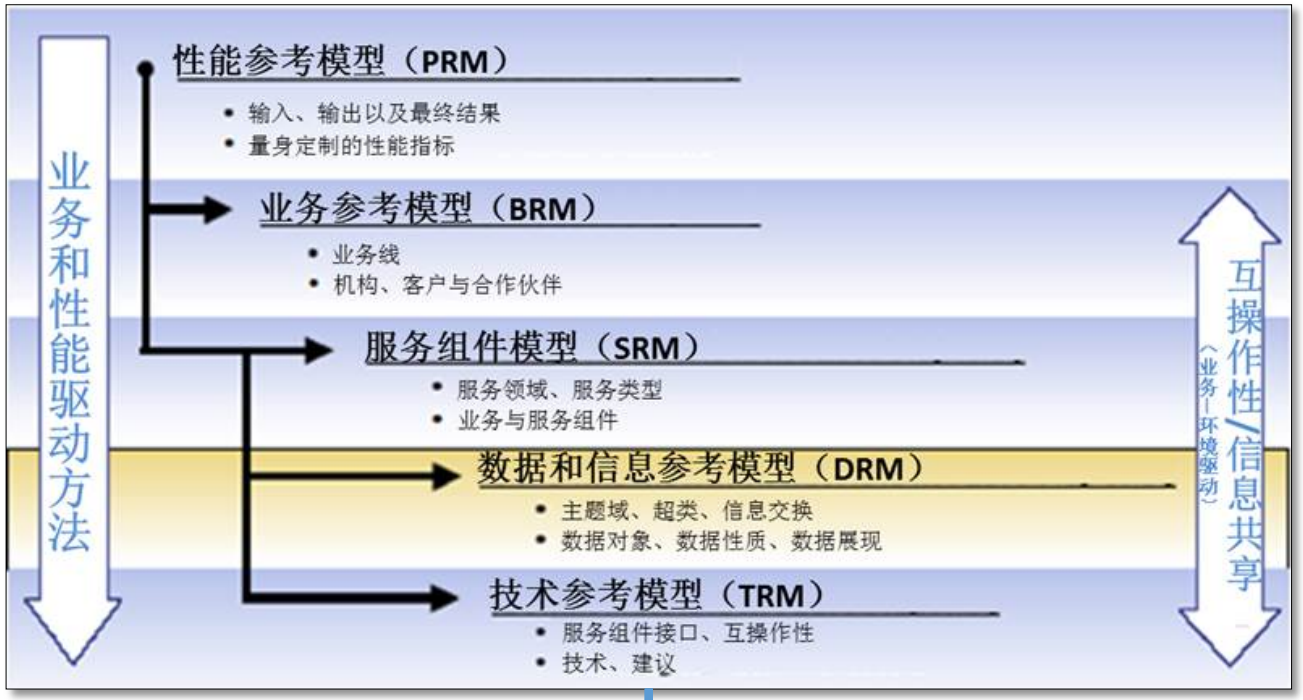
1.2 TOGAF模型

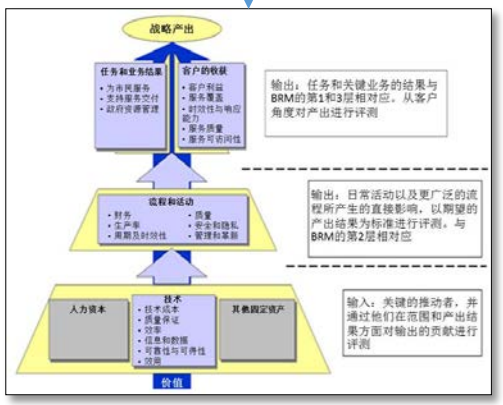
TOGAF 9 交付物：目录、矩阵、图

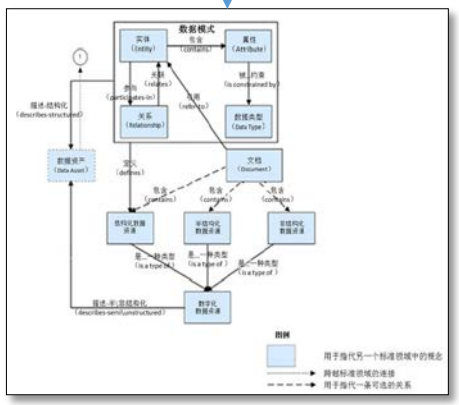
<table>
<tr>
<td>预备阶段
原则目录

阶段A，架构愿景
利益关系者映射矩阵
价值链图
解决方案概念图</td>
<td>阶段B，业务架构
组织/实施者目录
驱动力/目标/目的目录
角色目录
业务服务/功能目录
位置目录
流程/厚件控制/产品目录
契约测度目录
业务互动矩阵
施动者/角色矩阵
业务轨迹图
业务服务/信息图
功能的分解图产品
生命期图
目标/目的/服务图
用例图
组织分解图
流程图
事件图</td>
<td>阶段C，数据架构
数据实体/数据构件目录
数据实体/业务功能矩阵
系统数据矩阵
类图
数据发布图
数据安全图
类阶层图
数据迁移图
数据生命周期图</td>
<td>阶段C，应用架构
应用组合目录
接口目录
系统/组织矩阵
角色/系统矩阵
系统/功能矩阵
应用互动矩阵
应用通信图
应用和用户位置图
系统用例图
企业可管理性图
流程/系统实现图
软件工程图
应用迁移图
软件分布图</td>
</tr>
<tr>
<td colspan="2">阶段D，技术架构
技术标准目录
技术组合目录
系统/技术矩阵
环境和位置图
平台分解图
处理图
网络计算/硬件图
通信工程图</td>
<td>阶段E，机会及解决方案
项目背景图
效益图</td>
<td>需求管理
需求目录</td>
</tr>
</table>

1.3 FEA和DODAF模型（FEA模型）







1.3 FEA和DODAF模型（DODAF模型）

全景视图
（描述体系结果背景、目标、范围等）

数据与信息视图
（描述体系结构内容的数据相互关系与结构）

标准视图
（描述可应用到体系结构的标准指南和发展预测）

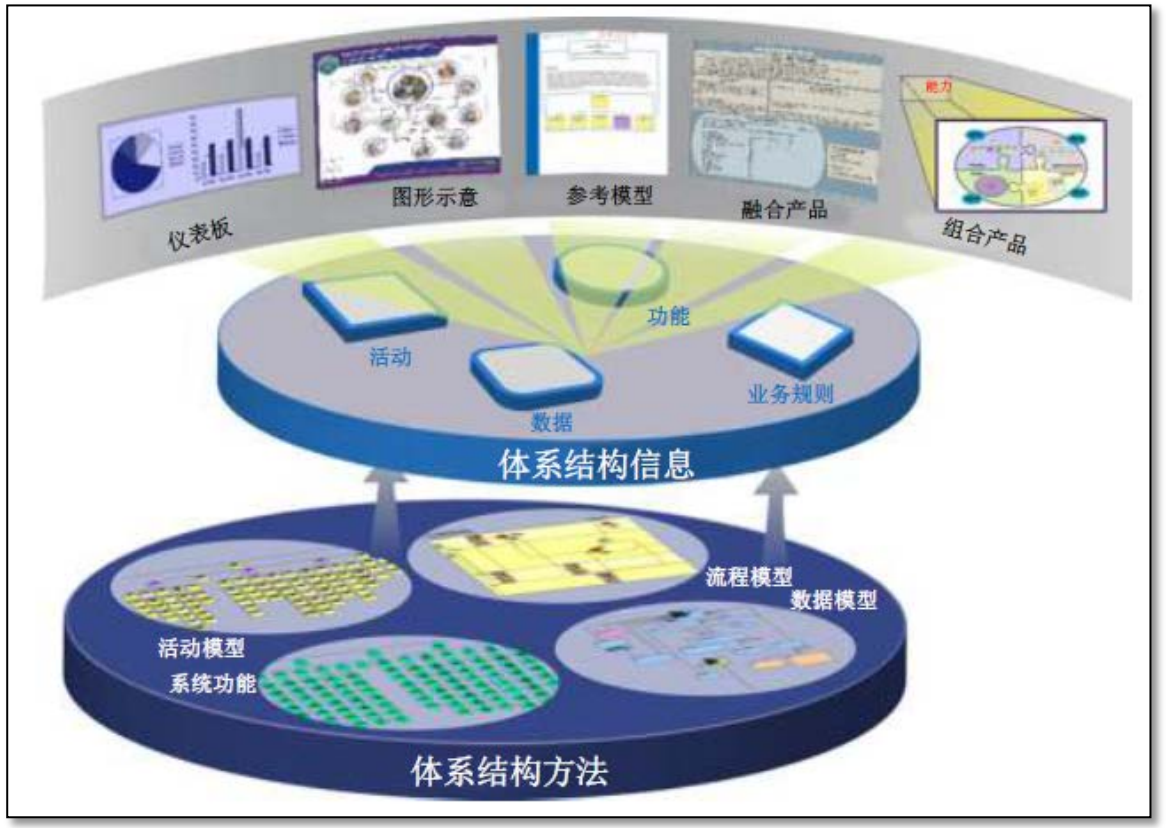
能力视图
（描述能力要素、交付时间及其部署后形成的能力）

业务视图
（描述业务愿景、程序、活动及其需要）

服务视图
（描述性能、活动、服务及其交换规则）

系统视图
（描述系统组成部分、系统间连接关系及其背景与规则）

项目视图
（描述业务与能力需求及正在实施中的不同项目间的关系，能力管理与采购系统程序之间的从属关系）



1.3 FEA和DODAF模型（DODAF模型）

DODAF 的核心——8个视图与52个模型

能力视图	作战视图	系统视图	服务视图
CV-1构想模型 CV-2能力分类模型 CV-3能力实现时段模型 CV-4能力依赖关系模型 CV-5能力与机构发展映射模型 CV-6能力与作战活动映射模型 CV-7能力与服务映射模型	OV-1顶层作战概念图	SV-1系统接口表述模型	SveV-1服务接口表述模型
	OV-2作战资源流表述模型	SV-2系统资源流表述模型	SveV-2服务资源流表述模型
	OV-3作战资源流矩阵	SV-3系统一系统矩阵	SveV-3a服务一系统矩阵 SveV-3b服务一服务矩阵
	OV-4组织关系图	SV-4系统功能模型	SveV-4服务功能模型
	OV-5a作战活动分解树 OV-5b作战活动模型	SV-5a 系统功能与作战活动跟踪矩阵 SV-5b系统与作战活动跟踪矩阵	SveV-5服务与作战活动跟踪矩阵
		SV-6系统资源流矩阵	SveV-6服务资源流矩阵
		SV-7系统度量矩阵	Svev-7服务度量矩阵
		SV-8系统演变表述模型	SveV-8服务演变表述模型
		SV-9系统技术和技能预测	SveV-9服务技术和技能预测
	OV-6a作战规则模型 OV-6b作战状态转换模型 OV-6c作战事件跟踪模型	SV-10a系统规则模型 SV-10b系统状态转换模型 SV-10c系统事件跟踪模型	SveV-10a 服务规则模型 SveV-10b服务状态转换模型 SveV-c 服务事件跟踪模型
全景视图	**标准视图**	**项目视图**	**数据视图**
AV-1综述和概要信息模型 AV-2综合词典	StdV-1标准概要模型 StdV-2标准预测模型	PV-1项目与机构关系模型 PV-2项目实现时段模型 PV-3项目与能力映射模型	DIV-1概念数据模型 DIV-2逻辑数据模型 DIV-3物理数据模型

1.4 沉吟至今

业务架构师不常见，业务架构有点虚的原因

- **用得少**：单体式或竖井式开发更为常用，这种开发没有横向视角，无需强调业务架构
- **难设计**：业务架构对战略的分解、业务架构自身的整合和标准化，到IT设计的过渡都存在不少陷阱，业务越复杂就难以驾驭。
- **易偏离**：施工期间由于客观因素可能导致实施对业务架构的偏离，这种偏离如果没有得到及时纠正或架构调整，积累久了就会造成业务架构的失真。
- **难维护**：度过了业务架构落地困难期，感受到维护架构的难度而心生放弃，从而降低了对业务架构的评价。

1.5 业务架构的定义

业务架构：
以实现企业战略为目标，构建企业整体业务能力规划并将其传导给技术实现端的结构化企业能力分析方法。

使命：
面向复杂系统构建，降低复杂度。

作用：
应当将业务架构从IT战略中独立出来，更多的面向业务人员，以充当业务与技术之间的桥梁。

使用场景：
可以用于单个产品线或业务种类的领域级分析，也可以用于跨越产品线、业务领域的企业级分析。

第2章 业务机构的作用与IT架构的关系

2.1 业务架构的作用

传统行业中的先行者已经实现了“数字化”

- 星巴克16%的收入来自手机客户端；
- 滴滴2016年日生产数据超过50TB，每天规划90亿次路径；2017年全年提供出行服务74.3亿次；
- 美团为骑手提供智能语音服务。

后觉者如何启动自身的“数字化”

- 给你一把狙击枪，不代表你已经成为一名合格的狙击手。
- 不是让所有人都去学习IT技术，而是转变思维方式，架起一道跨越“数字鸿沟”的桥梁。
- 业务架构可以帮助业务人员整体化、结构化思考问题，从业务和系统的视角，去理解业务和技术。
- 帮助技术人员理解业务人员，让业务和技术能够使用同一种“语言”工作。

业务架构带动深度融合

- 深度融合代表互相深入理解，从思维方式的转变开始，通过建立业务架构，业务和技术都能向对方的领域迈出一步。对业务人员挑战更大。
- 业务如果不能很好的结构化、模块化，那么技术人员也很难做出一个具有良好架构的系统。

来自银行的声音

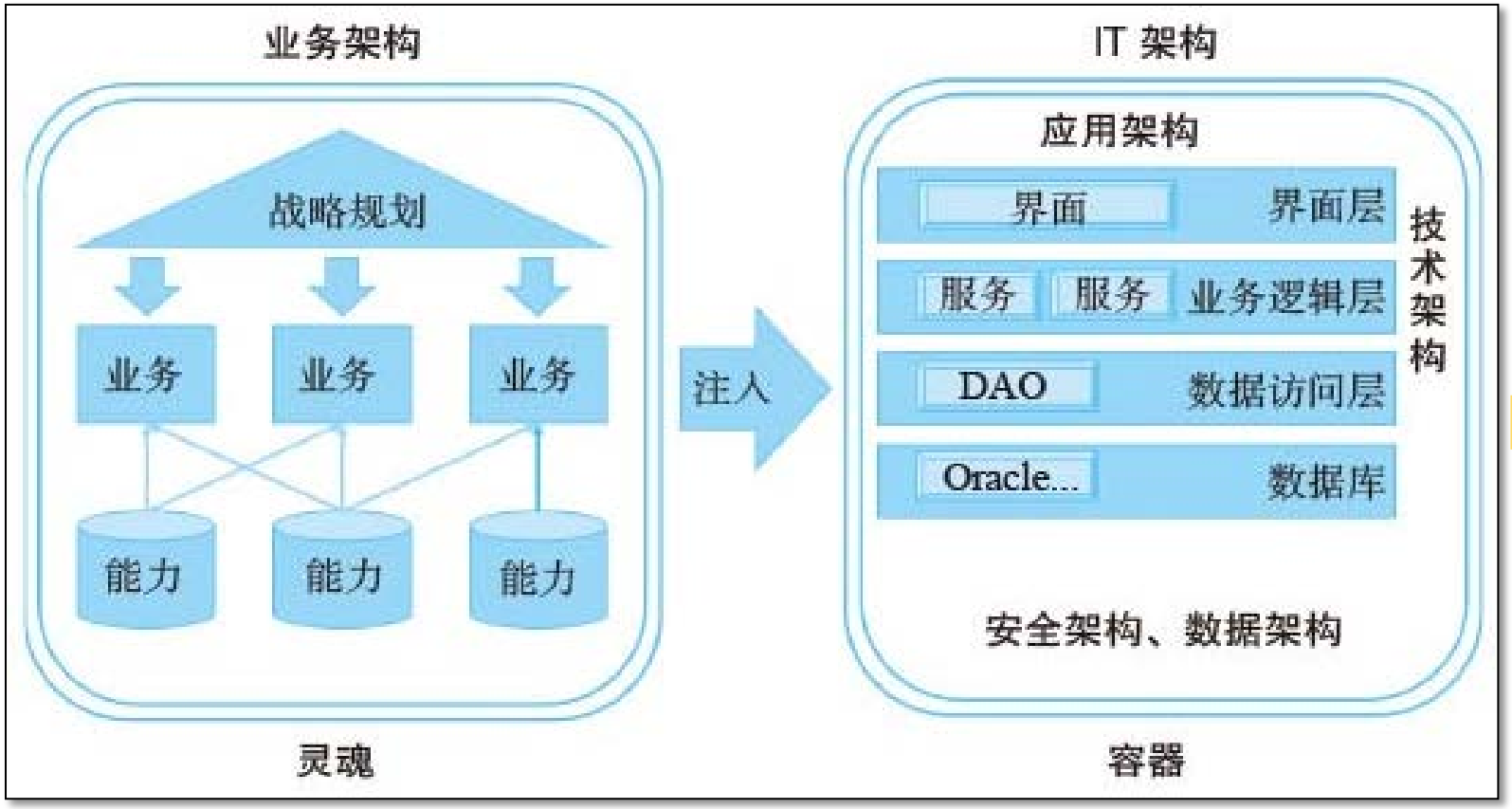
- 银行业务种类繁多，组织结构庞杂，更需要合理的“企业级”规划。
- 中国建设银行历时6年的新一代核心系统建设，通过采用企业级建模，形成了以“四个一（一套模型、一套IT架构、一套实施工艺和一套管理流程）”为特征的企业级工程实施方法，及集团层面的流程模型、数据模型、产品模型和用户体验模型。
- 工行重构了业务架构视图，重新规划了28个领域的业务架构，形成了2300余个任务组件。

2.2 业务架构与IT架构的关系

业务架构从企业战略出发，按照企业战略设计业务及业务过程，业务过程是需要业务能力支撑的，从战略到业务再到对业务能力的需要，就形成了支持企业战略实现的能力布局，这个布局理解为业务架构。

业务架构设计尽可能的追求以更集约的能力实现更为多变的业务或服务。这也是中台战略追求的目标。

业务架构设计完成后，“灵魂”就诞生了。



IT架构是根据“灵魂”的需要来设计“容器”。

IT架构通常分为：
应用架构、
技术架构、
数据架构、
安全架构。

将“灵魂”注入“容器”是技术人员的重要工作，能否顺利注入，让“灵魂”有个适宜的居所，则是有赖于对“灵魂”的充分认知；而引导这一认知过程，让原本朦胧神秘、纷繁复杂的“灵魂”清晰可见的，正是业务架构。

2.2 业务架构与IT架构的关系

应用架构：重点关注功能布局，与业务架构关系紧密，称为业务架构设计的“紧后工序”。

技术架构：主要关注分层结构，一个逻辑分层很可能要通过多种平台才能实现，在分层中要加入平台规划。

数据架构：与业务架构关系密切，甚至可以归类为业务机构的组成部分。

安全架构：与业务架构的关系不十分紧密，发展趋势向业务架构（或企业战略）靠拢，更贴近实际业务需要，符合业务发展方向。

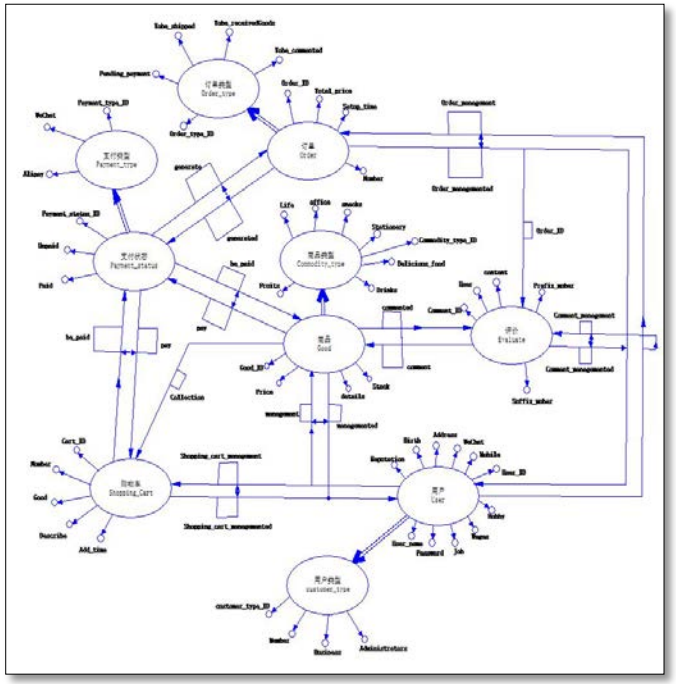
第3章 架构伴侣：业务模型

3.1 模型与业务模型

模型是所研究的系统、过程、事务或概念的一种表达形式，也可指根据实验、图样放大或缩小而制作的样品。

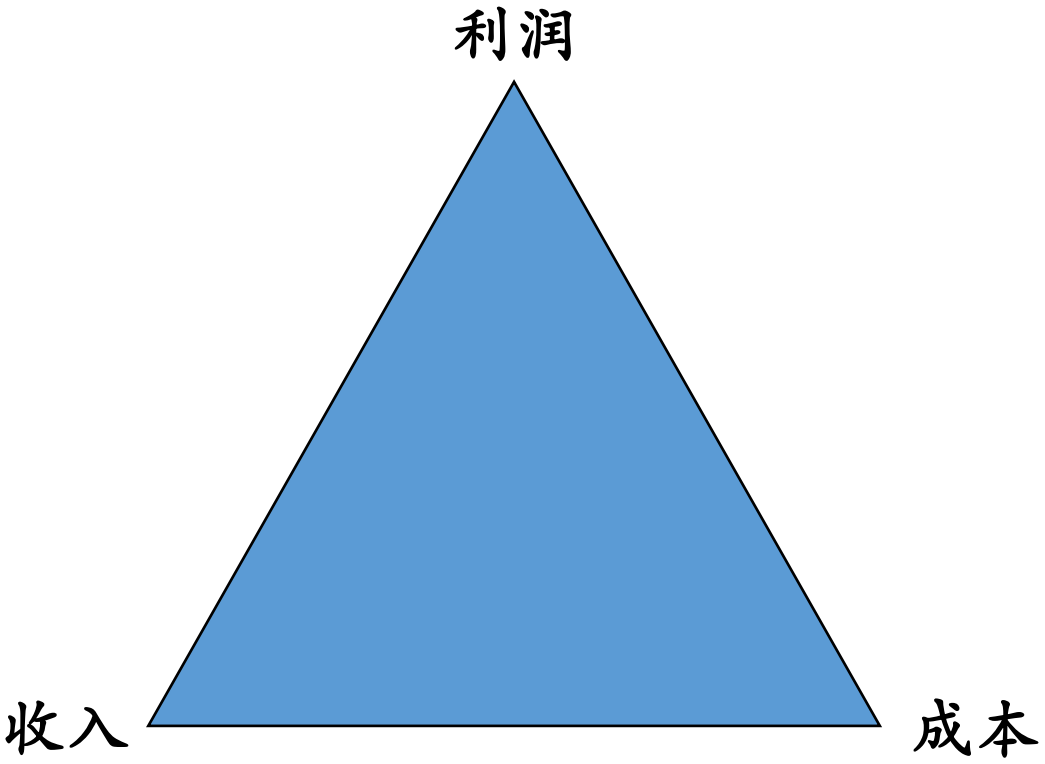
建筑模型

高达玩具



对象模型

3.1 模型与业务模型



企业的业务模型

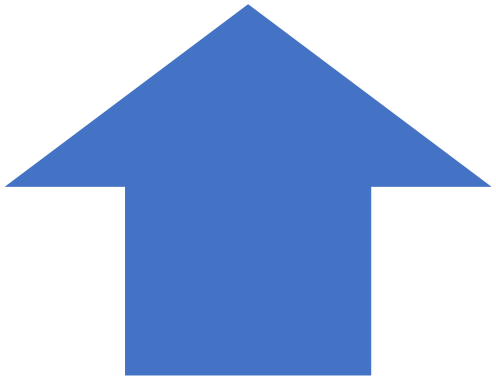
“大道至简，衍化致繁”

这个三角形可以说是一切盈利性企业的基本行为，企业为生产而投入成本，产品或服务销售后获得收入，而衡量企业业绩的最基本方法就是计算收入减去成本所得到的利润。

- 企业准备组织哪些人进行生产、销售，在什么样的渠道上销售，为此投入什么样的资源，这就是企业的生产和销售流程。
- 收入和成本都需要几张，这就是财务会计的流程。
- 对利润实现情况的衡量、盈亏原因的分析等，都体现在管理会计中。

3.2 常见的建模方法（ISO 9000模型）

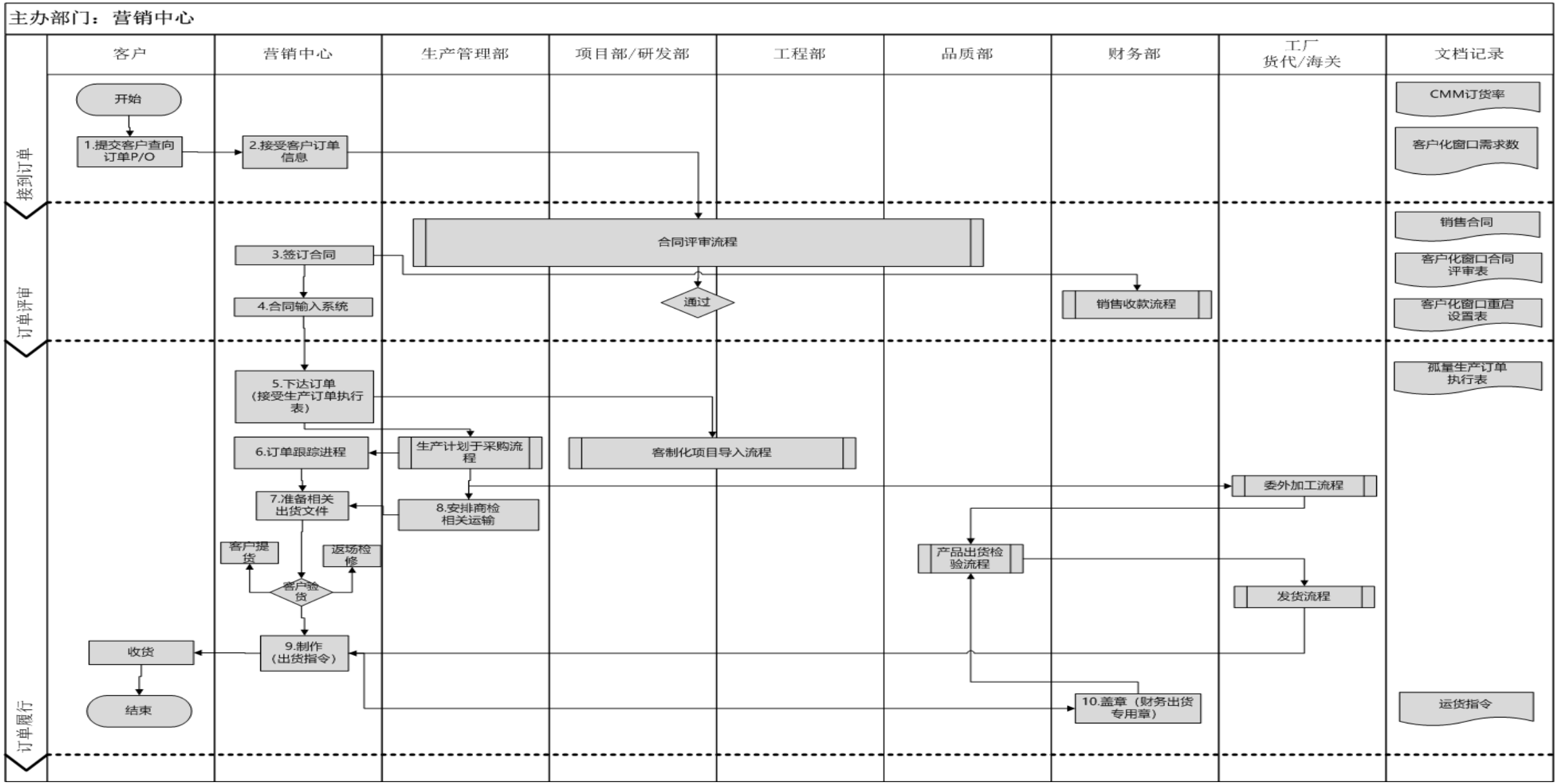
ISO 9000质量管理体系是国际标准化组织（ISO）制定的国际标准之一，是指有ISO/TC 176（国际标准化组织质量管理和质量保证技术委员会）制定的所有国际标准。该标注可以帮助组织实施并有效运行质量管理体系，是质量管理体系通用的要求和指南。



申请ISO 9000质量认证的企业，通常要绘制企业的业务流程图，流程图的样式为垂直职能带型，通常使用Visio工具进行绘制。

ISO 9000模型对业务人员非常友好，但是将其应用到软件设计领域，则会出现表达能力比较单一，对技术分析而言有所不足的问题。

3.2 常见的建模方法（ISO 9000模型）



3.2 常见的建模方法（BPMN模型）

BPMN（Business Process Model and Notation） 即**业务流程建模与标注**，是由BPMI（The Business Process Management Initiative）开发的一套**建模标准语言**。

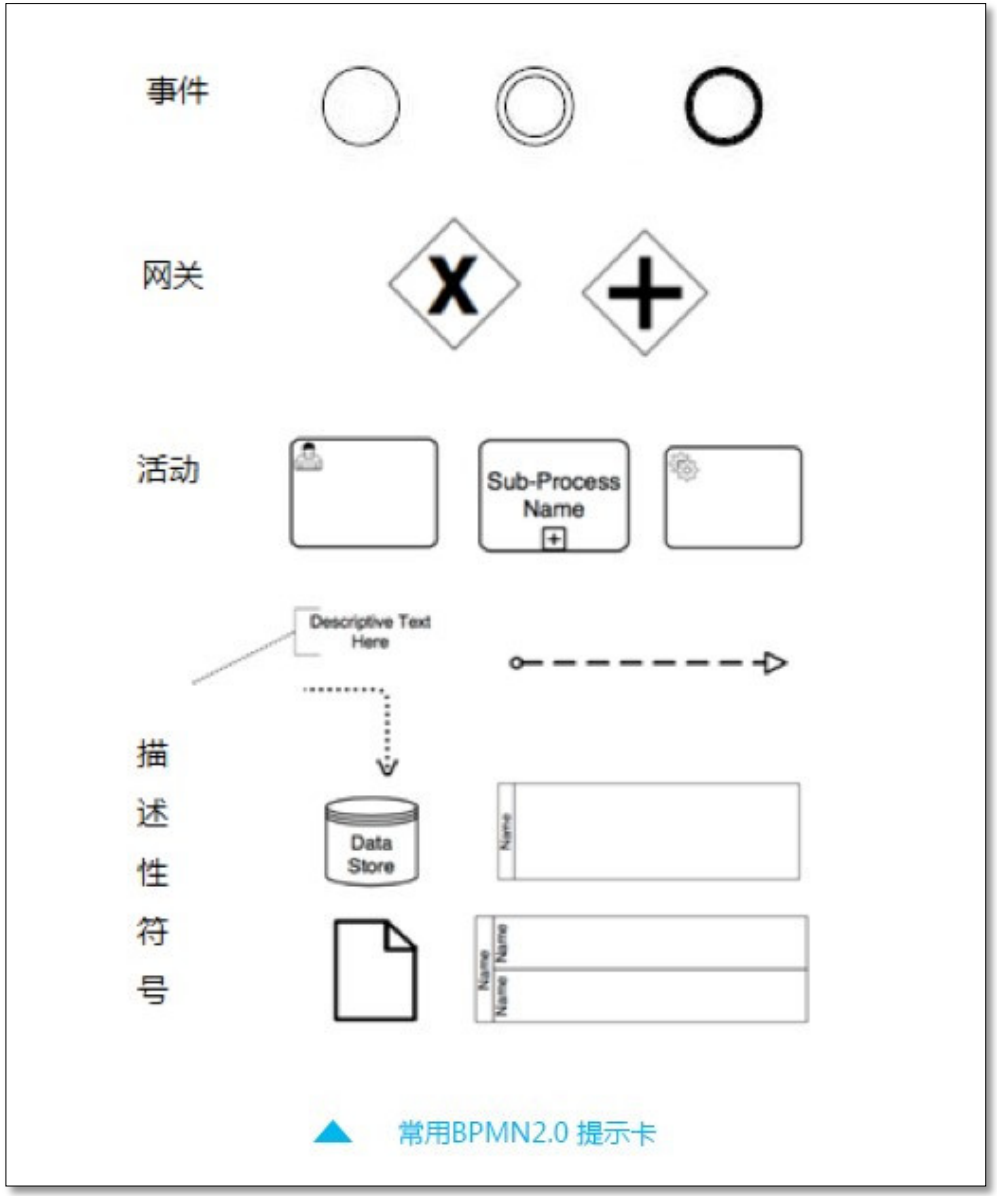
2004年5月，BPMN 1.0规范，

2011年，BPMN 2.0标准。

BPMN主要目标是为所有用户提供一些易于理解的符号，支持流程的创建、分析和实现，知道最终用户的管理和监控。

BPNM的元素的核心集合包含事件、活动和网关在内的**流对象（Flow Objects）**，含顺序流、消息流以及关联在内的**连接对象（Connecting Objects）**，含数据对象、文字注释和组在内的**人工信息（Artifacts）**，以及作为图形化容器的泳道。

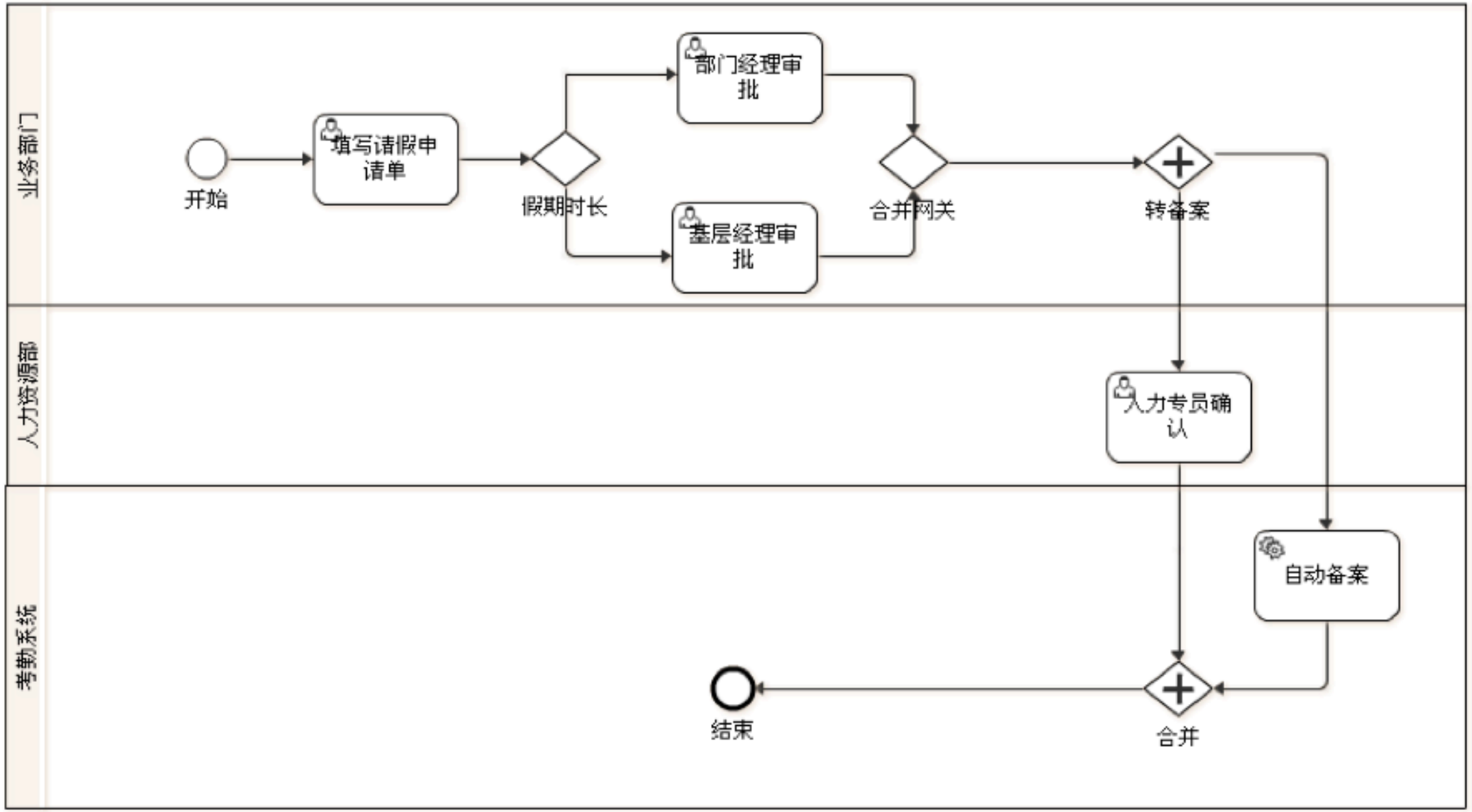
3.2 常见的建模方法（BPMN模型）



常用BPMN2.0 提示卡

序号	图元	类型	功能描述
1	开始	Start Event	标志一个流程的起始
2	填写请假申请单	User Task	填写请假表单，注明请假人、天数、原因
	部门经理审批	User Task	如果假期时长大于 3 天，由部门经理审批
	基层经理审批	User Task	如果假期时长小于 3 天，由基层经理审批
	人力专员确认	UserTask	人力专员确认
3	假期时长	ExclusiveGateway（Diverging）	互斥分叉网关，判断假期时长，用来决定下步流程分支
4	合并网关	ExclusiveGateway（Converging）	互斥合并网关，合并条件分支，重回流程主干
5	转备案	ParallelGateway（Diverging）	并行分叉网关，并行执行"人力专员确认"和"自动备案"任务
6	合并	ParallelGateway（Converging）	并行合并分支，待"人力专员确认"和"自动备案"任务都完成后，合并流程分支，重回流程主干
7	自动备案	ServiceTask	系统自动记录请假信息及审批结果，备案
8	结束	EndEvent	标志流程结束

3.2 常见的建模方法（BPMN模型）



3.2 常见建模方法（UML建模）

UML语言是非专利的第三代建模和规约语言，最佳实践是针对大规模、复杂系统进行建模。

UML体系包含了3个主要模块：

1）功能模型：从用户的角度展示系统的功能，包括用例图。

2）对象模型：采用对象、属性、操作、关联等概念展示系统的结构和基础，包括类图、对象图。

3）动态模型：展现系统的内部行为，包括序列图、活动图、状态图。

3.2 常见建模方法（UML建模）

UML对技术人员比较友好，但是对业务人员非常不友好。

业务架构的任务是搭建业务与技术之间的桥梁，所以业务架构再结构化表达方面不可或缺的工具。如果企业已经习惯了某种建模方法，需要考虑根据如下因素进行调整：

1）是否可以对原有方法进行改造以弥补缺陷。

2）原有的模型成果是否还有复用的价值。

3.2 常见建模方法（UML建模）

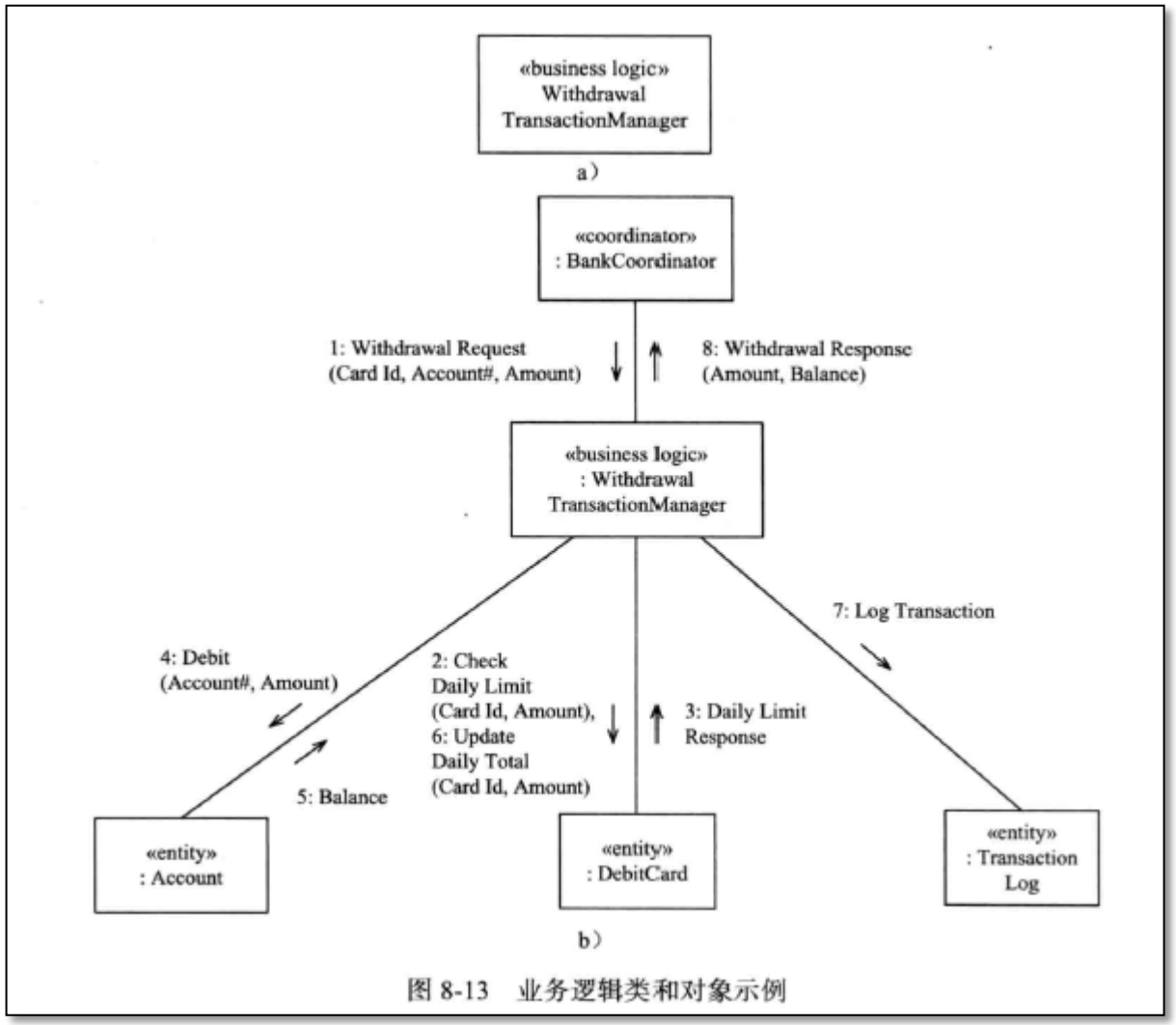


图 8-13　业务逻辑类和对象示例

例1 业务逻辑类和对象示例

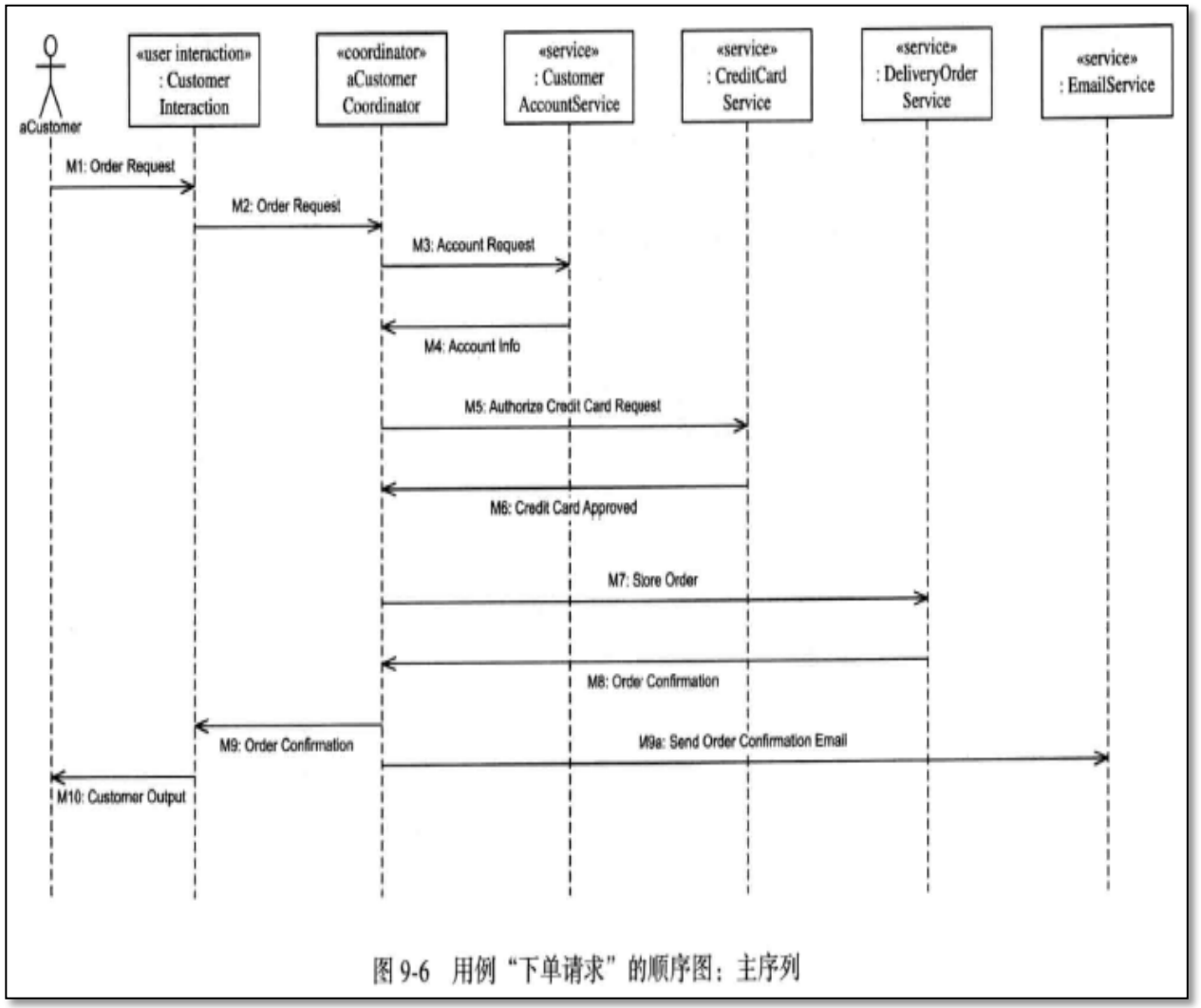


图 9-6　用例“下单请求”的顺序图：主序列

例2 用例“下单请求”的顺序图：主序列

3.2 常见建模方法（UML建模）

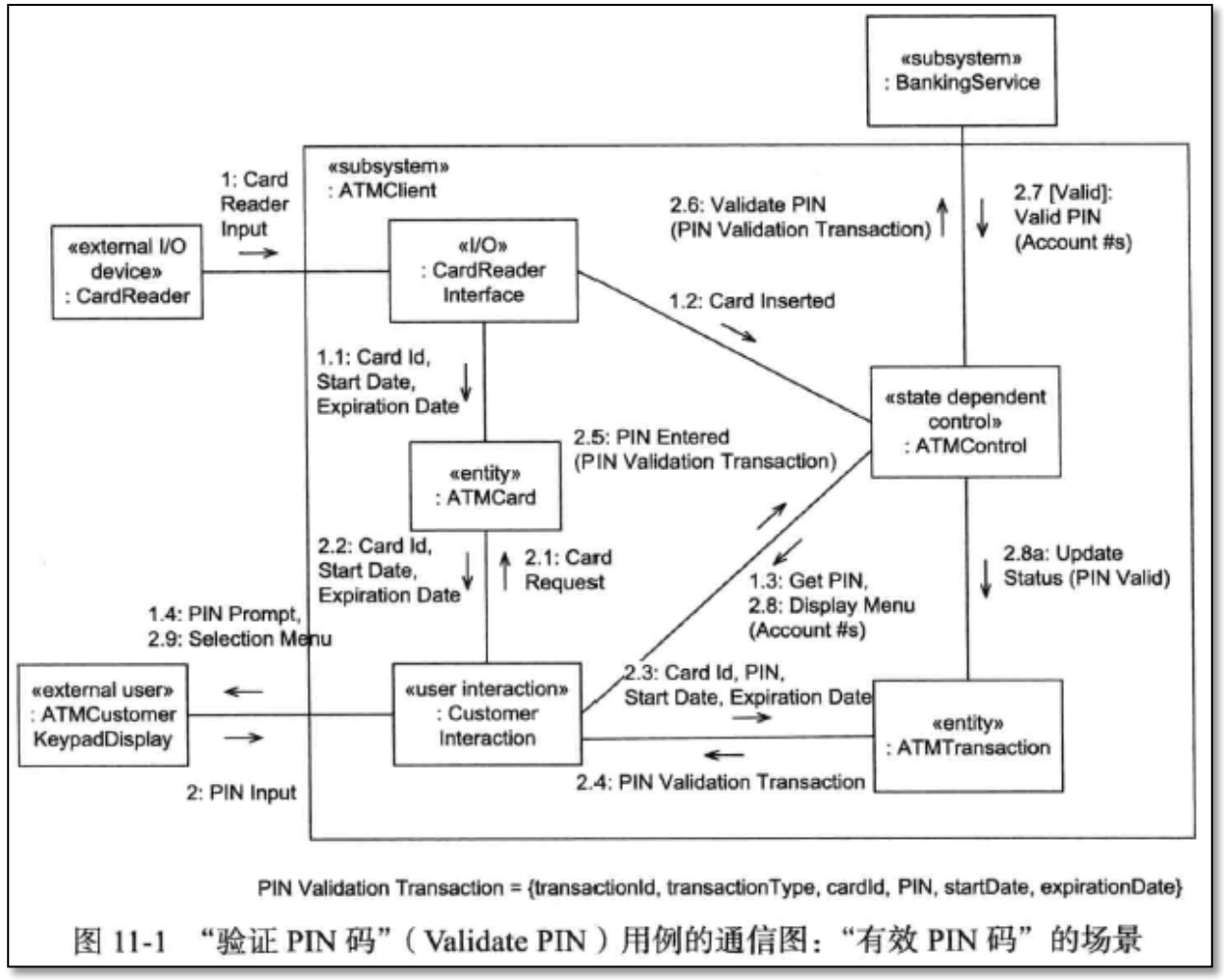


图 11-1 “验证 PIN 码”（Validate PIN）用例的通信图：“有效 PIN 码”的场景

例3 “验证PIN码”用户的通讯图：“有效PIN码”的场景

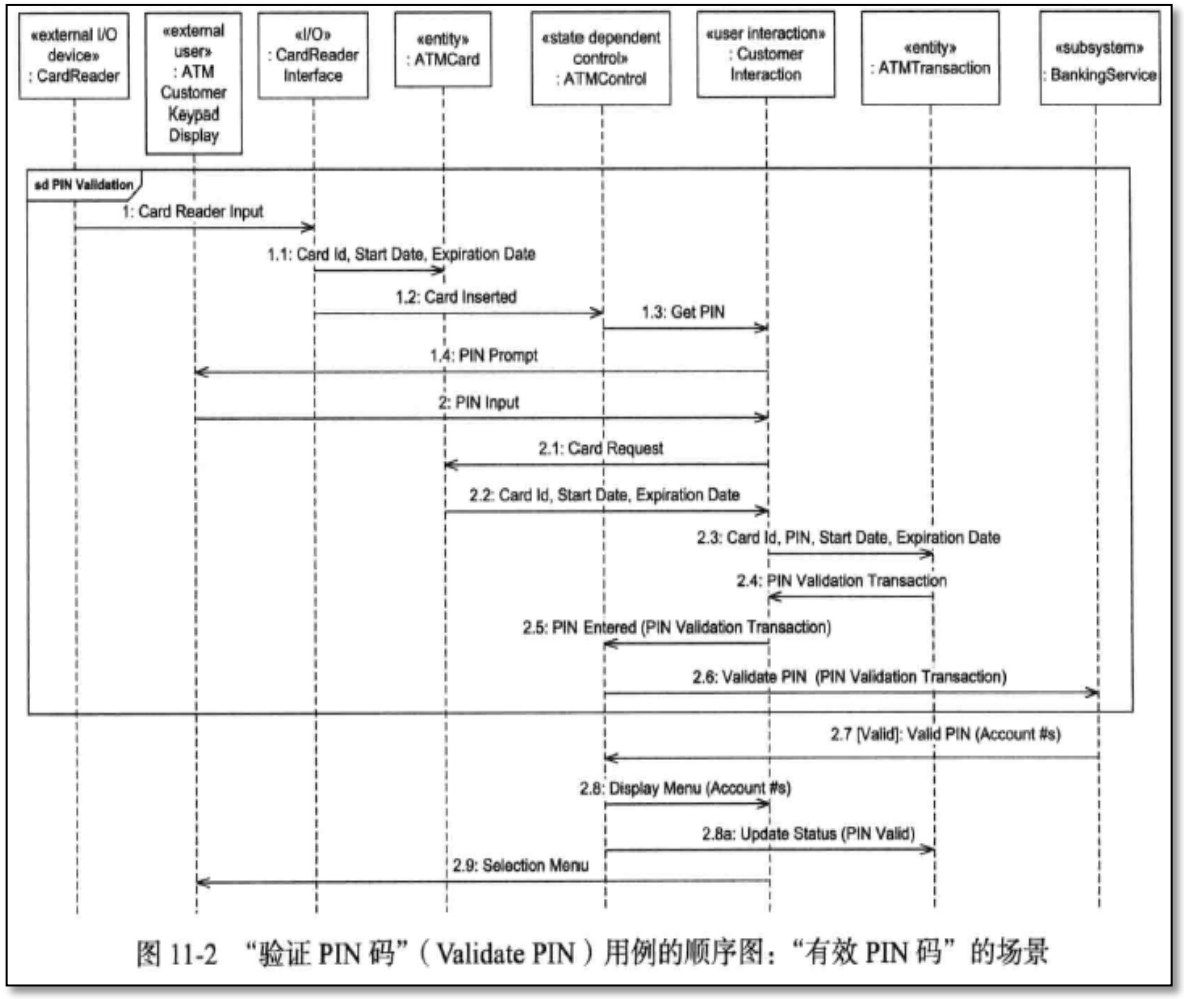
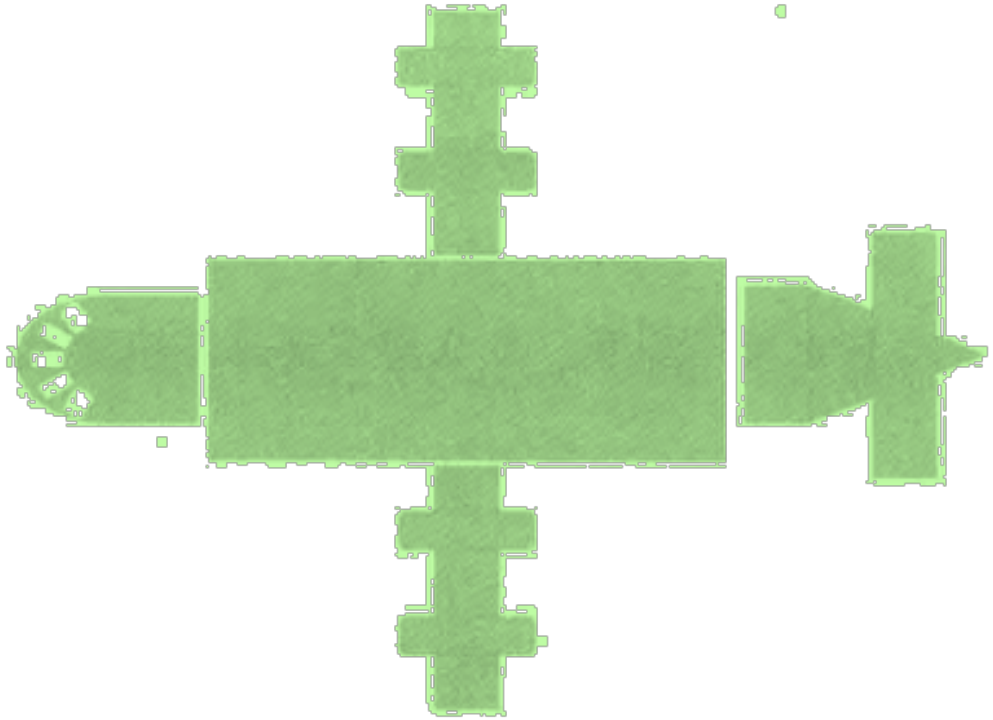


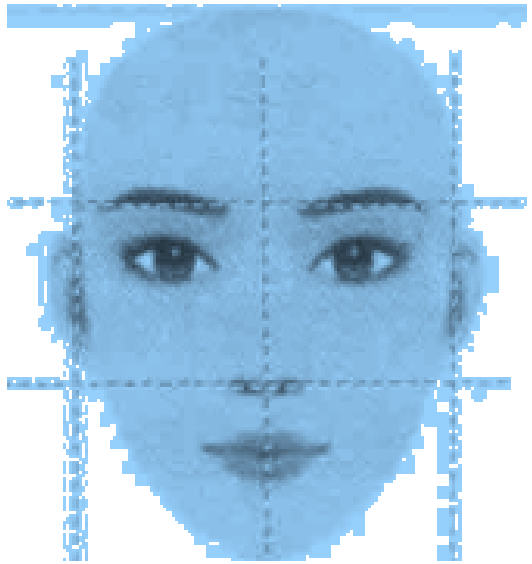
图 11-2 “验证 PIN 码”（Validate PIN）用例的顺序图：“有效 PIN 码”的场景

例4 “验证PIN码”用例的顺序图：“有效PIN码”的场景

3.3 建模原则与思维模型的应用



拼不上的大飞机



拼凑脸型

第二部分：业务架构设计篇

第4章 业务架构的设计起点

- 4.1 企业战略分析
- 4.2 对标分析
- 4.3 组织结构的影响不容忽视

第5章 业务架构的设计过程

- 5.1 价值链分析
- 5.2 行为分析：业务领域和业务流程
- 5.3 数据分析：企业级数据模型
- 5.4 组件分析：行为与数据的结合
- 5.5 业务架构的整体逻辑关系

第6章 业务架构的设计难点

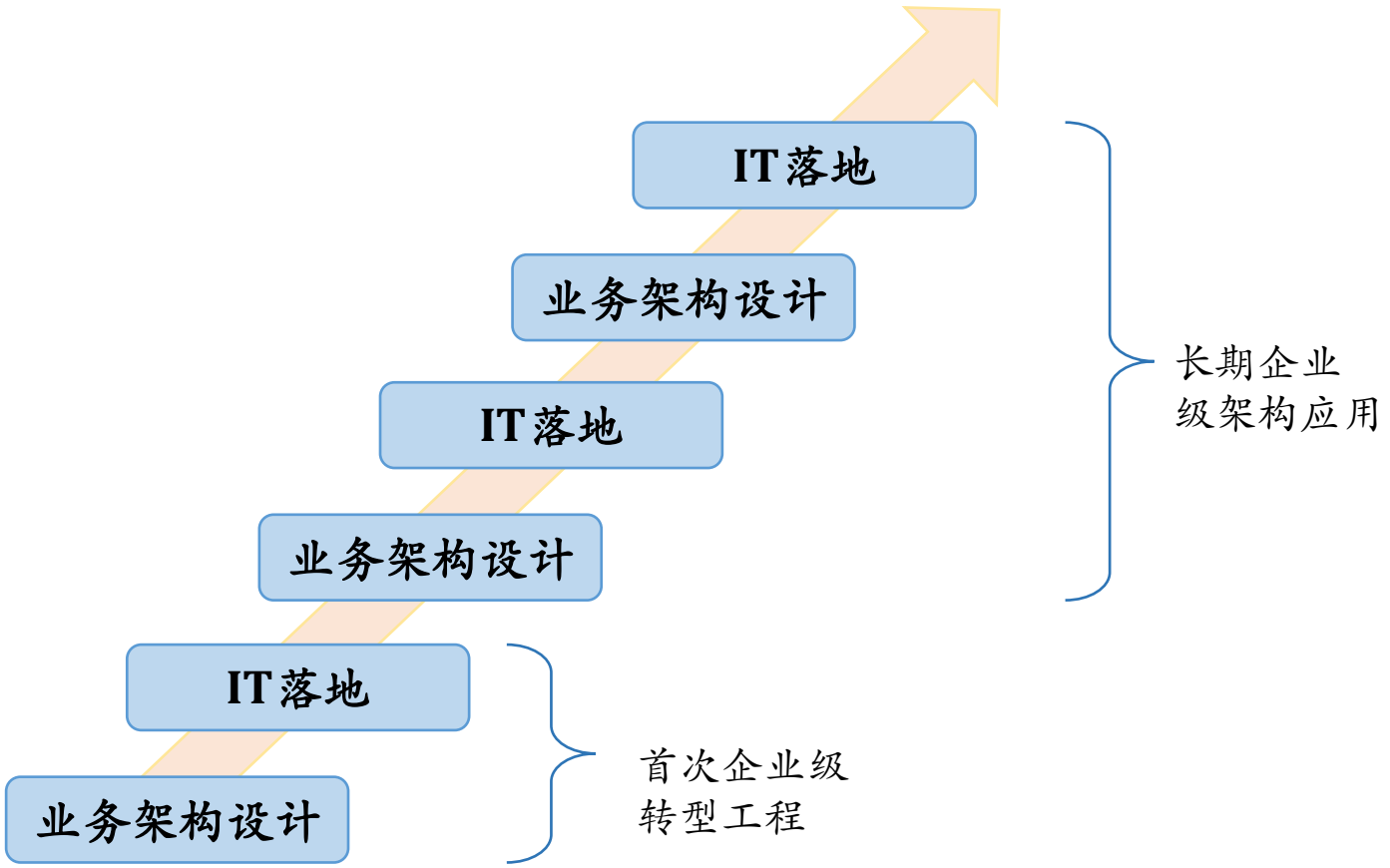
- 6.1 基本的标准化方法
- 6.2 避免“过度整合”
- 6.3 何以解忧，唯有“融合”

第7章 虚拟案例：商业银行业务架构设计

- 7.1 价值链设计
- 7.2 存款领域的模型设计
- 7.3 贷款领域的模型设计
- 7.4 跨领域的标准化
- 7.5 组件设计
- 7.6 案例总结

第4章 业务架构的设计起点

4 交替上升的业务架构设计与IT实现



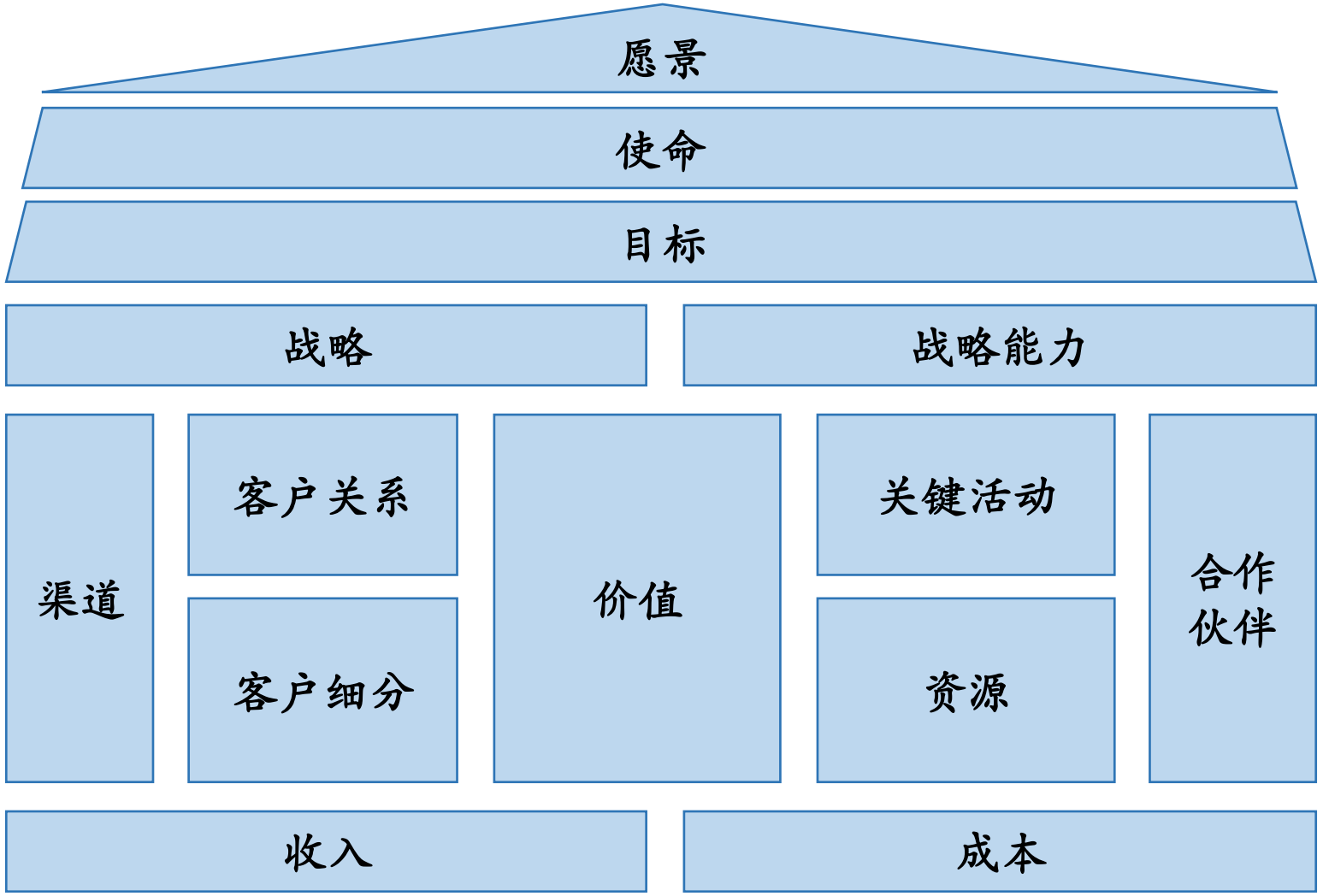
优秀的业务架构实践不能仅单纯地停留在业务分析层面，也不能只满足于业务能力的组件化聚类，而是要时刻关注新技术、新业态的变化，适时引入新理念、新方向，使之具备与时俱进的能力。

以前倡导“**业务推动技术**”，业务提需求，技术管实现，业务发展催生技术发展；

而现在，随着技术进步越来越快，“黑科技”也越来越多，“**技术引领业务**”的声音逐渐增多。但是，所谓的“技术引领业务”并不是指由IT人员直接抛出新技术给业务人员去琢磨和研究，而是指将对新技术的理解首先通过业务架构设计引发业务变化，用新技术理念推动业务模式演变，再进行IT技术开发，这才是理想的“**业务与技术融合**”，

由此可见，**业务架构意义重大**。

4.1 企业战略分析



BMGovernance 公司设计的企业战略分析模型

- 务必与企业的管理者沟通清楚
- 务必让所有参与方都达成一致共识
- 概念一致，决不能出现偏差，差之毫厘谬以千里

- 战略：是为了完成上面提到的目标所需要采取的路线、方法
- 战略能力：是为了完成这一策略所需要的能力。

- 左侧表述的是为实现战略而在客户一侧采取的行动
- 右侧所对应的3个方块则是在企业内部还需要采取的行动，关键活动是支持激活客户战略所必备的业务处理过程

- 左侧最下边的是收入，也就是说上述行动成功后，应当产生预期的收入。
- 右侧最下边的是成本，也就是说上述行动会带来合理的成本支出。收入与成本的差额就是收益了，这就是对激活存量客户策略的最终检验。

4.2 对标分析（方法论）

SWOT模型

波特五力分析模型

4.2 对标分析（方法论）

领先实践真的研究透了吗?

- 即便请咨询公司出手相助，对领先实践的研究也多是表象研究，很难充分了解其机理
- 企业的发展是有时间过程的，一个优秀的表象是经过了什么样的过程形成的，其实很少有人能说得清楚，而形成这个表象所依赖的企业内部各组成部分之间的联系和“过往”就更少有人人知道了。毕竟，即便是企业内部人士，通常也只了解自己所负责的工作，清楚企业全景的人很少
- 简单化地学习领先实践，无异于照着别人的药方抓药给自己吃。

对标之前了解清楚自己了吗?

- 企业对自己的认知其实是有限的，很多问题并没有深入挖掘根本原因和产生环境，而这两点对于形成正确的对标分析结论却是非常重要的。为了提升对标的针对性，首先还是要从自己身上下功夫，只有真正的在内部无法攻克的问题才值得去对标。
- 对业务架构设计中的企业战略分析而言，对标是为了通过对比与模仿，引入优秀的发展经验。
- 在企业战略设计方面切不可简单照搬、盲目引进，光在“别人家的事情”上下功夫，导致无效对标，甚至让对标分析“带偏”了自己。

4.3 组织结构的影响不容忽视

组织结构对业务架构设计的反作用力是极大的

康威定律

- “康威定律”告诉我们，任意一个软件都能反映出制作它的团队的组织结构，这是因为人们会以反映他们组织形式的方式工作。
- 分散的团队可能采用分散的架构生成系统，集中的团队更有可能采用单体的架构生成系统。
- 需求方的组织结构不可避免地会影响到系统的组件结构

对业务模型整体性的作用

- 当然希望能够通盘考虑整个企业，而不要因为部门利益影响组件边界的划分
- 凡是公用的部分，应该照顾到所有利益相关方的需求；凡是已实现的功能都应该对新的需求方开放并支持必要的扩展
- 一旦系统的设计边界跨越了单个部门的职能范围时，都会出现部门利益问题

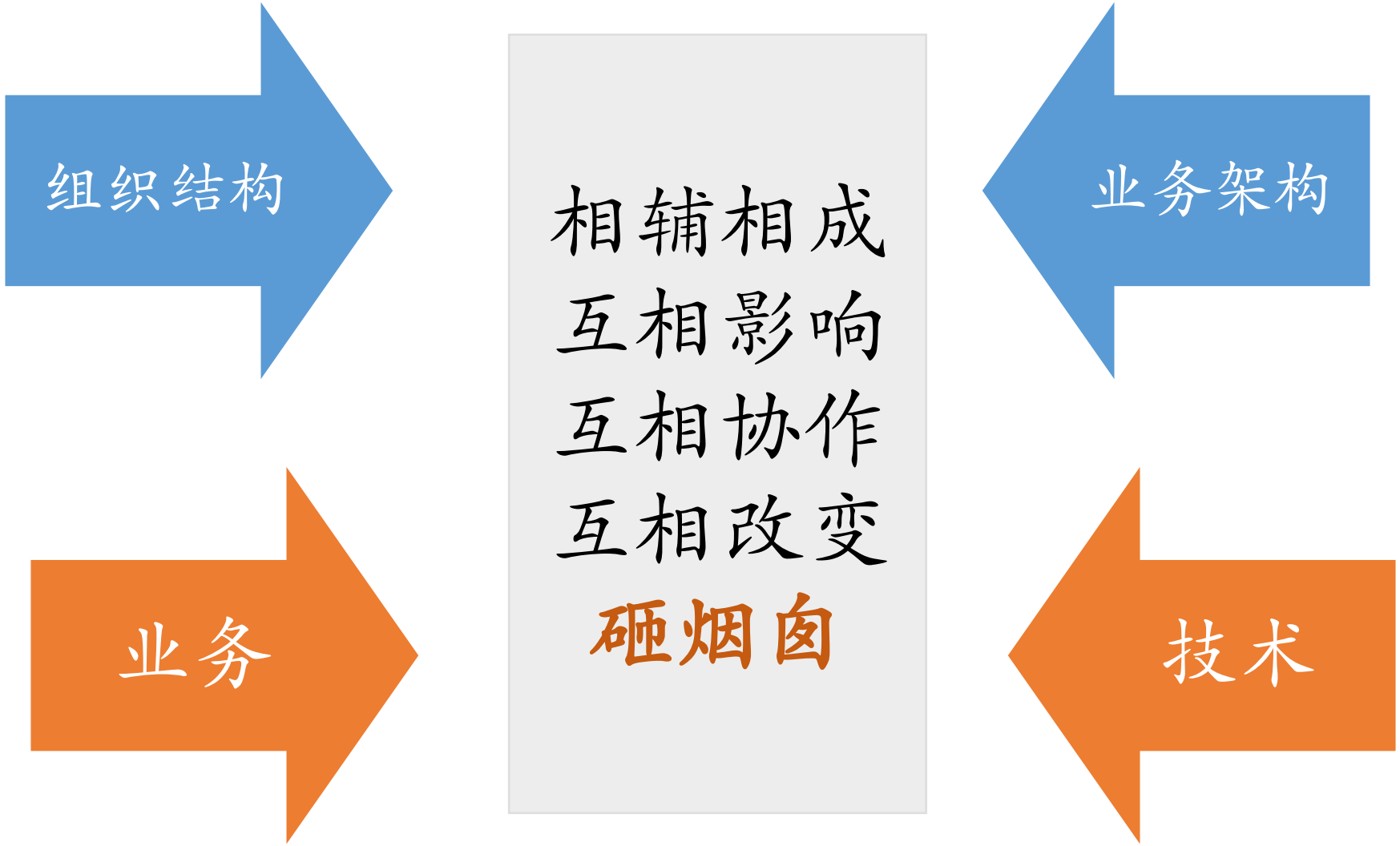
搞折中会让企业级架构失去核心架构

- 折中既无法保证系统的先进性，也无法保证用户体验，甚至还可能发生退步。
- 部门利益是做企业级的最大障碍，跨越这个障碍是对业务架构师设计能力的最高挑战。
- 接受这个挑战的还有企业文化

影响其内部沟通效率

- 壁垒森严的大型企业，沟通效率通常较低。
- 一个问题久拖不决，就会产生两种结果：一是项目组担心工期延误直接改变架构方案；二是采用非正式沟通方式，项目组间通过私下交流解决问题，而后者也极有可能是以改变架构方案为代价的。
- 这两种结果都会使得企业架构的地位变得很“尴尬”，导致架构失灵。

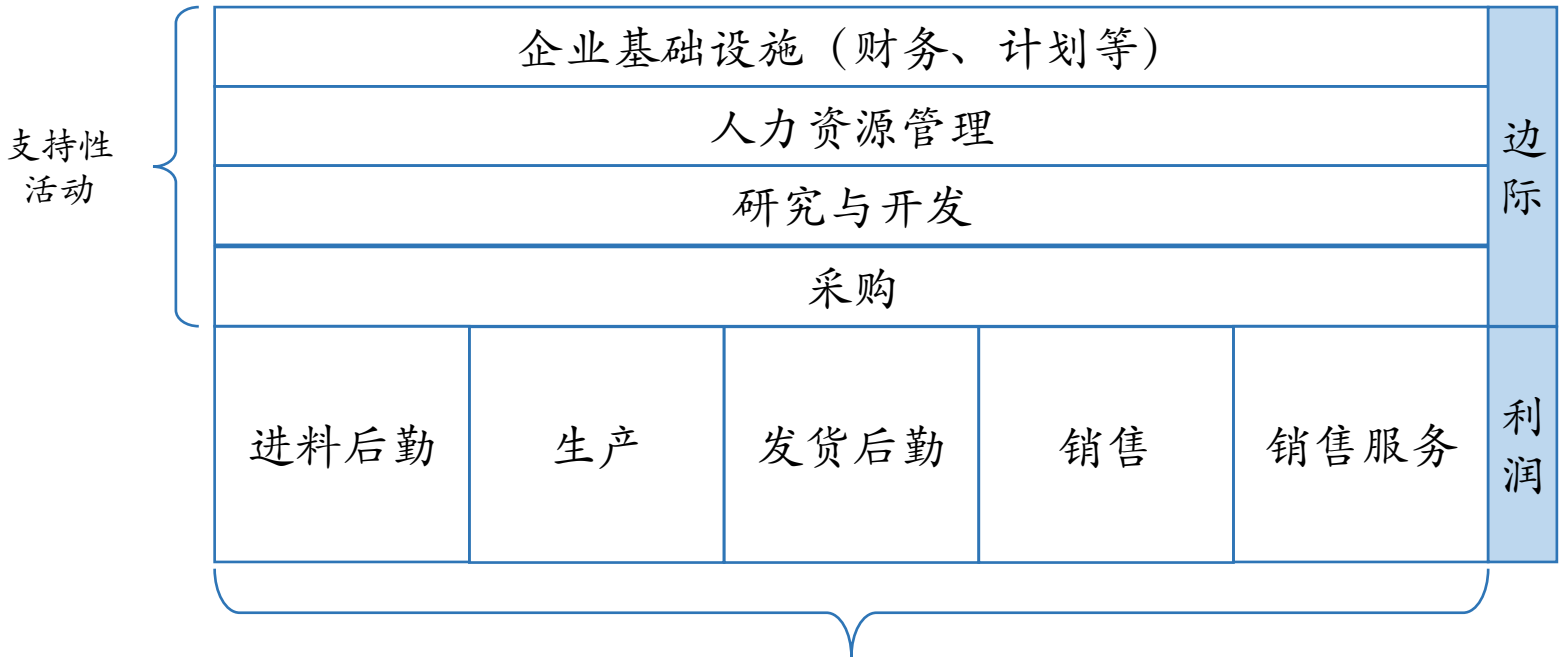
4.3 组织结构的影响不容忽视



组织结构上高度部门化的企业是很难建成一个真正的企业级业务系统
方案与组织结构要匹配，否则在落地时很可能会困难重重、面目全非

第5章 业务架构的设计难点

5.1 价值链分析

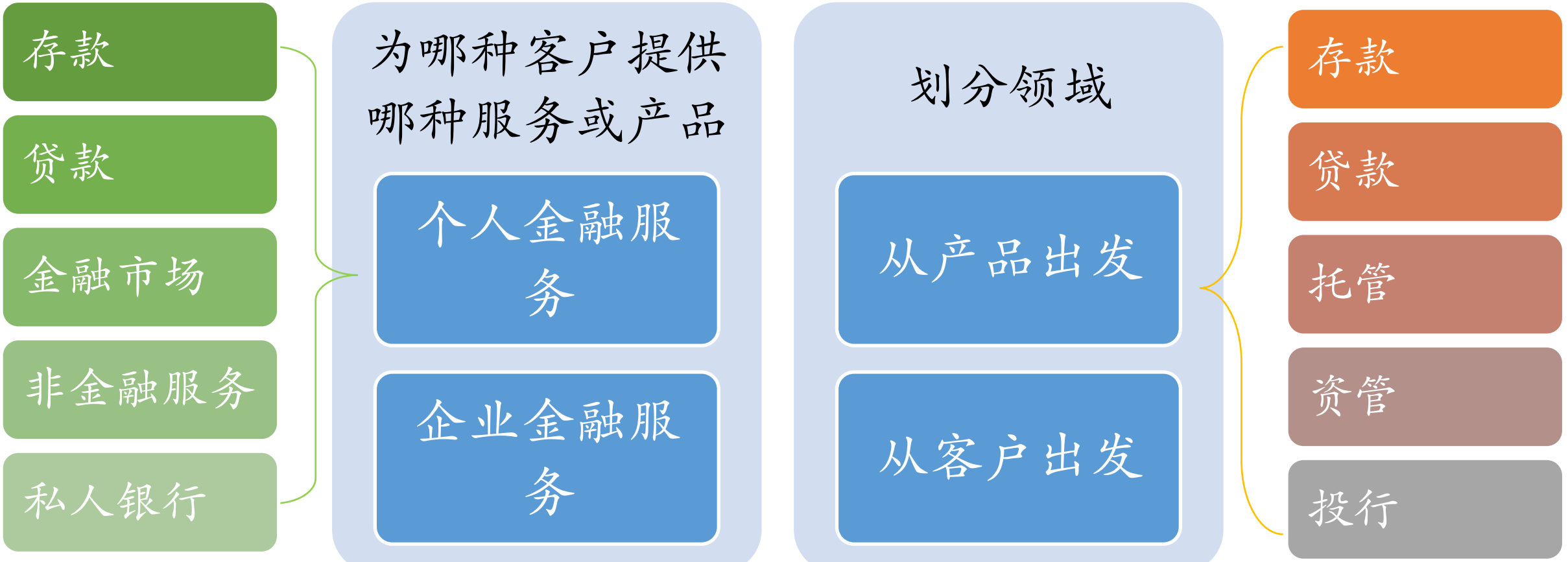


波特价值链

- 波特所定义的价值链主要是针对垂直一体化公司业务提出的，强调的是单个企业的竞争优势。
- 后来出现的全球价值链的概念提供了一种基于网络、用于分析国际性生产的地理和组织特征的分析方法，揭示了去全球产业的动态性特征。
- 价值链主要包括基本活动和支持性活动，基本活动指的是主要生产过程，支持性活动则是指对基本活动起辅助作用和维持企业基本运转的各类活动。
- 价值链主要描述企业价值的创造过程，引入价值链分析主要是为企业横向审视自身的业务能力提供分析框架。

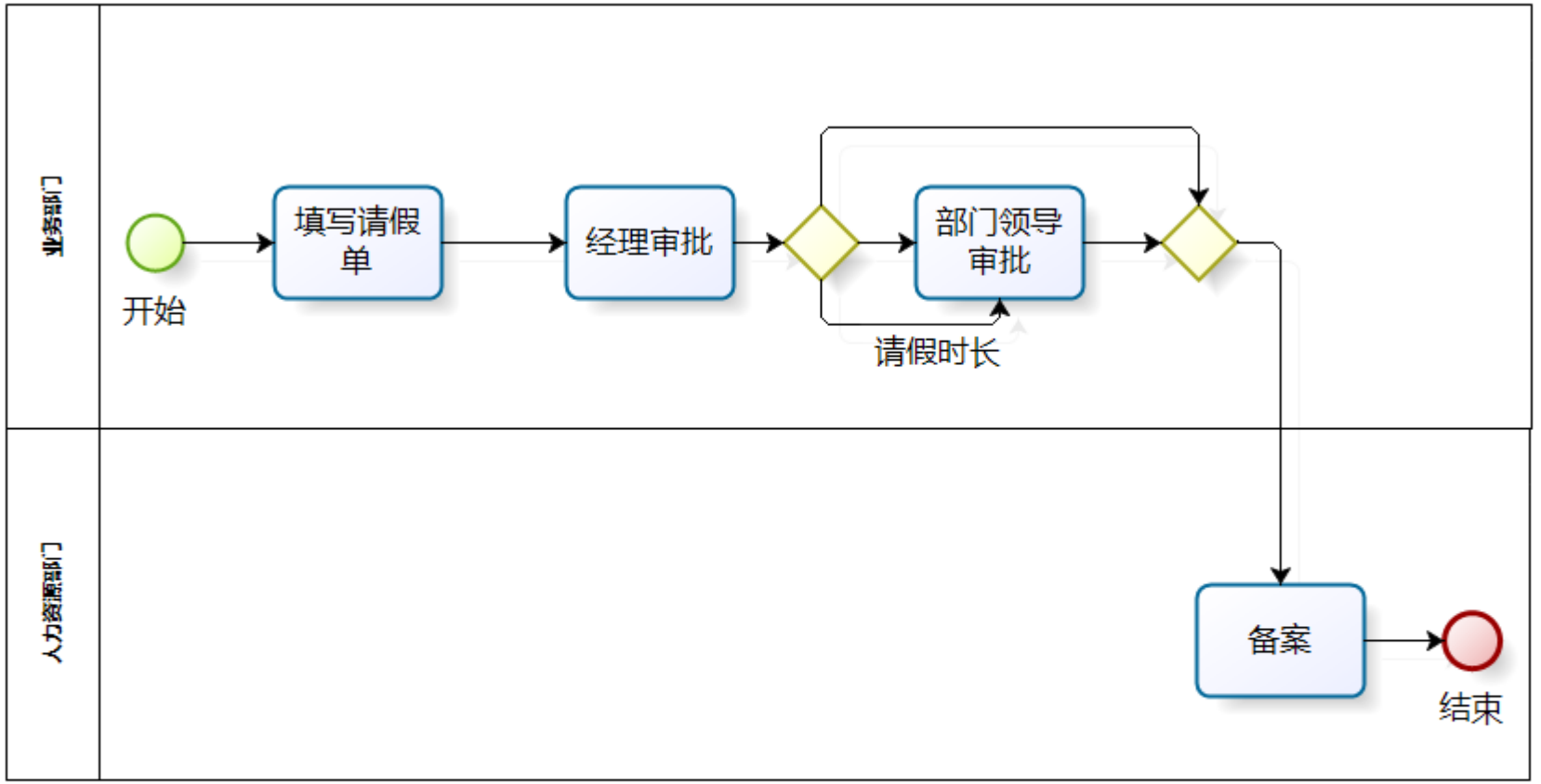
5.2 行为分析：业务领域和业务流程

1、划分业务领域

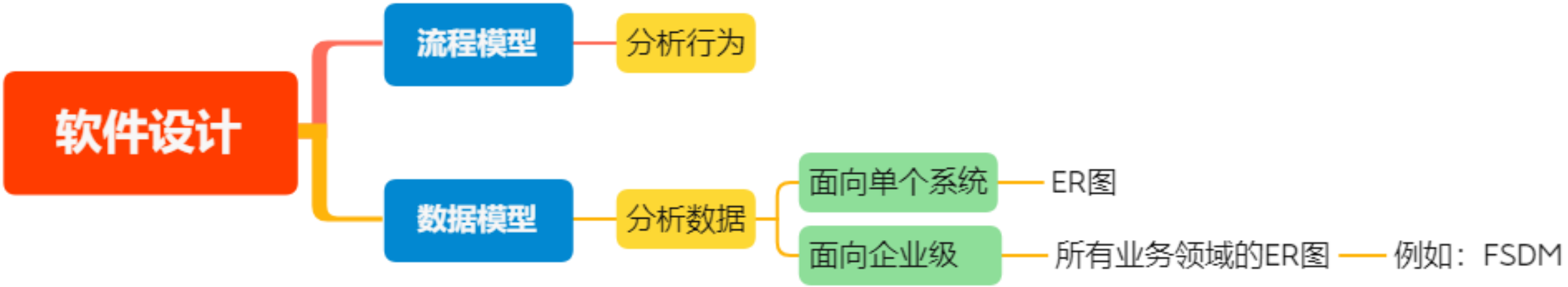


5.2 行为分析：业务领域和业务流程

2、分析业务流程



5.3 数据分析：企业级数据模型



5.3 数据分析：企业级数据模型

FSDM（Financial Services Data Model）是IBM的一个企业级数据模型，囊括了银行80%的业务数据。FSDM将数据分了九大类，分别是关系人、合约、条件、产品、地点、分类、业务方向、事件、资源项目，具体定义如下表所示。

分类名称	简称	定义
关系人	IP	银行各项业务开展过程中的相关各方，个人、机构、柜员
合约	AR	参与者之间达成的合约、合同、协议等
条件	CD	描述银行的业务正常开展所需要的前提条件、资格标准和要求
产品	PD	产品是为客户提供的、以换取利润的产品和服务，产品也包括合作伙伴或竞争对手的产品和服务，是金融机构销售或提供的可市场化的产品、组合产品和服务
地点	LO	参与人相关的所有地址，如家庭地址、公司地址、邮政信箱、电话号码、电子地址、网址等或地理位置区域
分类	CL	适用于其他数据概念的分类或者分组
业务方向	BD	银行或参与人开展业务所在的环境和方式
事件	EV	指参与人和银行的交互，以及银行内部的业务交互，它包含最详细的行为和交易数据，例如存款、提款、付款、信用/借记卡年费、利息和费用、投诉、查询、网上交易等
资源项目	RI	指银行有形或无形的有价值的资源项目，是银行拥有、管理、使用的，或支持特定业务目的的资源项目

5.3 数据分析：企业级数据模型

核心内容

建模阶段：工具筛查

生产管控阶段：严防死守

数据实体、数据属性唯一性

实施步骤

对业务数据全面建模

对生产严格管理

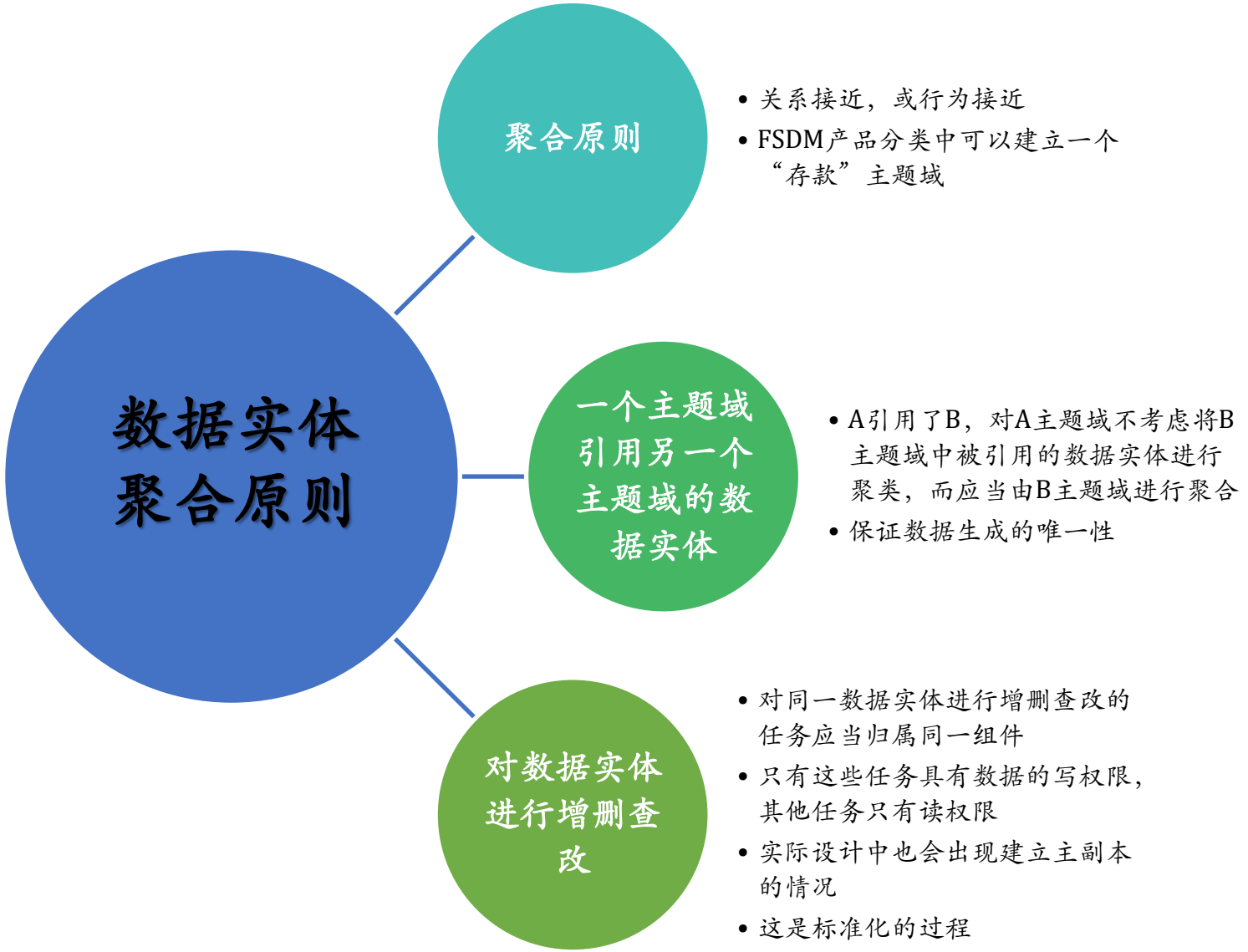
对历史数据进行处理

5.4 组件分析：行为与数据的结合

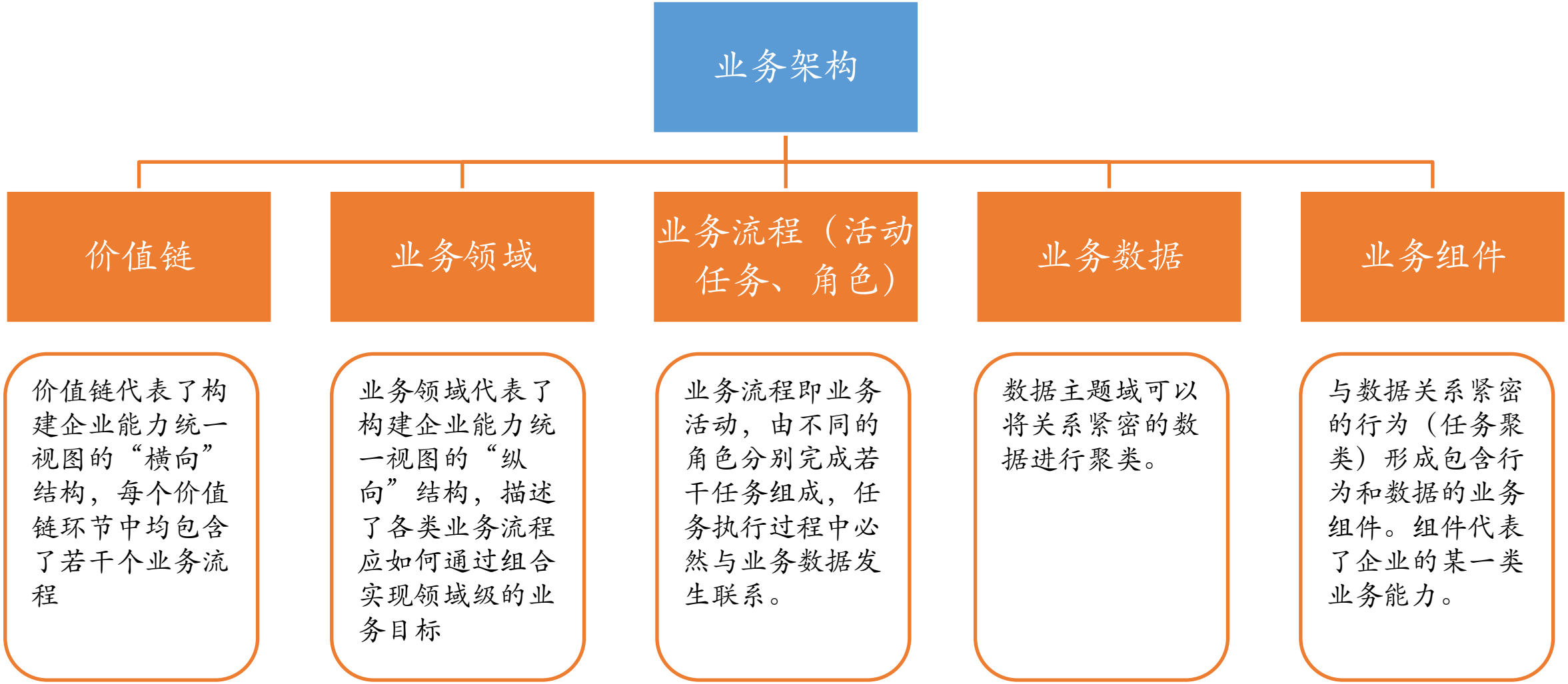
基本工作流：

流程模型表达的是“处理”，数据模型表达的是“输入”和“输出”，合起来就是计算机的基本工作流，这在大部分软件设计方法论中都是共识。数据模型和流程模型的组合，可以清楚地描述出，什么样的事件或条件可以触发一组业务活动，业务活动需要的输入有哪些，业务流程的处理规则是什么，经过业务流程的处理，输出又有哪些。

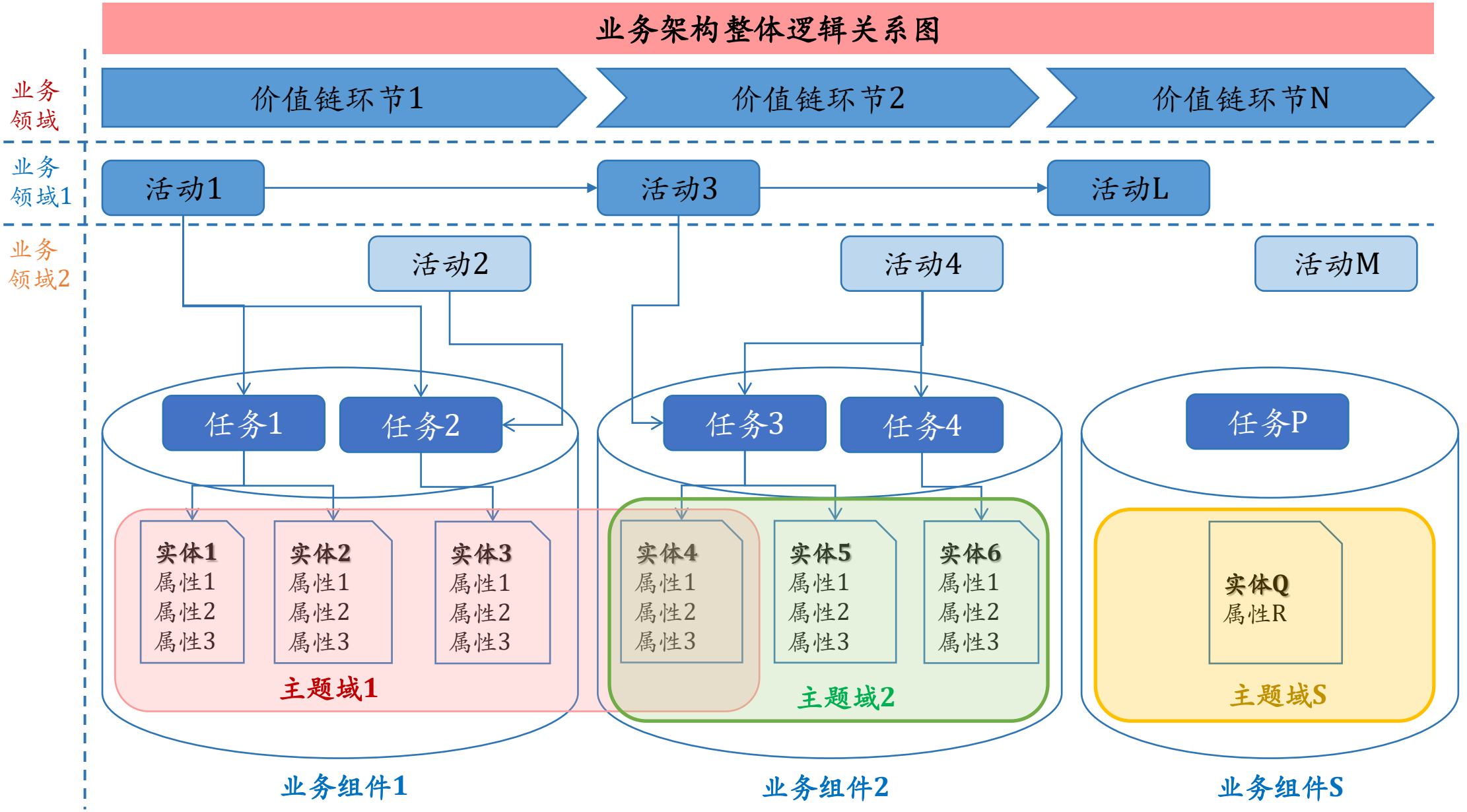
5.4 组件分析：行为与数据的结合



5.5 业务架构的整体逻辑关系

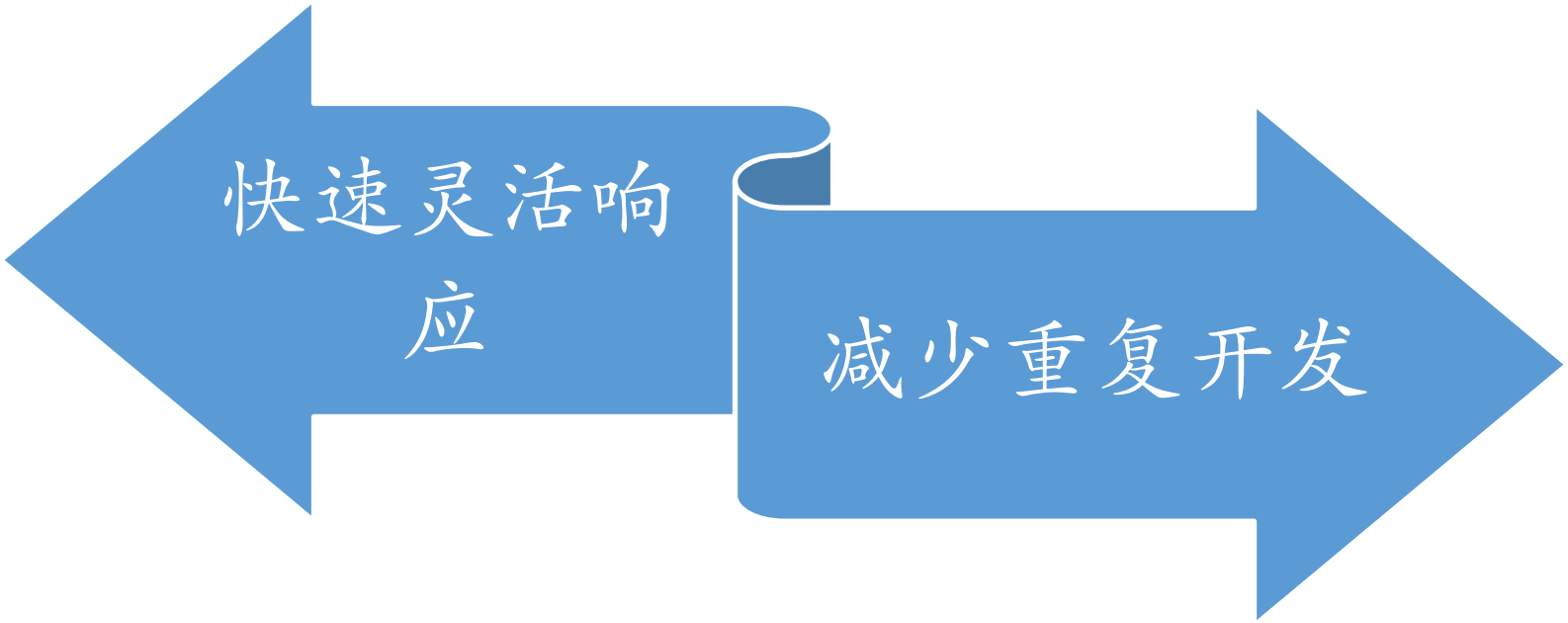


5.5 业务架构的整体逻辑关系

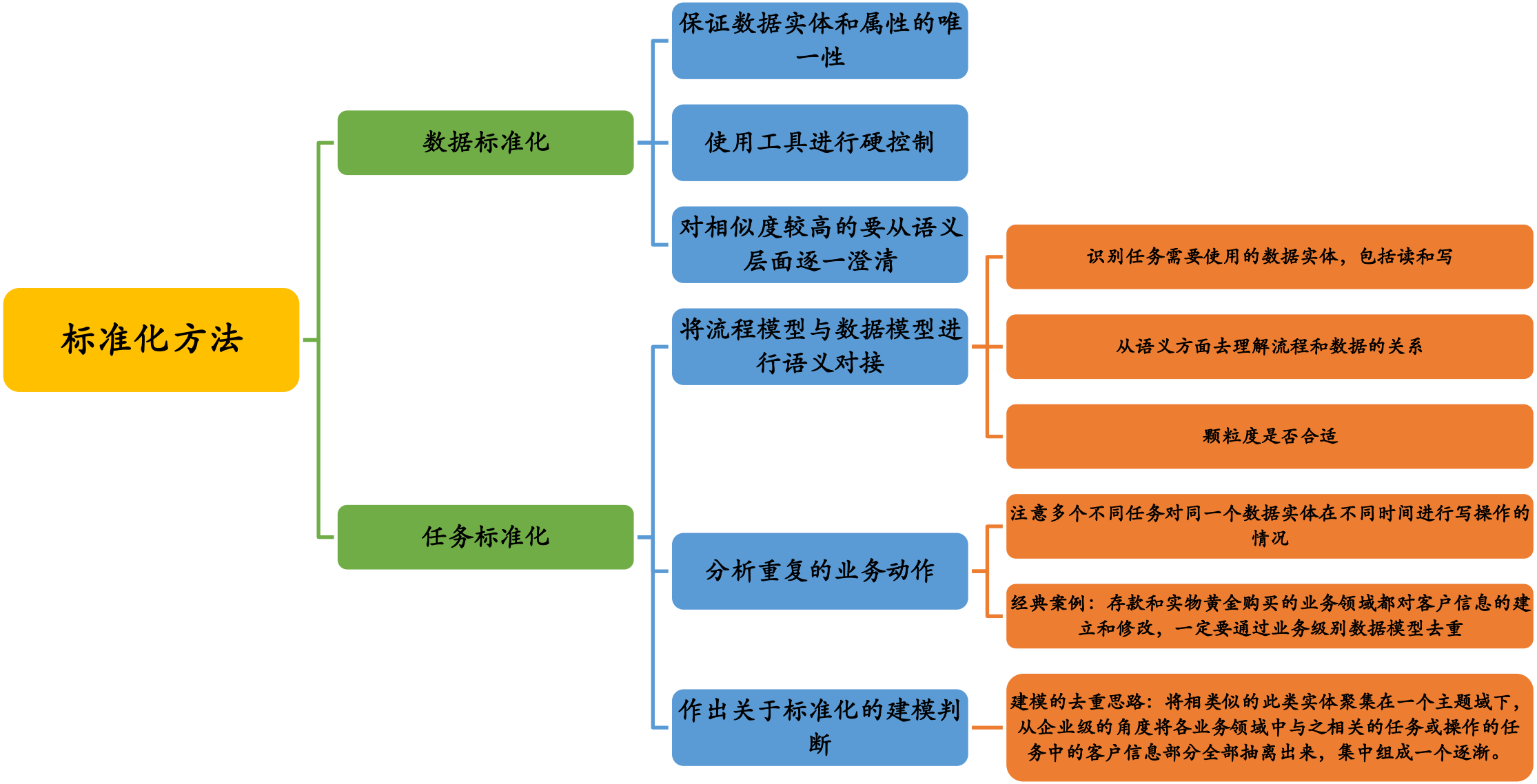


第6章 业务架构的设计难点

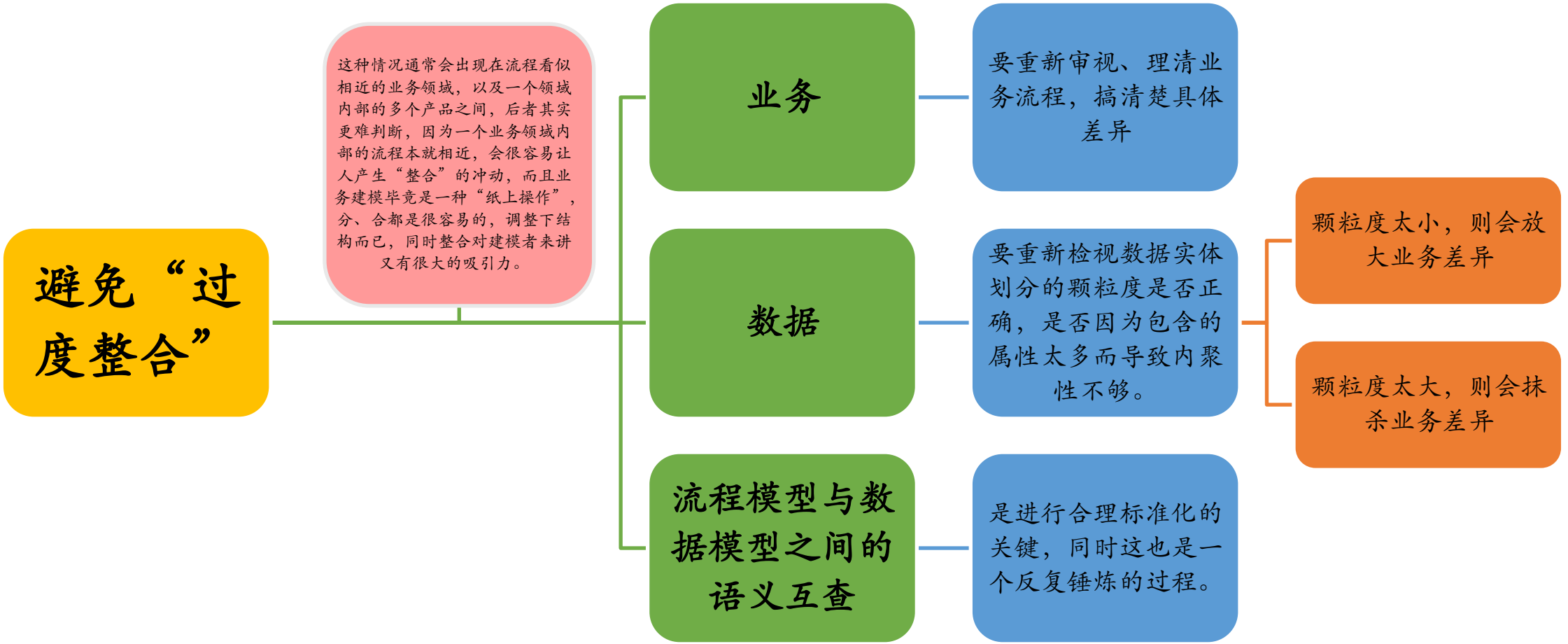
企业级转型工程的目标



6.1 基本的标准化方法


标准化方法
数据标准化
保证数据实体和属性的唯一性
使用工具进行硬控制
对相似度较高的要从语义层面逐一澄清
任务标准化
将流程模型与数据模型进行语义对接
识别任务需要使用的数据实体，包括读和写
从语义方面去理解流程和数据的关系
颗粒度是否合适
分析重复的业务动作
注意多个不同任务对同一个数据实体在不同时间进行写操作的情况
经典案例：存款和实物黄金购买的业务领域都对客户信息的建立和修改，一定要通过业务级别数据模型去重
作出关于标准化的建模判断
建模的去重思路：将相类似的此类实体聚集在一个主题域下，从企业级的角度将各业务领域中与之相关的任务或操作的任务中的客户信息部分全部抽离出来，集中组成一个逐渐。

6.2 避免“过度整合”


避免“过度整合”
这种情况通常会出现在流程看似相近的业务领域，以及一个领域内部的多个产品之间，后者其实更难判断，因为一个业务领域内部的流程本就相近，会很容易让人产生“整合”的冲动，而且业务建模毕竟是一种“纸上操作”，分、合都是很容易的，调整下结构而已，同时整合对建模者来讲又有很大的吸引力。
业务
要重新审视、理清业务流程，搞清楚具体差异
数据
要重新检视数据实体划分的颗粒度是否正确，是否因为包含的属性太多而导致内聚性不够。
颗粒度太小，则会放大业务差异
颗粒度太大，则会抹杀业务差异
流程模型与数据模型之间的语义互查
是进行合理标准化的关键，同时这也是一个反复锤炼的过程。

6.3 何以解忧，唯有“融合”

如何标准化？

- 建模者经验
- 完善的业务资料
- 经验丰富的业务人员
- 需要一套清晰的业务与IT的架构映射关系做指导

业务与科技的融合

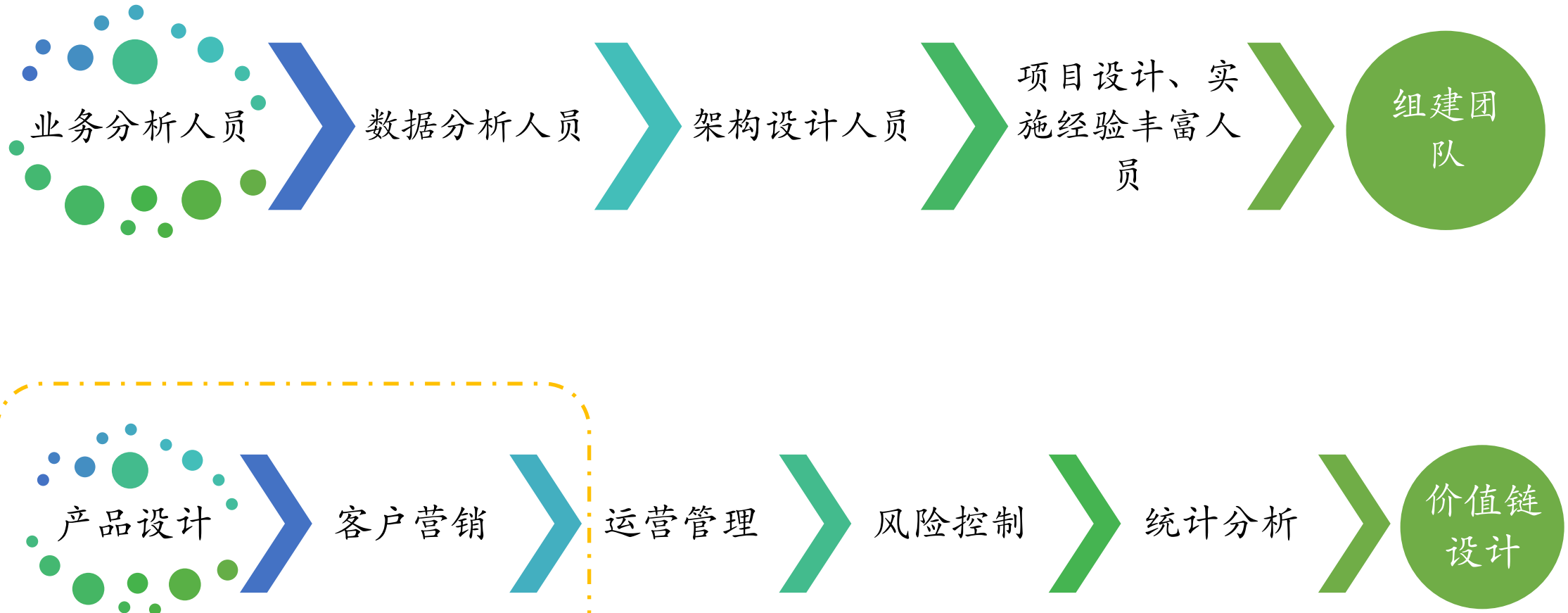
- 优秀国外金融机构数字化转型引入15%-20%技术人员到业务部门
- 国外一般金融机构技术人员占比8%
- 国内不足4%

意义

这个阶段的调整代价最低，如不合理的设计一旦传导到开发生，将会产生高昂的纠错成本

第7章 虚拟案例：商业银行业务架构设计

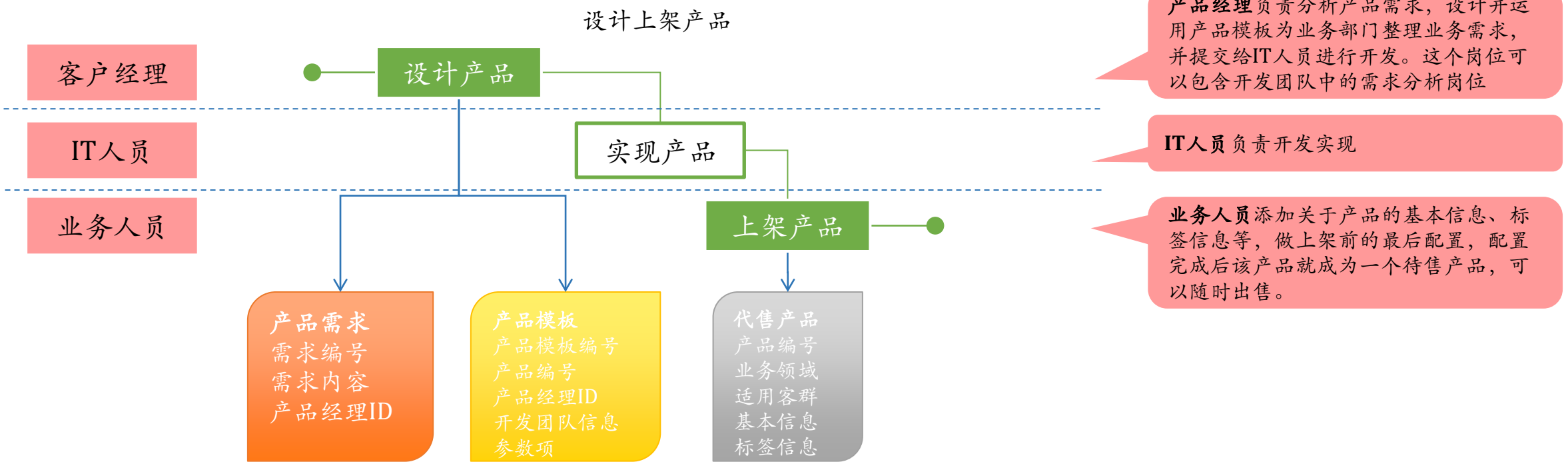
7.1 价值链设计


业务分析人员
数据分析人员
架构设计人员
项目设计、实施经验丰富人员
组建团队
产品设计
客户营销
运营管理
风险控制
统计分析
价值链设计
本案例只考虑前2个环节

7.2 存款领域的模型设计

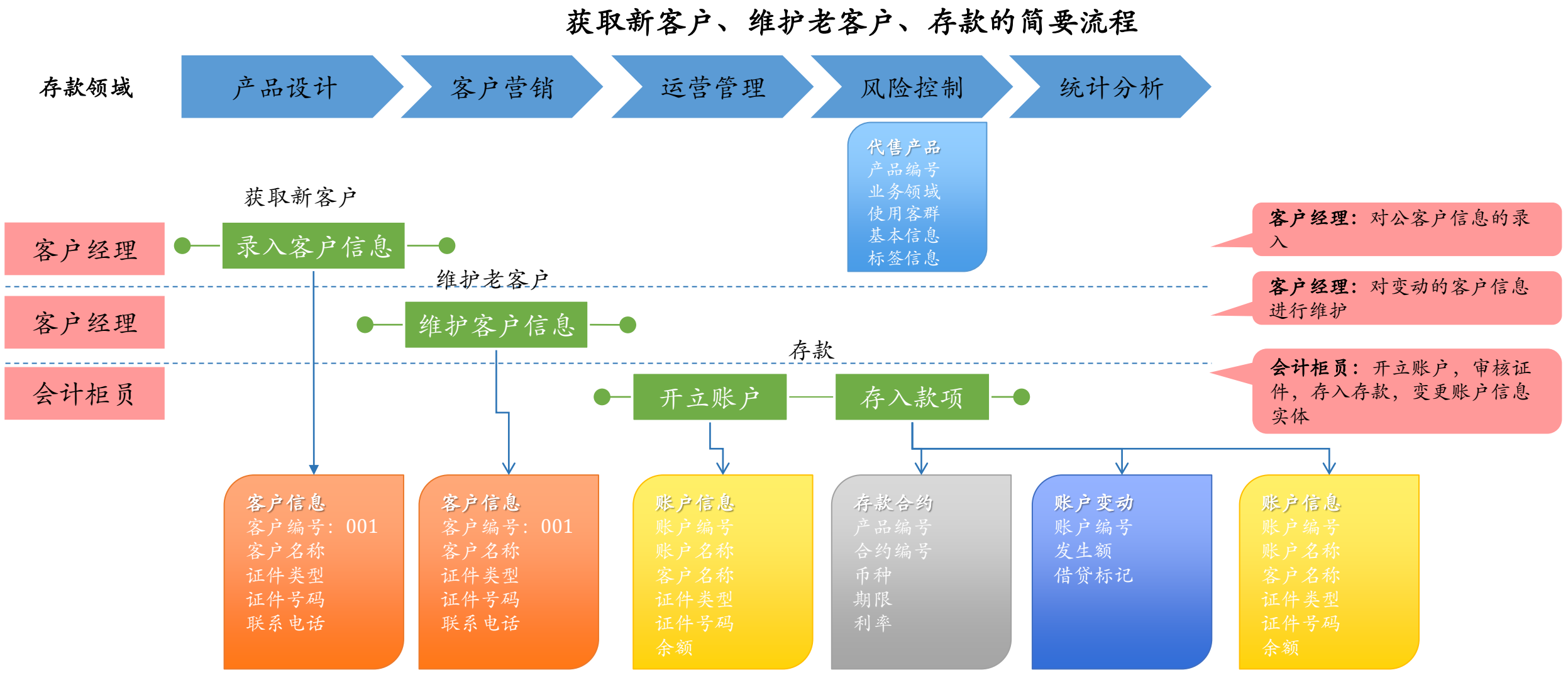
1.产品设计环节的分析

存款领域产品设计流程图



7.2 存款领域的模型设计

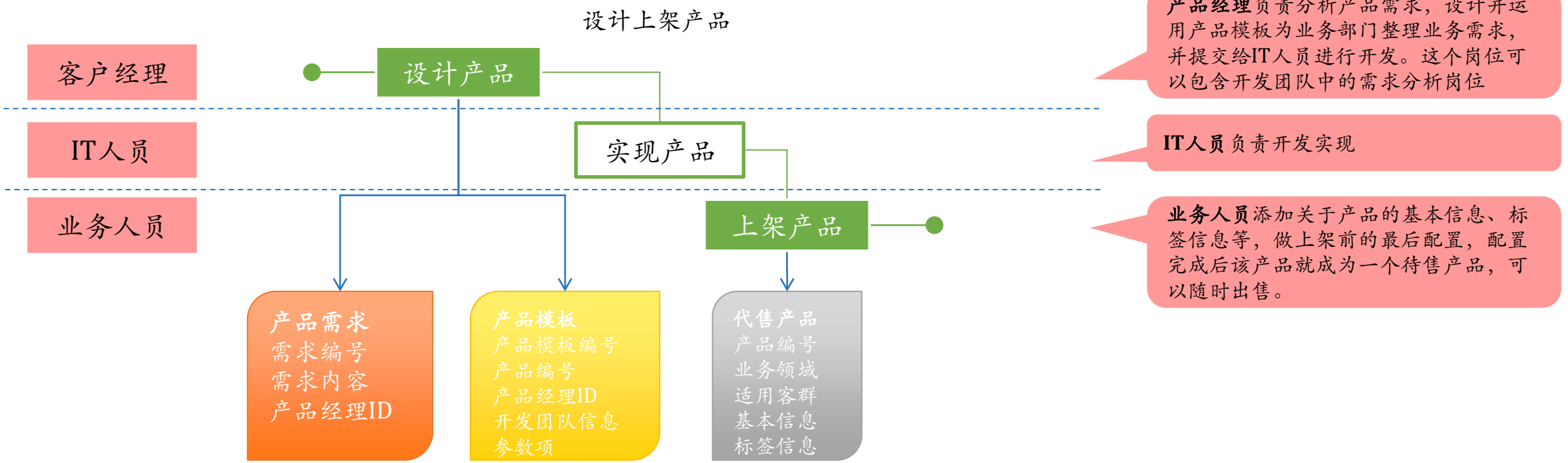
2.客户营销环节的分析



7.3 贷款领域的模型设计

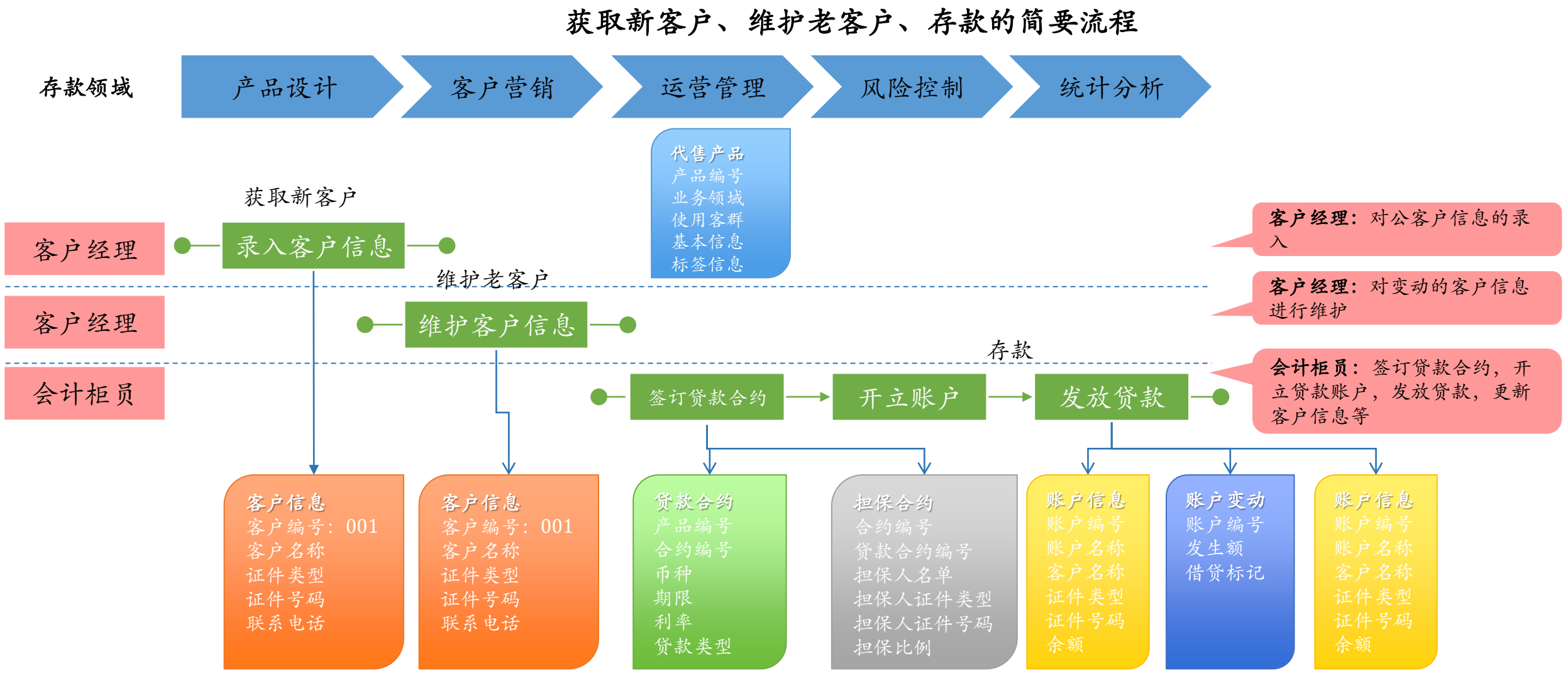
1.产品设计环节的分析

贷款领域产品设计流程图



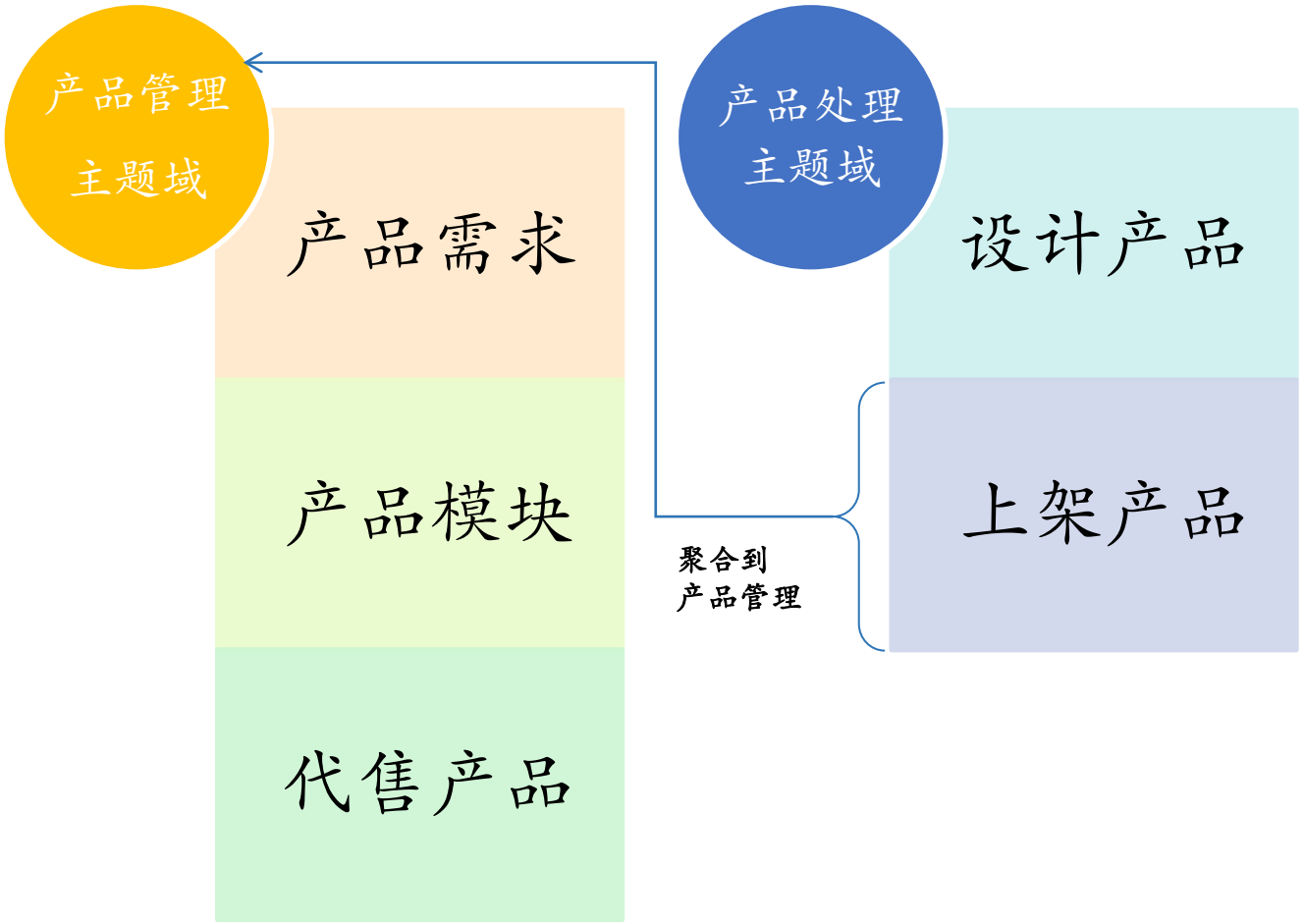
7.3 贷款领域的模型设计

2.客户营销环节的分析



7.4 跨领域的标准化

1.产品设计环节的标准化与组件分析

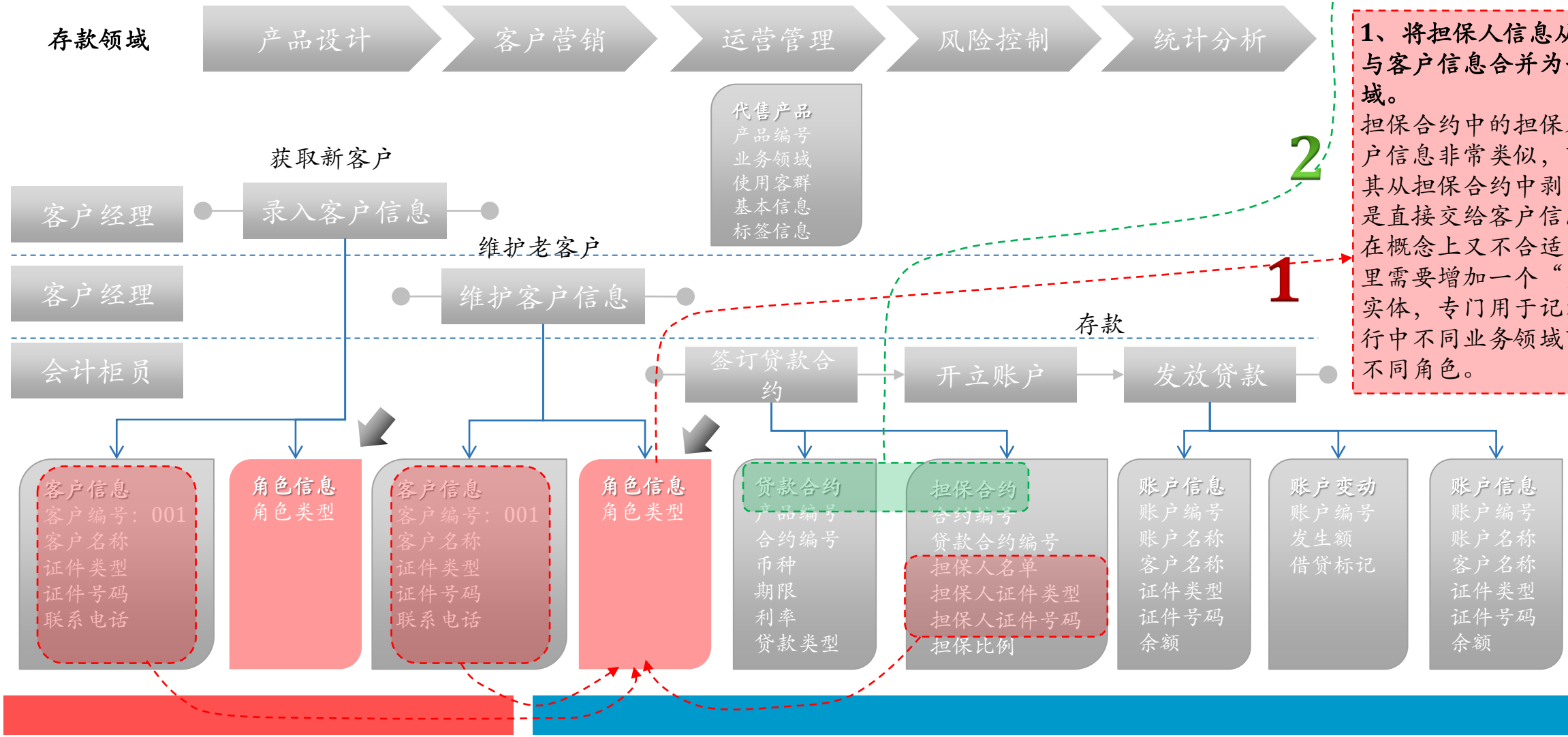


7.4 跨领域的标准化

2.客户营销环节的标准化与组件分析

2、可以将存款合约从“存入款项”任务中分离出来，建立一个签订存款合约的任务。
可以让存款和贷款均引用。

获取新客户、维护老客户、存款的简要流程

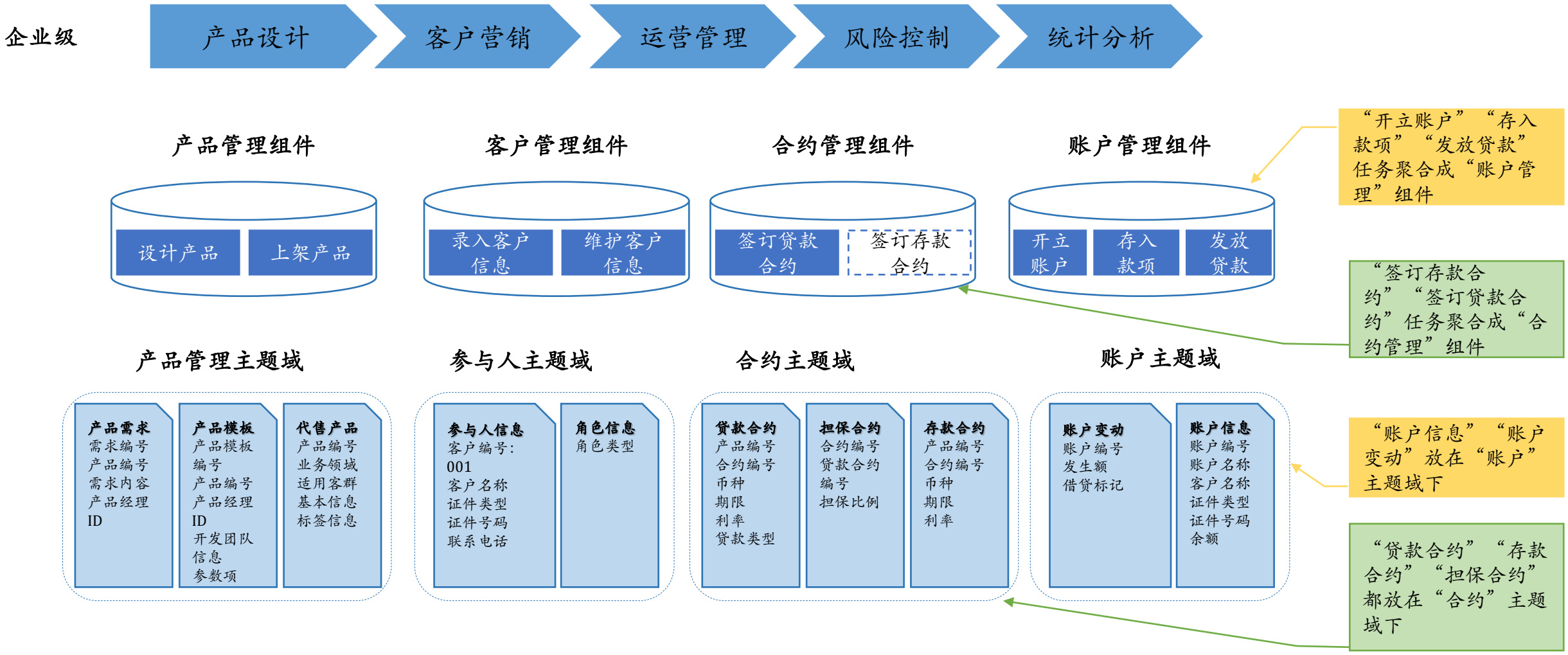


1、将担保人信息从合约中分离与客户信息合并为一个主题域。
担保合约中的担保人信息与客户信息非常类似，可以考虑将其从担保合约中剥离出来。但是直接交给客户信息实体处理在概念上又不合适，因此，这里需要增加一个“角色信息”实体，专门用于记录客户在银行中不同业务领域可能承担的不同角色。

7.5 组件设计

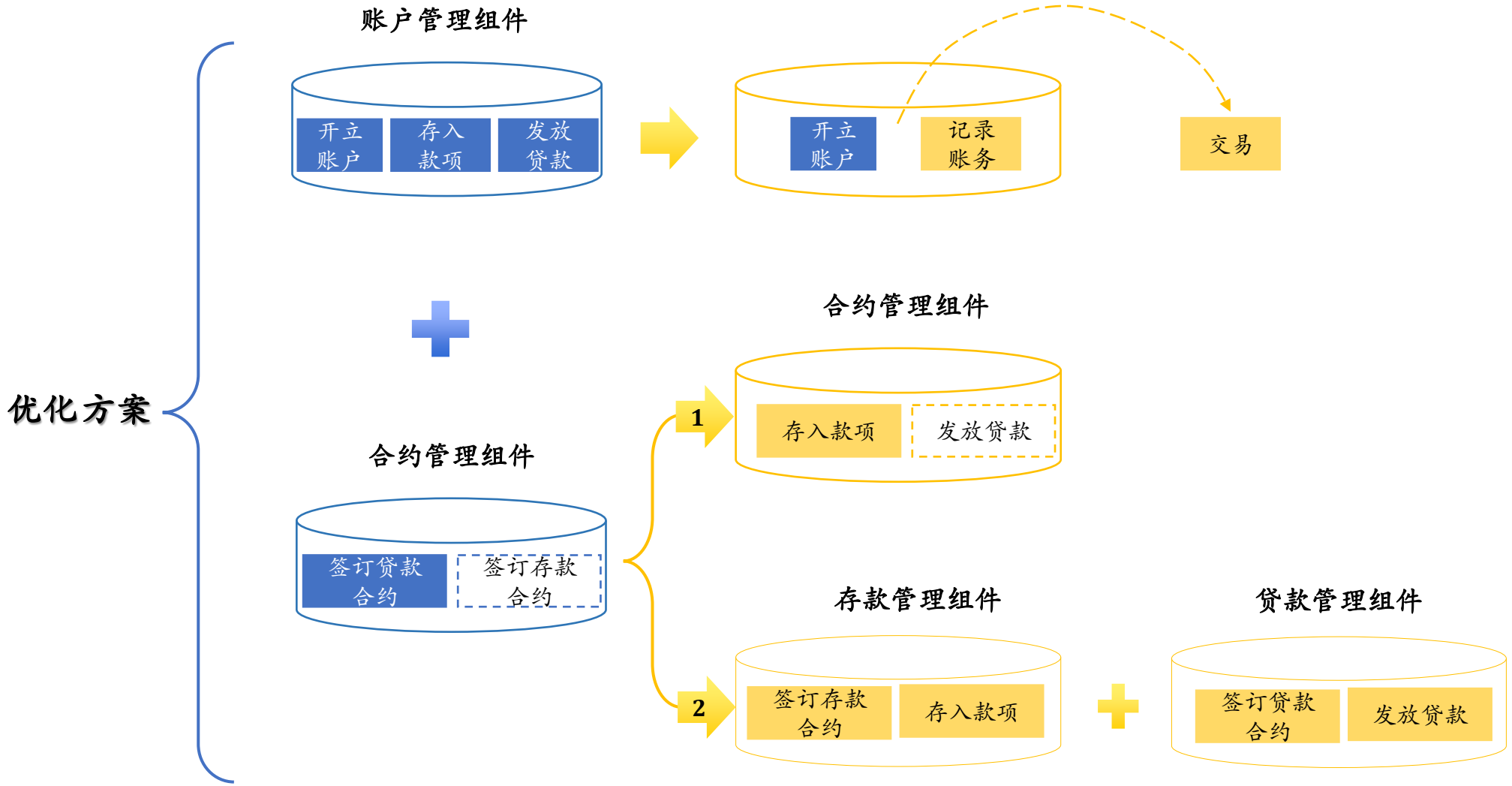
1.组件划分优化

组件划分示意图



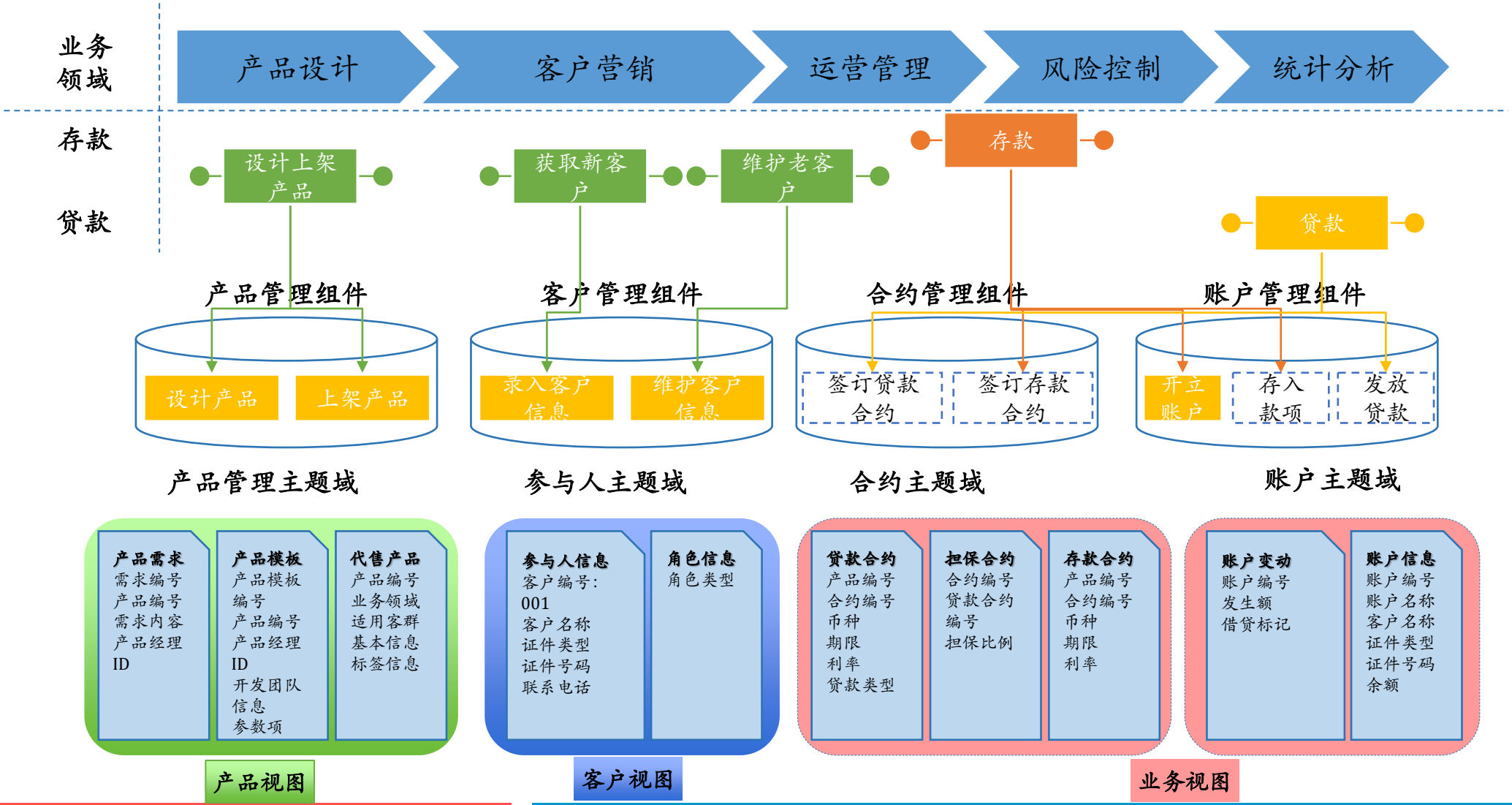
7.5 组件设计

1.组件划分优化



7.5 组件设计

2.组件第二次优化后的逻辑示意图



7.5 组件设计

企业级业务架构设计案例总结

- 企业唯一的价值链结构：产品设计、客户营销、运营管理、风险控制、统计分析。
- 依据价值链展开的2个垂直业务领域：存款、贷款。
- 在2个垂直业务领域中共用的3个业务活动：设计上架产品、获取新客户、维护老客户。
- 在2个垂直业务领域中共用的5个任务：设计产品、上架产品、录入客户信息、维护客户信息、开立账户。
- 基于数据主题域聚类任务后构成的面向各垂直业务领域开放自身能力的4个企业级业务组件：产品管理、客户管理、合约管理、账户管理。

业务架构更关注的是企业的整体设计，是典型的“不谋全局者不足以谋一隅”。如果再加上对战略的分析，就可以形成以战略为导向的整体企业能力规划

7.6 案例总结

1

- 业务架构设计并没有简单的衡量标准。

2

- 设计思路和关注点对架构方案有很直接的影响。

3

- 架构设计需要反复进行迭代，虽然这不是我们所情愿的，但是实际操作中这是不可避免的。

4

- 架构师的经验很重要，可以减少反复次数，尤其是关键设计的反复迭代。

架构设计是一个不断精炼和确认的过程，需要架构人员、技术人员在设计过程中努力“培养”客户，创造一个深度融合的机会。而且上述设计思路对于业务人员日常分析自己的工作环境、设计工作方案、改进工作流程都有帮助，是一种可以跨越IT边界的工作方法。对于这一过程的投入，业务与技术是双赢的。

第三部分：业务架构落地篇

第8章 从业务架构模型到业务架构方案

- 8.1 业务架构设计不是为了替代需求分析
- 8.2 制作业务架构方案
- 8.3 小团队的应对之道
- 8.4 需要充分解释架构方案
- 8.5 努力打造“通用语言”

第9章 基于业务架构方案的实施过程

- 9.1 基于业务架构的设计
- 9.2 基于业务架构的协调
- 9.3 处理架构调整的原则
- 9.4 企业级物有所值吗?

第10章 建立转型后的长期应用机制

- 10.1 项目结束了该怎么办?
- 10.2 促进深度融合的需求管理机制

第11章 这个“笨重”的过程与敏捷沾边吗?

- 11.1 传说中和现实中的双模开发
- 11.2 与正宗的敏捷对比
- 11.3 与非正宗的敏捷对比
- 11.4 且行且珍惜

第12章 企业级的“五难”

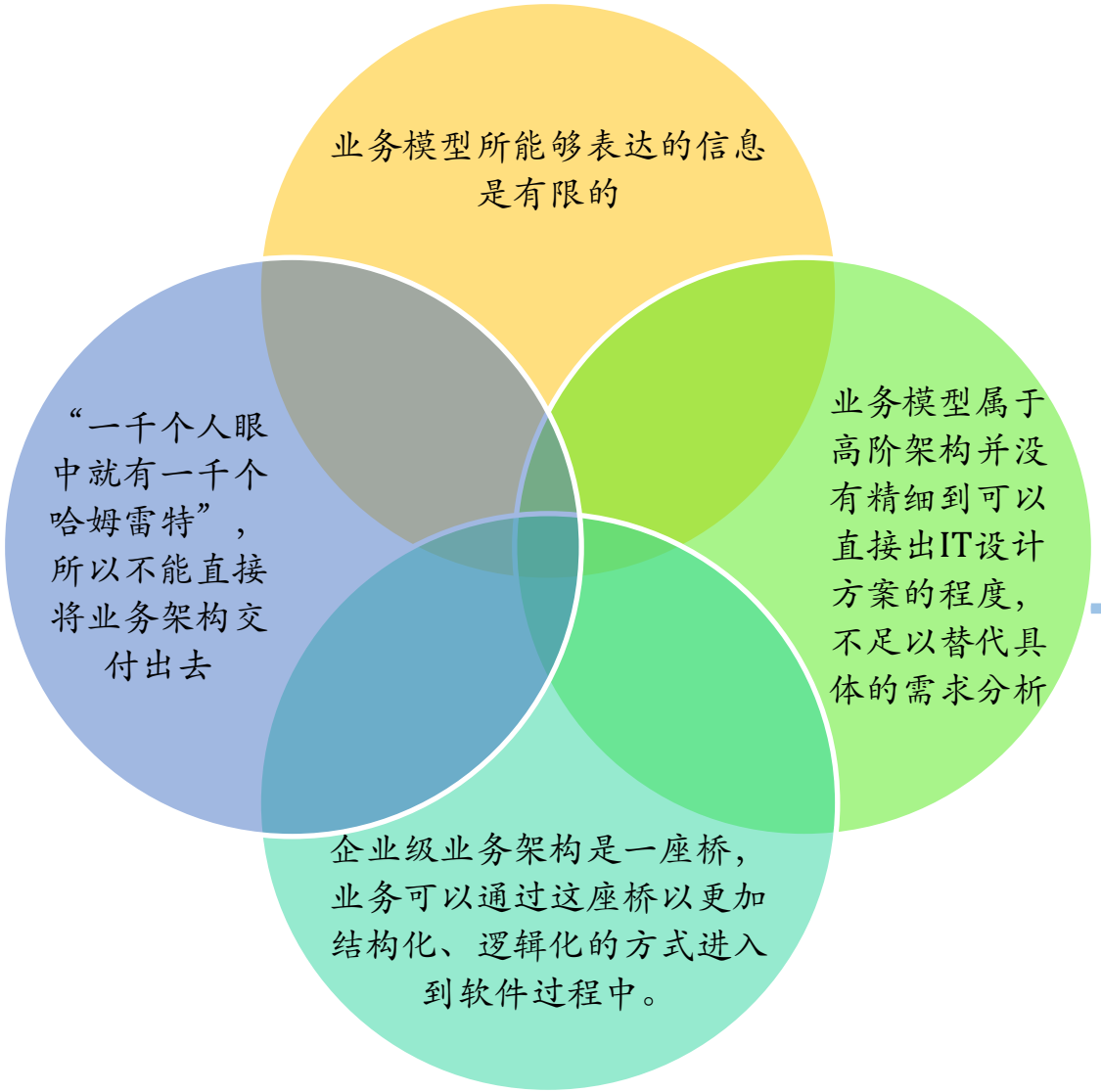
- 12.1 捷径难寻
- 12.2 文化难建
- 12.3 预期难控
- 12.4 权责难定
- 12.5 长志难立

第13章 实战：实现了快速设计的案例

- 13.1 项目背景及需求
- 13.2 设计思路和业务架构方案
- 13.3 案例总结

第8章 从业务架构模型到业务架构方案

8.1 业务架构设计不是为了代替需求分析



将模型转换为方案需要交付三个部分的文档

- 企业级架构方案的整体描述
- 分领域或分应用的方案描述
- 业务组件的方案描述

8.2 制作业务架构方案

将模型转换为方案需要交付三个部分的文档

企业级架构方案的整体描述

- 企业愿景、使命、目标介绍
- 外部干系人介绍
- 企业战略介绍
- 企业战略能力介绍
- 企业价值链介绍
- 全部业务领域的整体介绍
- 全部业务组件的整体介绍
- 全部数据主题域的整体介绍
- 企业整体组织结构的介绍

企业级业务架构方案的分领域描述

- 业务领域的目标、范围
- 业务领域的外部干系人介绍
- 业务领域的全部业务活动介绍
- 对部分任务的“降范”描述
- 业务组件协作关系介绍
- 业务领域完整的实体关系图
- 组织与角色描述

业务组件描述

- 业务组件的目标、范围
- 数据主题域的完整介绍
- 组件所包含任务的完整介绍

8.3 需要充分解释架构方案

建模过程并不能代替对架构方案的解释

- 建模是“少数人”的事，而参与项目的人数通常会较多
- 模型宣讲的传导可能会出现“信号衰减”

架构设计人员的缺位可能会导致方案失效

- 业务架构人员最好能够实时的跟进需求方和实施团队之间的沟通，防止双方协商而导致架构的崩坏
- 帮助业务和技术人员实现对模型的共同理解和充分沟通，遵守基于模型产生的项目边界和分工

8.4 努力打造“通用语言”

- 坚持选用具有足够能力的人员
- 坚持长期培养
- 坚持让其跟进项目而非只做架构设计

培养合格的业务架构师队伍

加强项目外的宣讲

- 模型化、结构化思维方式
- 每个条线、每个领域对企业架构都有所了解
- 用同一种建模方式，打造企业级项目的“通用语言”

第9章 基于业务架构方案的实施过程

9.1 基于业务架构的设计

1.细化业务架构模型

业务架构是将业务能力经过企业级规划后传导给IT端，并不会限制IT的实现方式

业务架构模型

业务架构模型

- 为了更好地关注企业级设计，必然会对活动、任务进行适当的抽象
- 减少对业务细节的表述，这样的模型可以让项目实施团队了解其标准化的核心以及功能复用方面的设计思路
- 建好的模型中却不能表达太多的细节，这些细节会让抽象表达变得混乱

IT设计

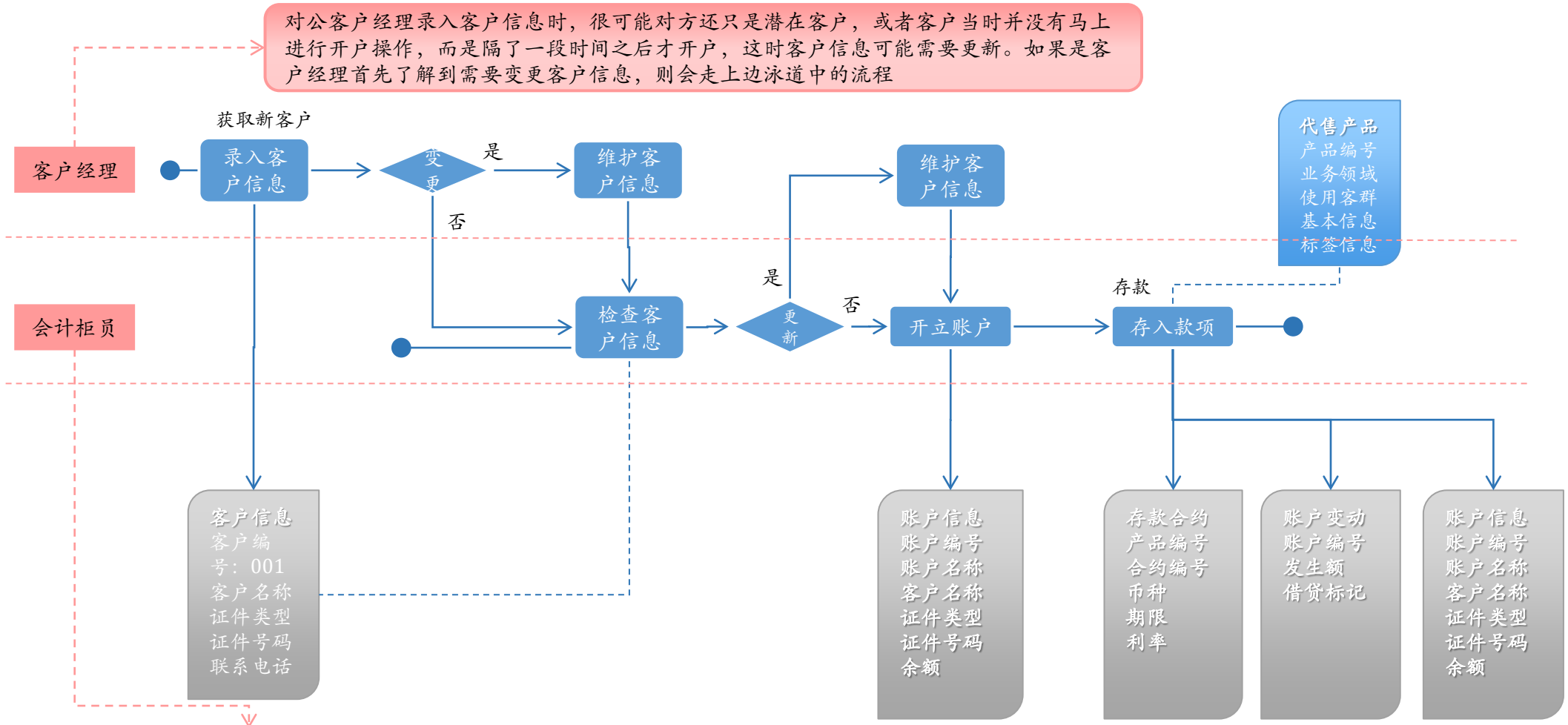
- 设计阶段就必须对模型表达的工作流再进行细化，
- 实施团队需要了解到细节需求
- 设计阶段的细化其实就是基于业务架构方案的需求分析过程
- 过程允许对原有的工作流进行拆分或者重构。但是有一个前提，那就是新产生的元素必须基于旧元素，最好能够显性地标明继承关系，这样才能保证设计的连续性。

IT设计

IT需要适配业务架构的规划，基于企业级业务架构的实施过程，其实是IT细化企业级业务架构设计，并基于企业级业务架构进行实施协调与管控的软件过程

9.1 基于业务架构的设计

1.细化业务架构模型



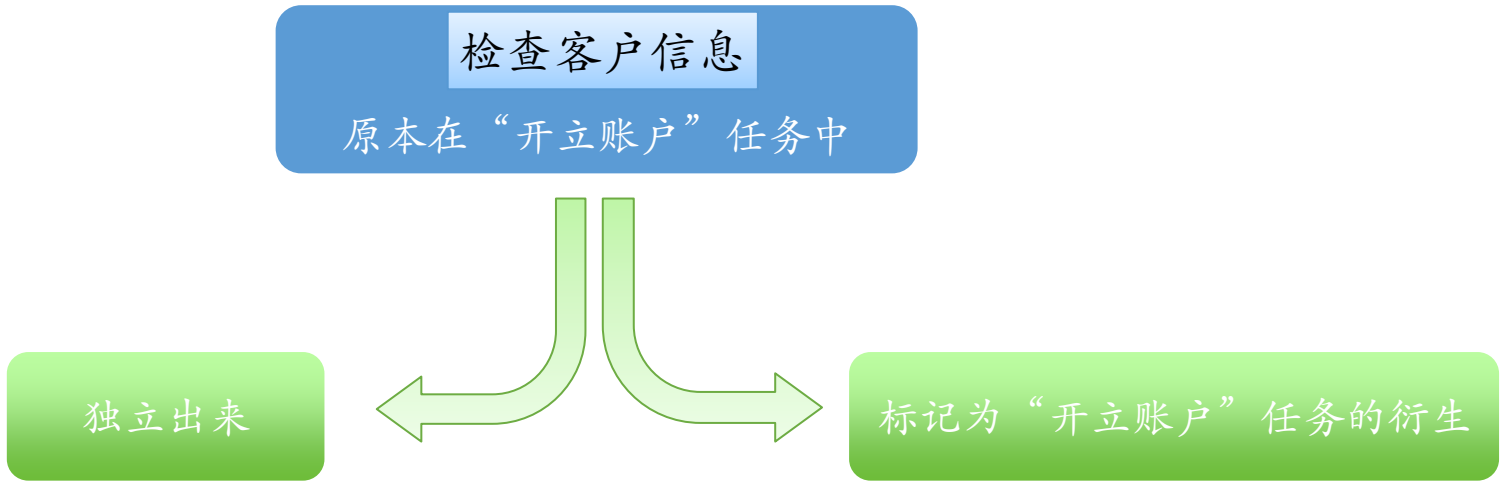
串接后的“长流程”业务架构模型示意图

“获取新客户”“维护老客户”“存款”这3个活动，在实际设计中，如果存款部分和客户管理部分按照组件边界划分了项目组，那么存款项目组负责合约管理组件和账户管理组件，客户项目组负责客户管理组件。在这种情况下，需求分析或者开发人员很可能更愿意用一个较长的流程来表示应用视角，这样可以更好地描述两个项目组提供的功能在具体的存款业务场景中是如何衔接的

9.1 基于业务架构的设计

2.处理“新发现”

原本没有“检查客户信息”这个任务，在最初的建模过程中，该任务是可以合并在开立账户这个任务中的，但是在细化流程阶段，则可以考虑是不是应该将其独立出来。是否独立取决于整体的架构判断，如果其他流程中也涉及了这种拆分，则确实可以调整企业级业务架构模型；如果不涉及或者很少有领域涉及，则可以将其标记为“开立账户”任务的衍生。这种细化可能会产生新的数据需求，设计新的数据实体，或者在原有的数据实体下产生根据某种维度细分的子实体。上述过程已经代表了对原有业务架构的一次迭代。

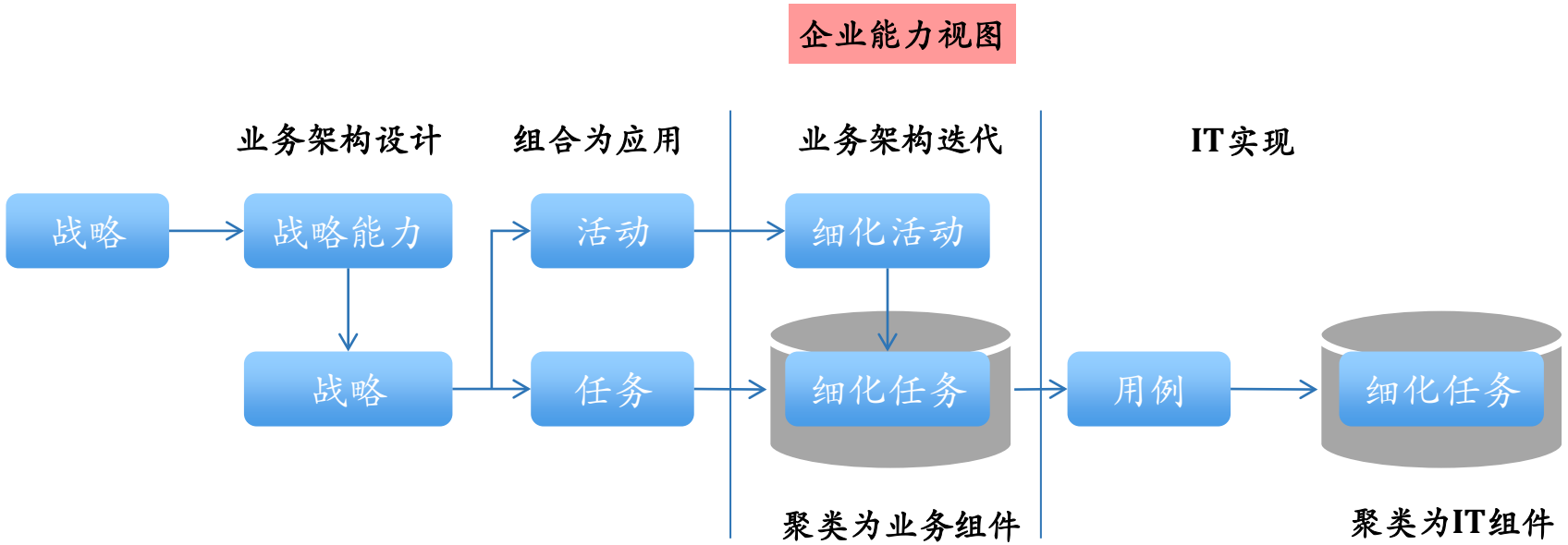


9.1 基于业务架构的设计

3.进行惯常的IT设计，但要建立一体化视图

基于业务架构的企业级项目很重要的一点是要建立各阶段工作成果之间明确的衔接关系，尤其是设计元素之间的联系，应建立“**战略一战略能力一需求一活动与任务一细化活动与任务-用例-交易或服务**”这样明确的关联关系，形成一个完整的、分层级的企业能力视图。

企业能力视图不能仅局限于业务侧或者仅到组件层级，而是应该延伸到最终的实现，业务与技术的深度融合是今后企业发展的大趋势。业务就是技术，技术也是业务，企业未来的作战地图必须是基于业务技术一体化的视图，而非分裂的视角。

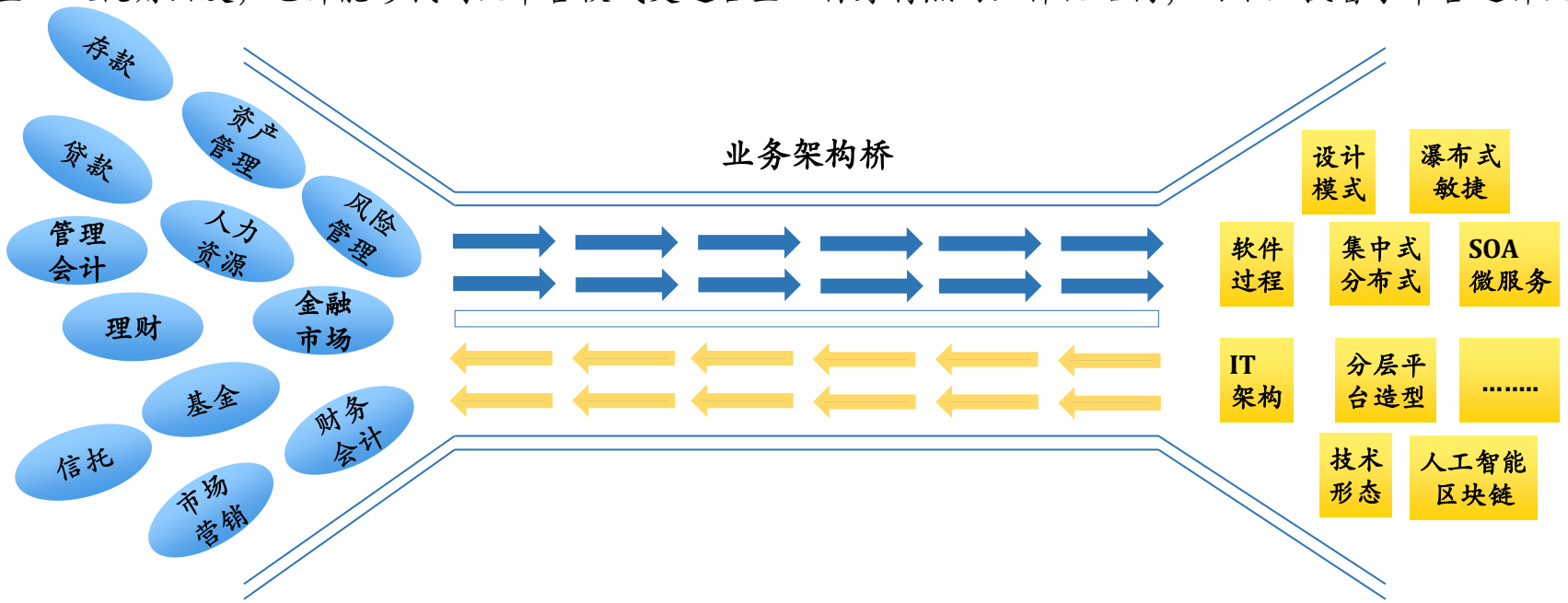


企业能力视图的元素积极关系示意图

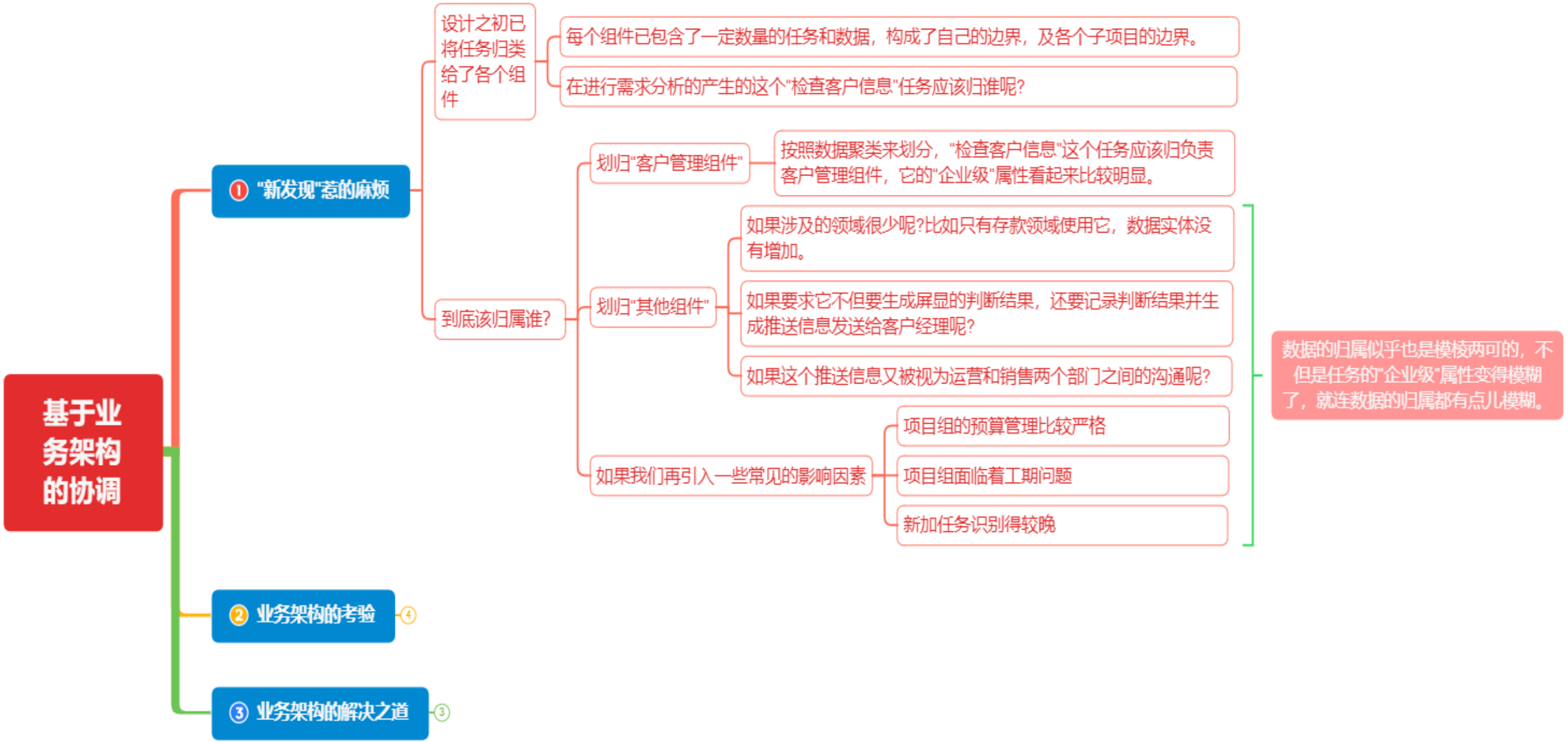
9.1 基于业务架构的设计

4.业务架构桥梁的作用

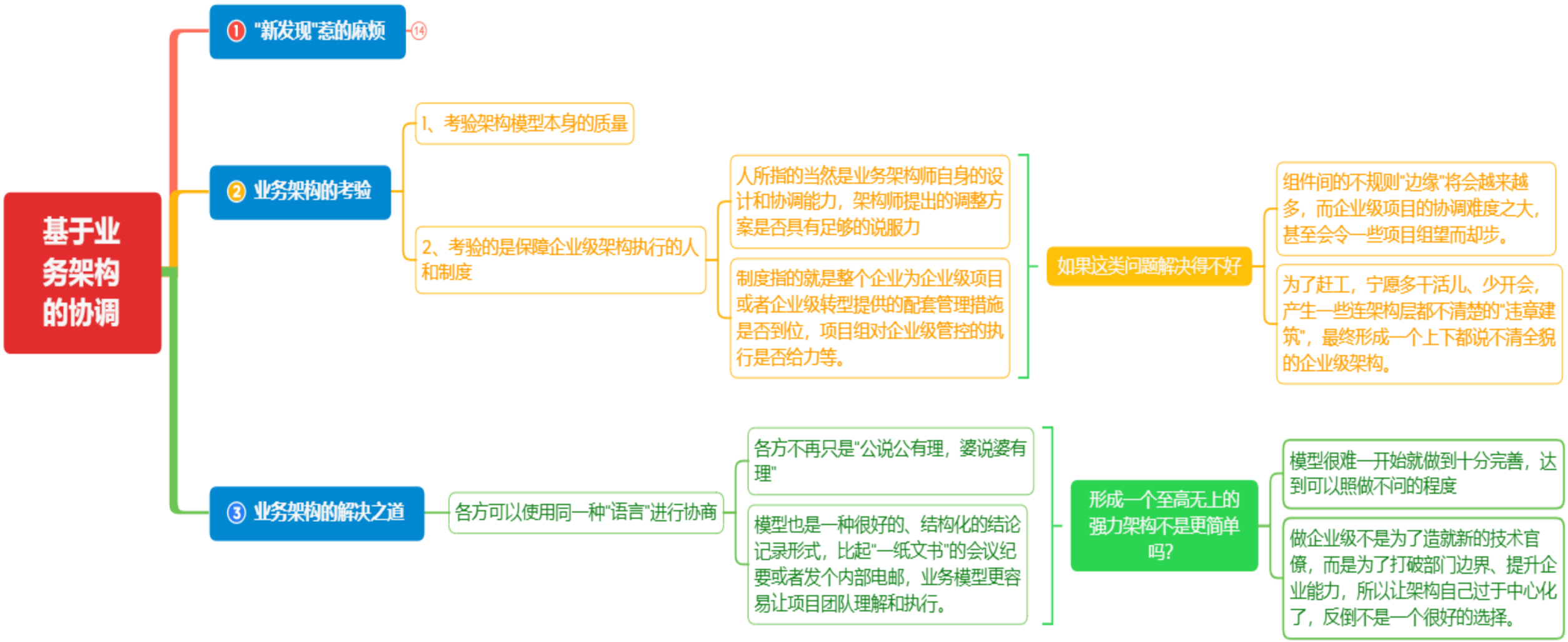
- 业务架构的桥梁作用强调一体化，指的是二者在设计元素层面的联系，但是业务架构与IT实现方式之间是具有一定自由度的。因此，企业级业务架构设计的落地并非要有一种特殊的开发过程与之配套，其后对接的IT设计与分析方式可以是灵活多样的。
- 业务架构是IT过程的一种输入，其要约束的是IT的设计理念和实现目标，而非具体的设计过程与设计方法，后者可以更多地取决于企业自身的开发习惯与特点。
- 纷乱复杂的业务“理想”经过“业务架构桥”的梳理，将展现出其企业级规划的“秩序”，从而使IT设计更接近于业务的核心诉求和内在逻辑，而不受业务表象牵引。至于实现“秩序”的具体设计，则是“不拘一格”的。
- 中台其实也是一个持续规划和迭代的结果。如果抛开规模导致的技术复杂度来看，只要坚持企业级方向或者合理的领域级分治与协作，经过设计和沉降过程，产生中台业务规划只是一个必然的结果。每个企业都能找到符合自己特点的业务中台，而这个寻找的过程对企业而言将胜过中台这个结果本身。从企业级视角出发，也许能够找到比中台模式更适合企业自身特点的组件化结构，而不必执着于中台这种比喻形式。



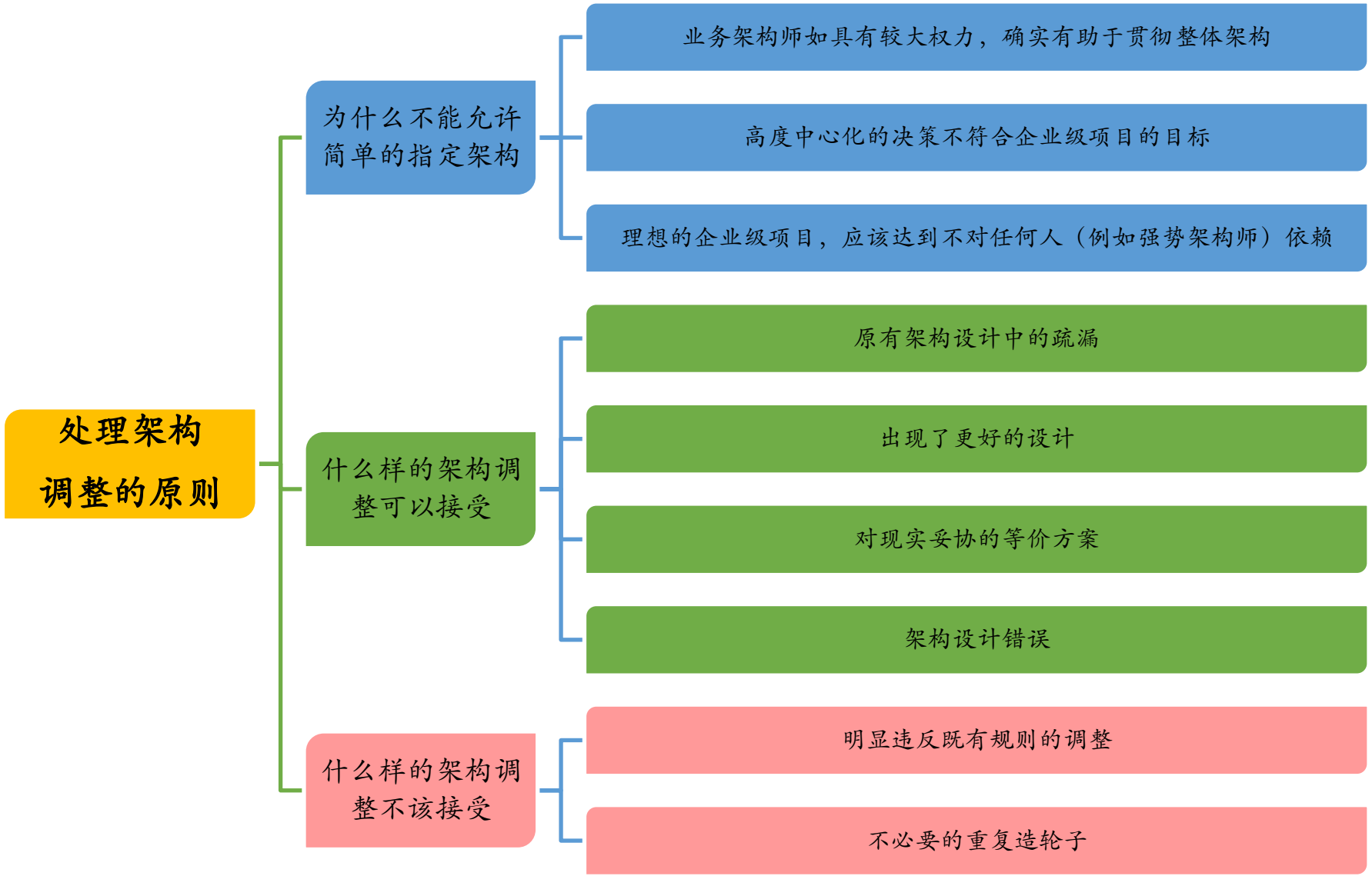
9.2 基于业务架构的协调



9.2 基于业务架构的协调



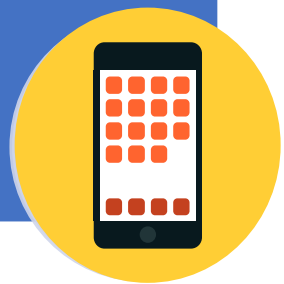
9.3 处理架构调整的原则



9.4 企业级物有所值吗？

- 系统互联互通、数据共享、灵活响应、快速上线、DevOps等开发提升的反应速度是有上限的，信息共享的价值可能更大
- 技术每隔5-8年换代，节省出来的成本难以计算
- 转型成本扣除
- 难以高度精细化管理

一本难算的账



- 企业级开发最重要的是转变企业文化，打破部门边界，让业务和技术融为一体，一体化带来的内部变化，清晰分工和高效协作才是最有价值的，是未来长期竞争力的关键
- 企业级项目成功的标志是文化、思维的转变；做企业级难点不在于技术，真正解决的是业务问题、组织问题、思想问题。
- 结果要由时间、感受和市场来检验，而非标准来检验

无形的核心价值

第10章 建立转型后的长期应用机制

10.1 项目结束了该怎么办？

1、别做“一锤子买卖”

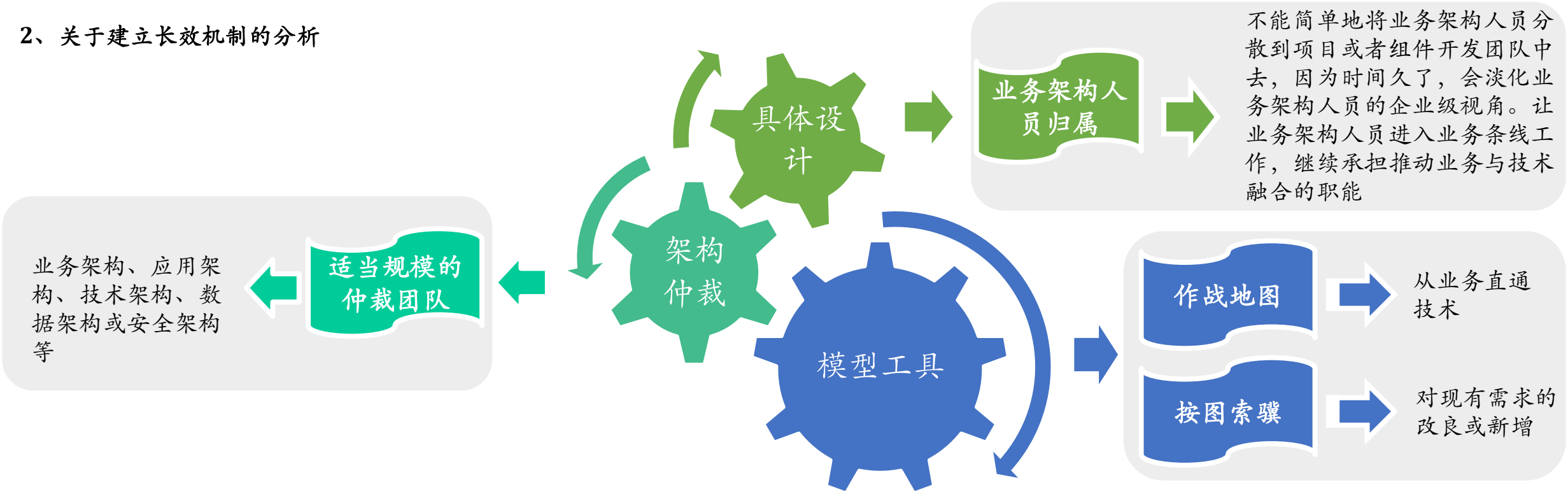
① 企业级转型，或者称为中台，都不是“一锤子买卖”，并不是项目上线后就万事大吉了。按照“熵增”理论，没有良好的维护，再好的架构也会慢慢崩坏，更何况架构自身也必须与时俱进。任何领先都是暂时的，尤其是在技术方面。

② 企业级业务架构建立之后，必须坚持使用这个架构去管理新的需求，随着业务和技术的发展，需要不断调整架构，以保持架构常用常新。

③ 做企业级转型时，为了保证项目的顺利进行，企业可以选择按照项目管理的套路构建临时管理机构，形成企业级架构管理。项目开展期间，临时机制，临时机构可以跨部门的项目管理组织做后盾，可以很好地执行下去的。

④ 但项目结束之后跨部门的项目管理组织，即便形式上可以固化，也很难长期维持其热情和效率。

2、关于建立长效机制的分析



10.2 促进深度融合的需求管理机制

1、什么是深度融合?

人的融合

让业务人员和科技人员能够坐到一起充分交流、沟通，多了解对方的想法，互相碰撞和渗透思想。

技术人员派驻业务部门

技术人员必须向前迈出一大步，参与到业务中去，一定数量的业务架构设计人员分散派驻到业务部门，但是人员管理不归属于业务部门

深度交融和渗透

技术人员与业务人员广泛交流，不断提升业务人员对企业级理念、技术实现、技术趋势的理解，激发业务人员更大的想象空间和跨部门协作的动力，减轻过去业务人员“冥思苦想”新需求的痛苦，让双方在工作交融在一起。

数字化思维转型终极目标

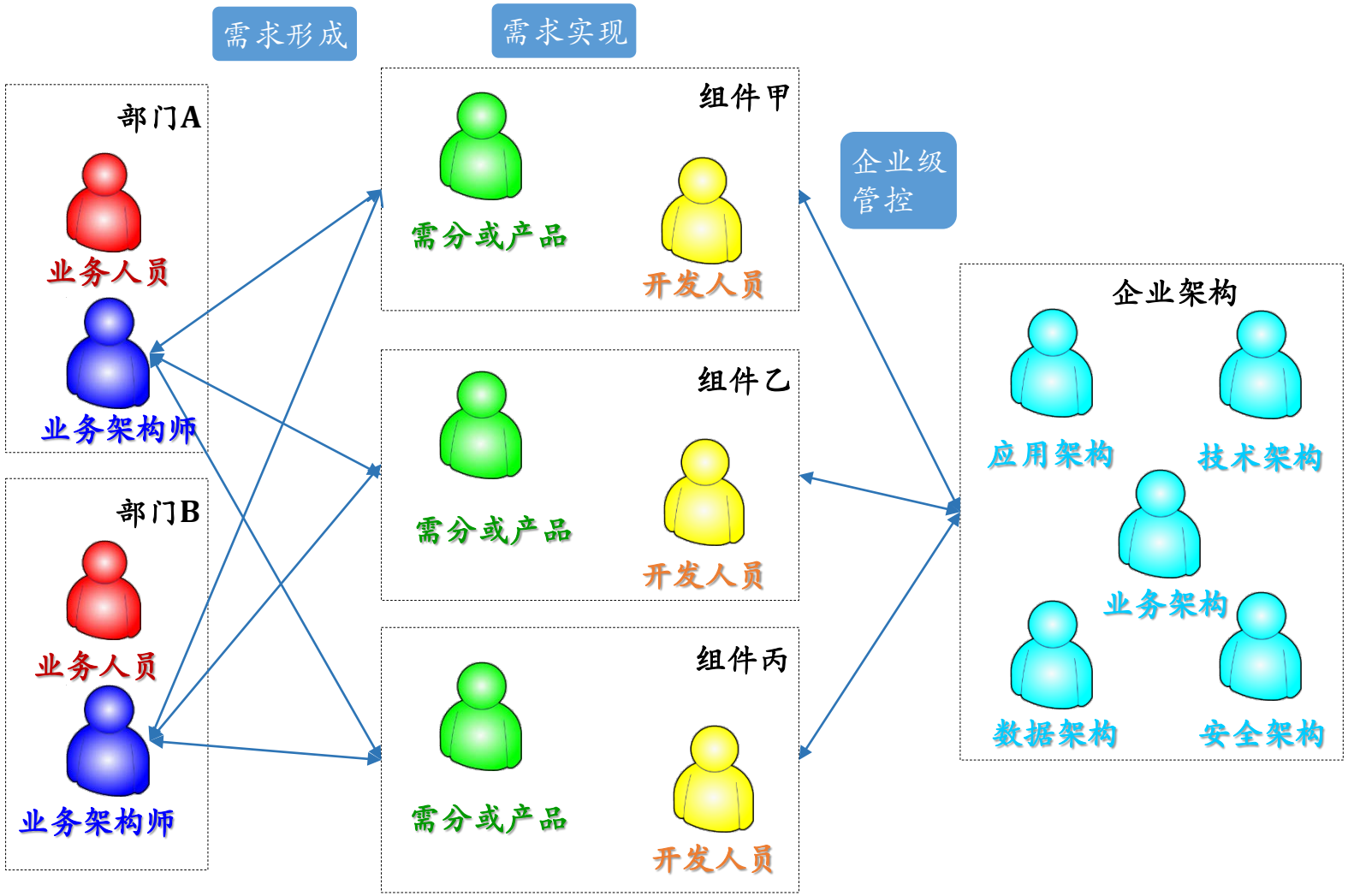
通过技术人员的增加带动更深入的交流，从而帮助业务人员实现数字化思维的转型，实现业务与技术深度融合的目标。

聘请若干技术大牛或者仰仗领头科技公司开展若干看起来“高端大气上档次”的项目

使用了一大堆“不知其所以然”的工具，变成为“数字领袖”

10.2 促进深度融合的需求管理机制

2、业务架构师的工作模式



业务架构师工作组织形式示意图

业务架构师非常适合作为与业务人员接触的“技术第一人”，在工作中，业务架构师可以及时调动需求分析人员、产品经理、开发人员提早参加到需求的形成过程中，将需求管理直接转变为业务规划，这才是各方都希望实现的融合与快速反应。

10.2 促进深度融合的需求管理机制

2、业务架构师的工作模式

① 业务架构师分散驻扎在各个业务部门

② 需求产生时即采用模型工具与业务人员共同讨论

③ 可以根据需要，要求组件开发团队的需求分析人员、产品经理或者开发人员提前介入分析

④ 需求成型后即进入实施过程，业务架构师可以代表业务人员进入项目组

⑤ 与开发团队共同工作一段时间；项目组若出现业务架构师不能处理好的协调问题时，则会提交给企业架构进行沟通

⑥ 必要的时候还会进行仲裁。

保证机制

- 一是要保证业务架构师的数量，不能每个部门一个，那样做是做不过来也做不到位的，而是要有多个，要有“替补”
- 二是要保证业务架构师能够每隔两年左右进行部门间的轮岗一次，业务人员不能轻易调换部门，但是业务架构师则要保持一定的流动性，以促进企业级理念的生长，架构师要有替补也是为了加强轮换能力。

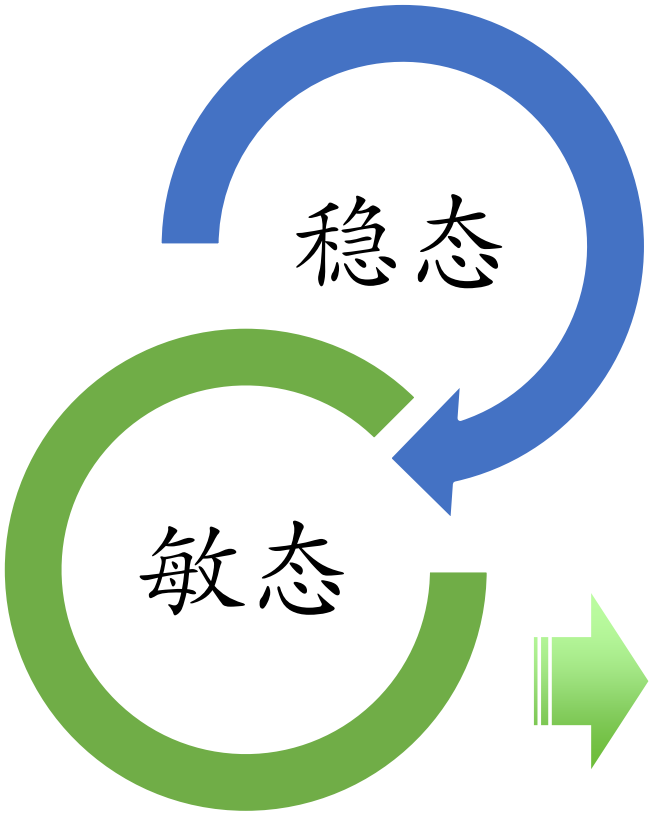
10.2 促进深度融合的需求管理机制

3、千里之堤毁于蚁穴

- 打江山容易，坐江山难
  - 做企业级项目过程痛苦，但仍能坚持过去的。然而企业级理念的长期保持和应用是更为困难的事情
  - 需求管理机制对非科技类企业而言，很难成为企业的工作重心。
- 没有“万年长青”
  - 中台模式也需要进行长期维护。只靠技术人员无法处理好，要考虑什么样的机制去管理需求、维护中台架构
  - 实现了中台就能达到快速响应、达到业务与技术深度融合了吗
- 技术人员仍然只是“少数派”
  - 相比互联网企业，传统企业技术人员仍然只是“少数派”
  - 技术人员抱怨业务与技术之间的互不理解
- 将业务整体拉入到开发机制当中
  - 从企业级业务架构入手、从顶层设计入手
  - 光靠技术人员无法持久规范中台

第11章 这个笨重的过程与敏捷沾边吗？

11.1 传说中和现实中的双模开发



稳态 → 按部就班 → 可预见性的业务使用传统瀑布式开发

敏态 → 大干快上 → 探索性业务使用敏捷开发

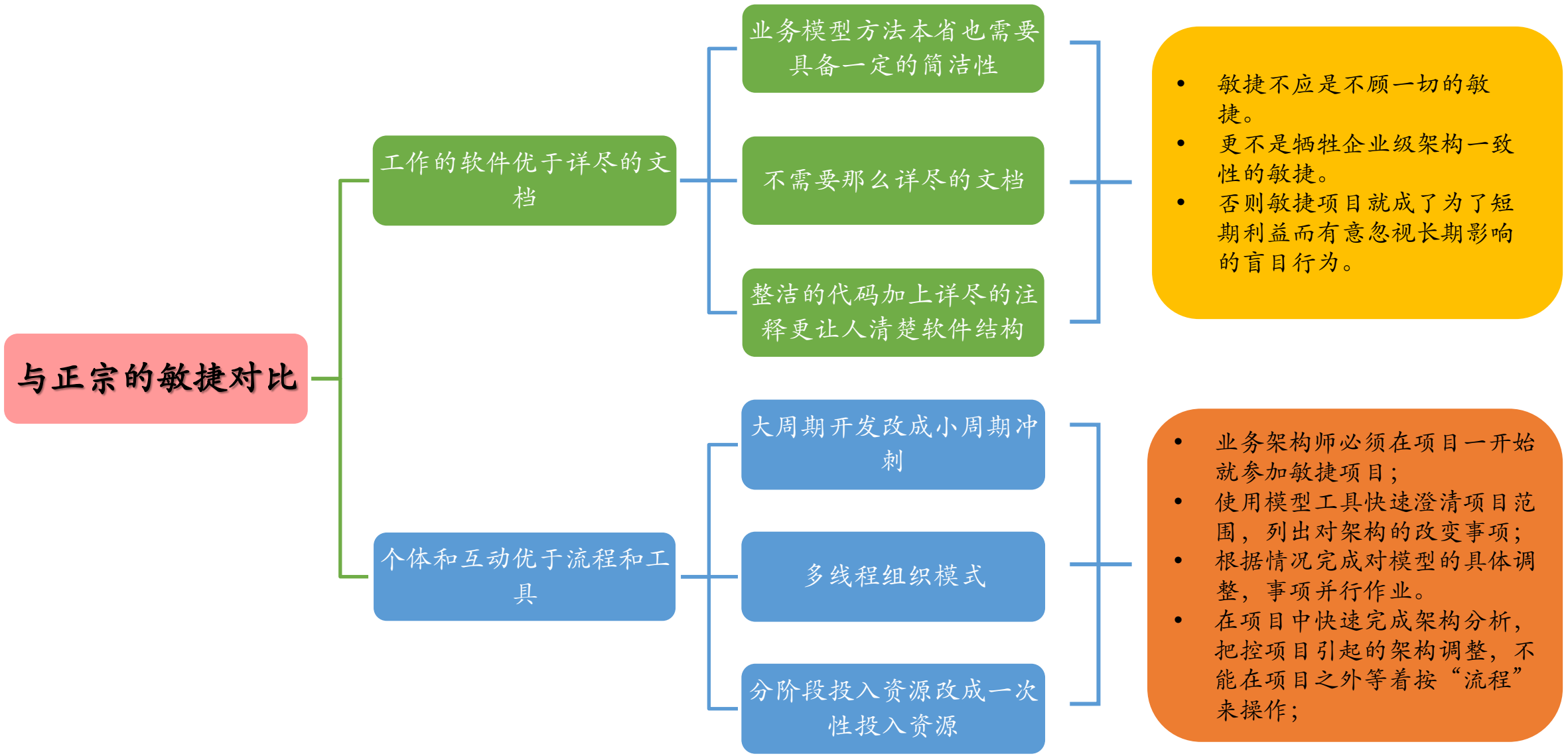
- **正宗敏捷**：符合敏捷开发定义
- **非正宗敏捷**：特事特办

经过业务机构设计、应用业务模型驱动的开发是不是就显得有些“笨重”了呢?

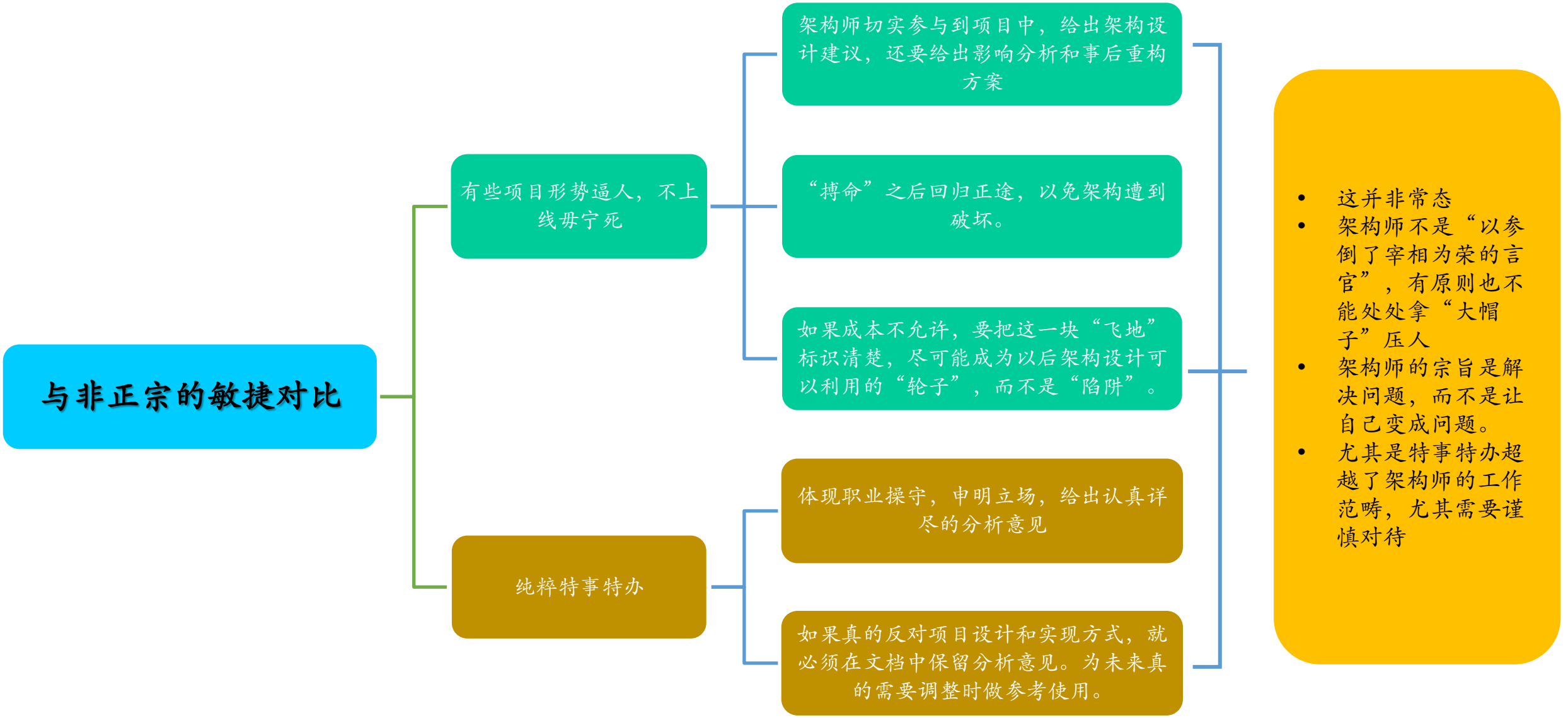
在敏捷过程中是否应该省去业务架构设计环节呢?

?

11.2 与正宗的敏捷对比



11.3 与非正宗的敏捷对比



11.4 且行且珍惜

核心是企业文化

管理大企业既要有担当主力、稳扎稳打的方阵不对，也要有处理特殊任务的游骑兵

企业的双模开发，让企业在“按部就班”和“特事特办”中切换也很正常

并非是一个单纯的开发模式或是技术机构的问题

更多的是企业文化，人和人所在的环境

第12章 企业级的“五难”

12.1 捷径难寻

自顶向下实现企业级很难

- DDD对企业级并不抱太大希望，其认为企业级的建设路径只能是一个领域一个领域地不断尝试和融合。
- DDD不认为企业级真的可以通过自顶向下的规划来产生，而只能是自底向上的生长。
- 科技公司中企业级的代表，可能莫过于阿里集团的中台模式，但这个模式也是逐渐演化出来的

金融是一个很不“专业”的专业

- 包含的内容五花八门，看似都在同一个领域，实际上却是“各不相干”。
- 各类业务的共性无非是客户都是同一群客户，这些业务围绕客户共建了一个账户体系，虽然业务之间的差别很大，但是大多数都得单独记账。
- 如果自顶向下地看，客户和账务应该是企业级的，而其他部分，严谨地说，就像DDD主张的那样，得一个领域一个领域地去研究，这也是建模和标准化的难点。

复制别人成功经验很难

- 企业级建设的难度与企业所在行业的特点有直接关系，没有一个通用的企业级业务模型可以随便套用。银行不能直接套用阿里集团的业务架构，阿里集团也不能直接套用银行的业务架构。甚至在同一个行业内，企业与企业之间内部特点的差别，也会决定企业级建设路径和结果的不同。
- 没有可以简单复制的模式帮你快速切换到企业级。别人的经验，无论成败，对你而言都只能是个借鉴，自己的路只能自己摸索前进。

12.2 文化难建

业务上需要做的调整不亚于技术上要做的调整，而对于企业文化的调整则尤为重要

彼得·德鲁克曾说过这样一句名言：“文化能将战略当午餐吃掉”，这的确是一个难题。

引人新方法、新思维产生的冲击也需要大量的时间去消化，这将是一个彻头彻尾的大转身

建立企业级不于一场“内战”，一场对部门边界、协同关系的重新界定。

12.3 预期难建

期望越高、失望越大

- 企业级项目，各方往往会对此寄予厚望，将蓝图描绘得太过美好。但是建设周期的漫长、建设过程的曲折，以及中间不断对现实做出的妥协，会让很多美好的“理想”大打折扣
- 或者由于项目的进度原因而一拖再拖，这也会让实现过程和最终结果都看起来并没有当初设想的那么美好。

成本和收益都是难以直接计算的

- 有点儿像从单体应用到SOA、微服务的演变，看起来好像因为零件化了，灵活性得到了提升，但通信和维护却变复杂了，企业级效果的积极方面可能也要随着时间的推移才能逐渐显现。
- 很多变化并不是那么集中发生的，其中一些变化可能在转型的过程中就已经出现了，只是没有被注意到，比如协同能力的慢慢提升。

不要为企业级项目戴上太多不该戴的“高帽”

- 需要事先管理好企业的预期，真正该戴的“高帽”——完成一次企业文化的建设，实现整体转型。
- “转型”转的就是行为习惯，转型项目成功与否，只需要观察企业中各级领导、员工的行为习惯是较之以前是否发生了变化。如果真的发生了改变，那至少可以说明“转型”是有作用的；
- 转型的目标也许无法一次性达成，但是如果员工的行为习惯已经朝向预期发展，那么系统目标的实现也许只是需要更多的时间而已。

12.4 权责难定

阿里智能云事业群总裁张建锋（花名“行颠”）先生在接受外部采访时表示，“大部分企业都缺少非常好的业务架构，这个业务架构你没法招聘，只能自己培养”

企业级转型期间

架构可以有较大的权力去保证项目落地

转型期结束之后

转入常态开发时，架构又该如何定位？毕竟架构就是架构，不是企业的管理者。

企业中本就有强烈的“官本位”思想。

架构的定位困难

架构定位的困难在于：若权力太小，则不足以维护企业级，企业级甚至会随着时间的流逝而“名存实亡”；

若权力过大，则又会发展成新的部门化组织，一旦开始以架构“卫道士”自居，就会导致对架构创新的阻碍。

必须关注架构师的重要性

他们是数字化转型的关键力量，而好的架构师能够帮助企业实现合理的整体规划。

他们虽不是管理者，却拥有基于项目实践积累的管理经验

他们并非仅仅是技术专家，而是实现企业整体战略不可或缺的力量。

12.5 长志难立

减肥、戒烟的失败

- 爱美之心，人皆有之，人们都知道体型好代表既漂亮又健康，因此花钱、花时间减肥的大有人在，但是真正坚持到底、后期不反弹的又有多少？

企业级的放弃和崩坏

- 未必是将架构组织撤销、机制停掉这类激烈的动作
- 而是各种“畏难情绪”“客观原因”导致的缓慢的无序，是由一个个需求的分配、落地的偏离堆积而来的

最终还是战略和文化的问题

- 企业级维护工作很难与个体、局部的利益有明确的结合，很多时候需要依靠员工自觉去维护企业级
- 除了所谓的制度、机制之外，还会绕回到战略和文化的问题上，即人们常说的如何打造一个“伟大的企业”。
- 对于大型传统企业，自始至终，非技术因素的作用与技术因素相比，至少是等量齐观的

第13章 实战：实现了快速设计的案例

13.1 项目背景及需求

项目背景及需求

本案例中的需求是某企业与某互联网公司合作进行的实物贵金属产品在线销售的业务。当时，其竞争对手与另一家互联网公司刚刚合作了此类项目，受市场形势所迫，该项目必须快马加鞭，紧急施工，特事特办。

- 需求产生时，该企业正在企业级转型期间，但**已经完成了企业整体战略设计、高阶能力需求分解，实现了企业级的业务架构设计和业务模型建设**，并已应用模型驱动企业级开发了数年时间，工具使用已经比较熟练。
- 在转型项目期间，该企业平均每年都有几十个大型项目同时开展。**所有业务需求都要经过业务架构分析，出具业务架构方案**，再落实到具体项目组，可以说，企业级分析是项目的先导和开发任务分工的依据。
- 业务部门为配合快速开发，已经**连夜写出了业务需求。需求共计15页，将近9000字，涉及11个大的需求项**，需要业务架构师马上根据需求文档形成业务架构方案，以推动快速立项、快速进入开发阶段。
- 需求的主要内容包括：为客户建立虚拟账户，用于记录客户买卖交易、持仓等；支持使用该互联网公司的黄金发送红包（黄金份额）；红包的账务性处理、红包资金的支付结算及划转；支持黄金实物兑换；支持黄金转赠；销售数据提取、报表统计、实物提取配送数据交互以及相应的核算规则等。

13.2 项目思路和业务架构方案

业务架构方案

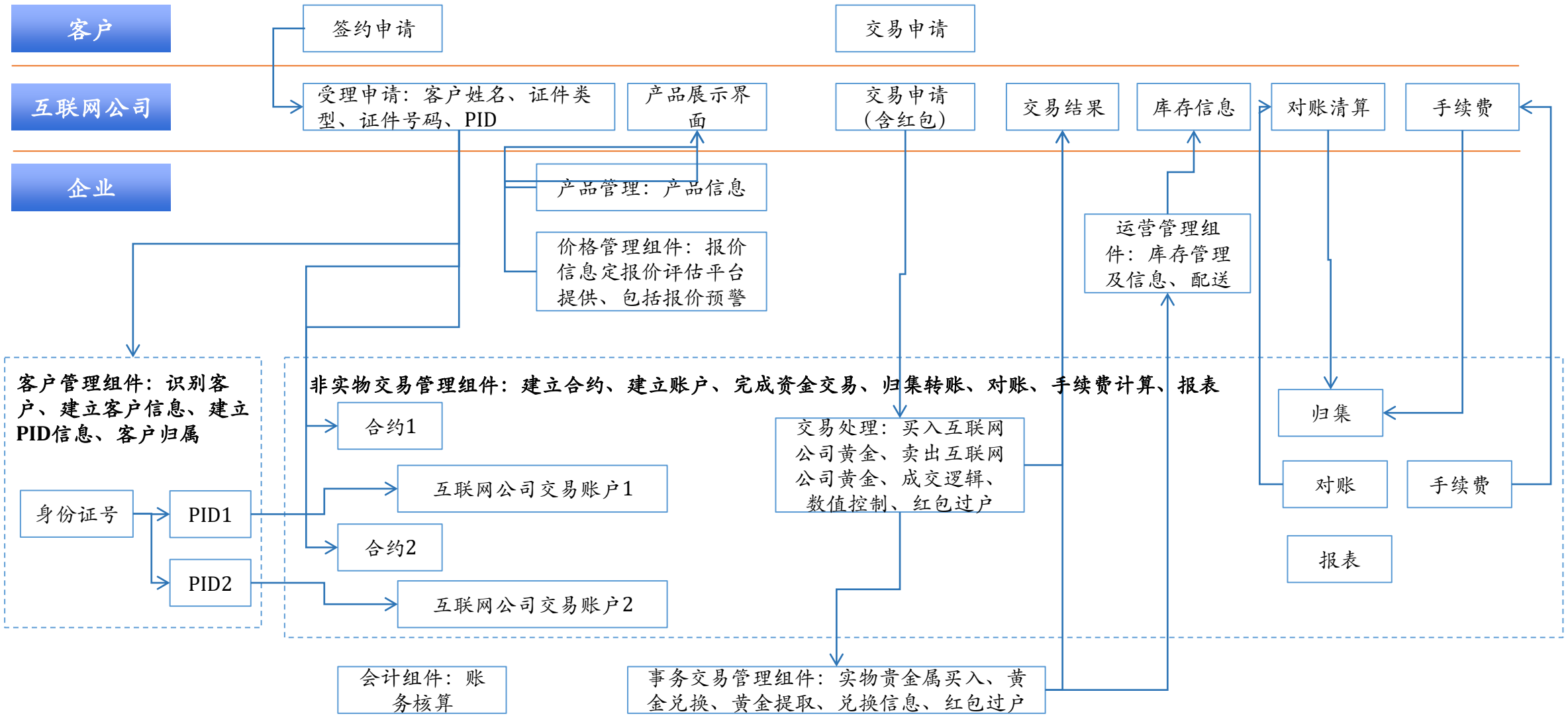
- 识别新业务流程与原有业务流程的差异
- 判断所涉及的模型范围，识别出需要使用的活动、任务、涉及的组件
- 识别需要新增的内容，新增的内容应该归属于哪个任务和组件等。
- 为项目组指派具体承接的需求，这不是简单地对号入座，架构设计最重要的就是理清结构和关系，说清各部分的具体分工和协作，给出明确的设计理由

业务需求分析

- 包含了大量的业务组件和领域级业务应用
- 涉及多个核心组件及公共类支持组件
- 涉及7个现有业务组件，分别是：**“签约”“产品查询”“交易（含红包）”“对账”“结算费用”**这几个主要的工作流程，

13.2 项目思路和业务架构方案

互联网黄金在线销售业务架构关系图



13.2 项目思路和业务架构方案

该过程的处理，由企业新增的渠道负责信息传输，“客户管理组件”负责识别是否应为企业新增客户，并记录客户在互联网公司的PID信息，与企业的客户信息进行关联；xx交易组件则负责为每个PID建立单独的交易合约和账户。考虑到未来可能与该互联网公司有更多的产品合作，因此，PID应该作为企业级信息进行管理，以方便各组件的查询并保持唯一性。

签约

- 客户向互联网公司提出建立购买企业的实物贵金属产品的合约申请，互联网公司将相关信息及客户在互联网公司的PID（客户在互联网公司的身份标识）传到企业。由于一个客户可能存在多个PID，因此需要分别为其建立账户。

产品查询

- 产品信息“产品管理组件”负责提供产品查询；报价涉及7×24小时实时人民币黄金报价，并允许在账户金报价、金交所报价、交易对手报价等各个报价来源之间自由切换，允许买入价、卖出价、中间价等多种形式报出，因此，需要由“价格管理组件”提供报价查询。

交易

- 交易涉及贵金属买卖和对红包转账的支持，由“非实物交易管理组件”负责账户金交易、资金归集转账，“实物交易管理组件”负责实物金交易，“营运管理组件”负责库存和配送管理，核算则由“会计组件”完成，这些业务都有现有业务活动和任务可以支持。

对账清算和手续费扣收

- 这两项都由“非实物交易管理组件”基于交易记录发起，清算结果和扣收都是通过互联网公司在企业的资金账户和企业内部账户之间转账完成，也有基础的业务活动可以支持。

报表

- 由于报表都是基于该组件的交易记录而生成的，因此可以由该组件负责提供支持。

13.3 案例总结

对原有业务架构和模型的充分复用

方案最终只是在原有的业务模型中增加了部分描述和规则，就一个抽象业务模型而言，不需要再增加活动、任务这些较大的元素了。

模型化的业务架构工具对企业级需求分析的加速作用

该方案涉及的客户信息管理、交易、核算、运营等，都是由不同业务条线负责管理的，是典型的企业级方案。由于是以业务模型作为架构分析工具，因此该架构能够快速识别需求的归属，并为跨条线的协调提供客观依据。

业务架构师必须非常熟悉项目情况

该方案能够快速形成也得益于架构设计人员对项目的深入了解。所以，方案的设计乃至后续项目组对分工的接受都很顺利。由于工作边界的原因，企业级业务架构师与项目之间可能会存在一定的“距离”，因此务必要时刻注意补充自身对项目的了解，以防止“距离”变成“疏离”。

业务架构还可以对提高项目的效率发挥更大作用

因为业务上已经准备了较为详尽的需求文档，业务架构设计工作少去了一些了解过程。但是，如果业务架构人员从需求梳理就开始介入，也可能会进一步提高需求分析的效率，推动一种基于业务模型的敏捷过程。

第四部分：业务架构落地篇

第14章 如何支持面向构件的设计

- 14.1 “乐高积木”式的软件设计
- 14.2 “颗粒度”问题
- 14.3 构件模型的设计方式
- 14.4 建立构件模型的虚拟案例
- 14.5 构件模型的技术设计建议
- 14.6 本章小结

第15章 构建轻量级架构管理工具

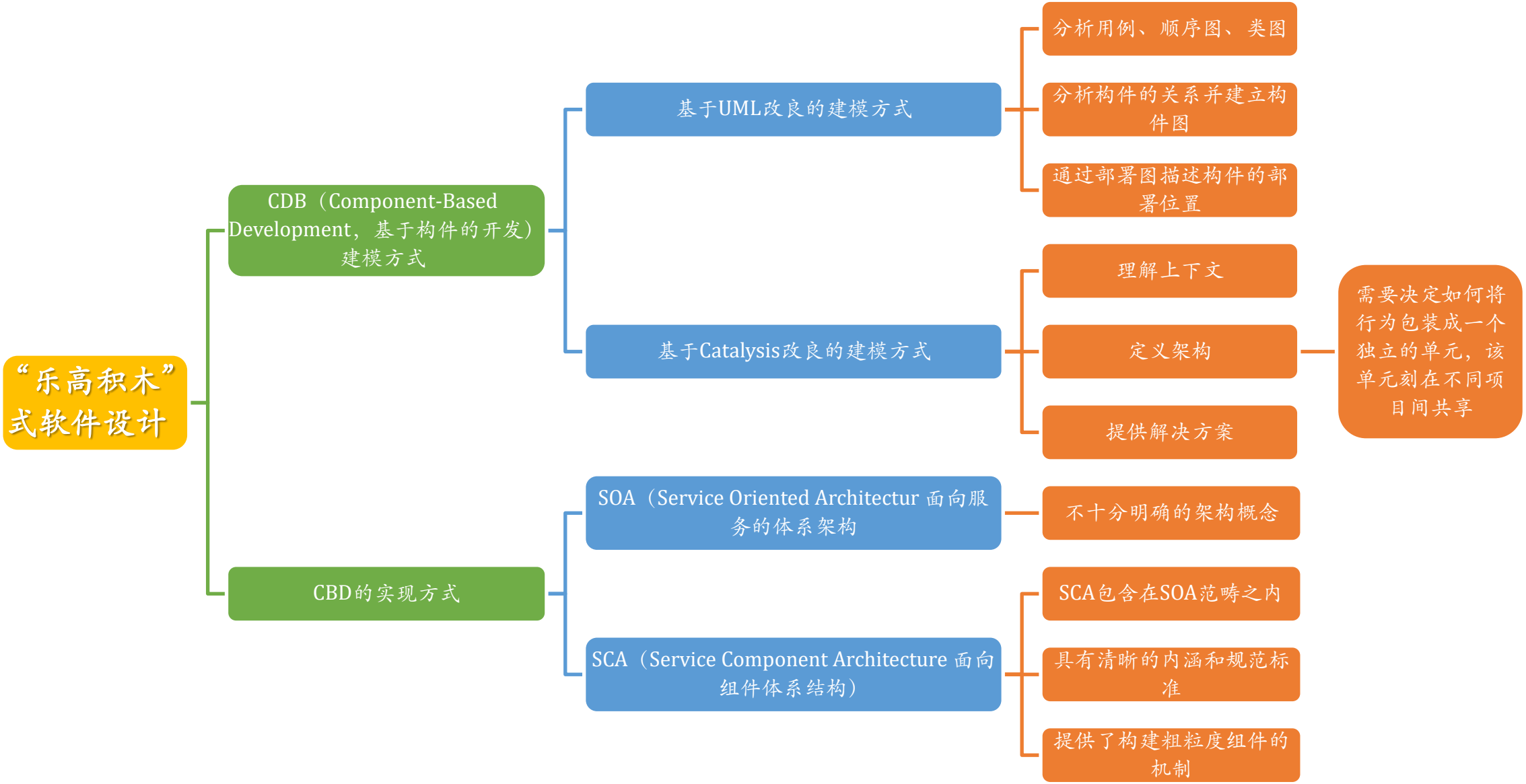
- 15.1 构件模型的抽象要素及逻辑关系
- 15.2 轻量级架构管理工具的设计原理
- 15.3 采集项目信息的价值
- 15.4 轻量级架构管理工具的优缺点
- 15.5 应用轻量级架构管理工具管理新需求

第16章 基于构件模型谈谈传统企业的产品创新

- 16.1 信息传导：打造信息传递高速公路
- 16.2 信息分析：创造高维数据
- 16.3 创新平台：扩展构件模型
- 16.4 构件模型及其应用设想的不足

第14章 如何支持面向构件的设计

14.1 “乐高积木”式的软件设计



14.2 “颗粒度”问题

“颗粒度”很重要

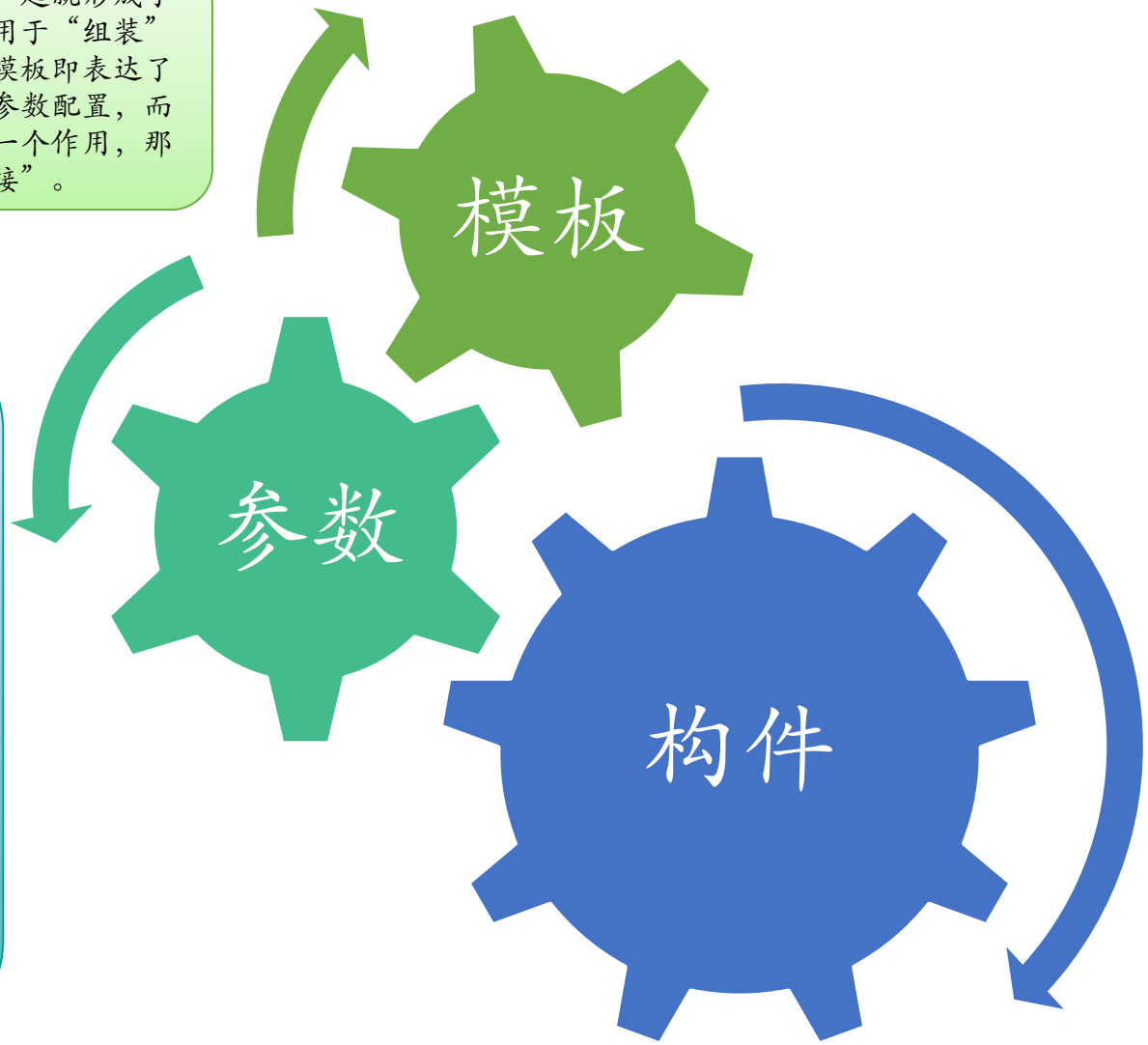
- 构件开发并不是服务之间可以互相通信、互相调用就万事大吉了，这样的逻辑对于技术人员而言好像还可以接受，但是却**无法传递给业务人员**，这种逻辑成了单纯的技术组装。
- 如果服务“颗粒度”**划分过细**，那么这样的构件开发将会拖累死通信机制，最后形成一个混乱的网状通信，使得迭代、升级变得异常复杂。
- **“颗粒度”若处理不好**，那么业务人员将无法明白IT到底在做什么，IT人员也无法让自己得到解脱。
- 不关心业务人员是否理解，只在乎**技术“自由度”**的设计方式将使技术人员也难以确认自己的“组装”到底是不是业务人员期待的“组装”

现有架构设计方式的不足

- **SOA**的缺点是在实际操作中并不真正关心“颗粒度”问题，一个遗留系统既可以直接被封装成一个服务，也可以将很小的功能服务化，二者的地位是一样的。所以，经常有人说，SOA本质上是一个集成架构，它能有效地解决异构系统的集成问题，统一内部的通信方式，重担一般会直接压给企业总线。
- **微服务**虽然很关心“颗粒度”的问题，但是却很难判断服务的合适大小。若服务太大了，则内聚性不好；若服务太小了，则通信会过于复杂，从而降低效率。目前微服务领域也只存在一些参考性的原则，并没有通用的标准。
- **DDD**方法指导微服务设计，并取得了一些成果。但是DDD方法本身的学习门槛比较高，不容易掌握

14.3 构建模型的设计方式

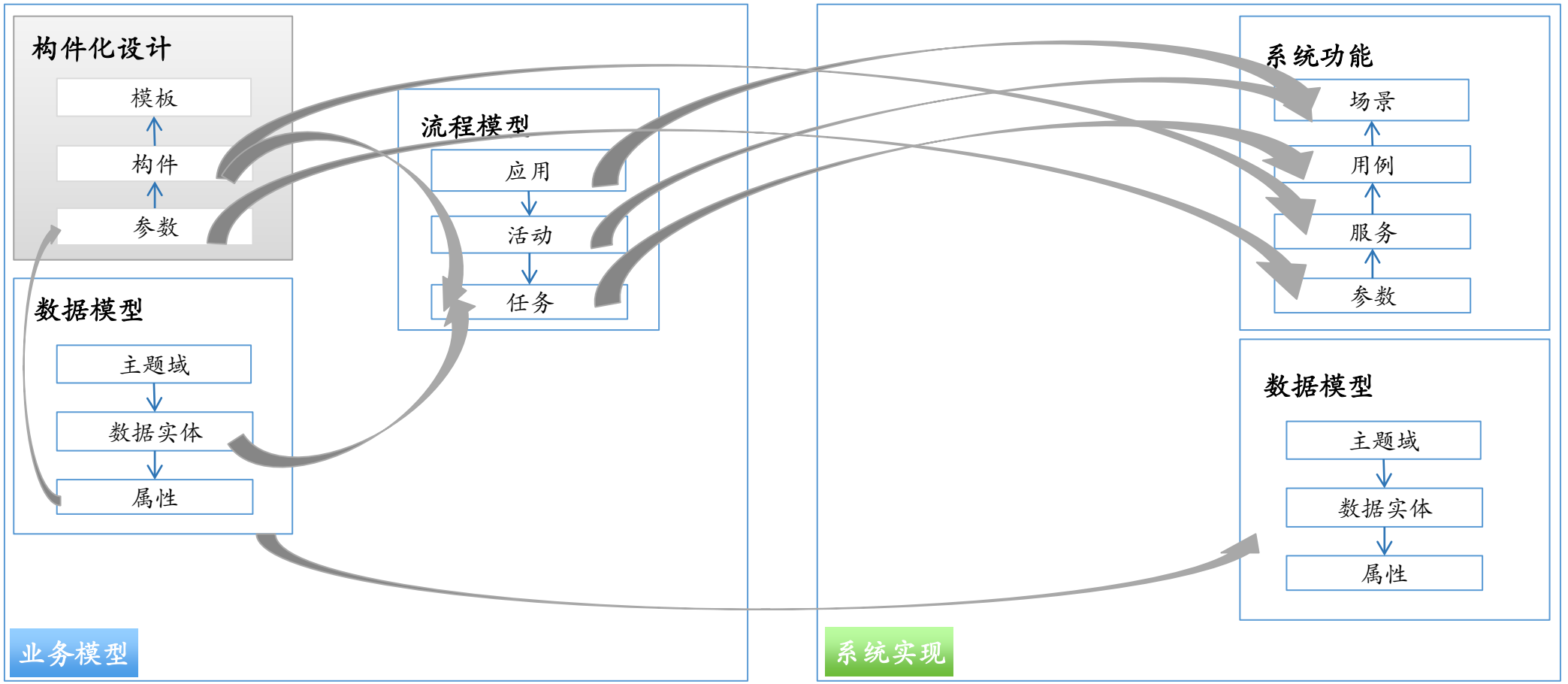
- 这些带有参数的构件加在一起就形成了一个有结构的装配模板，用于“组装”业务实例或产品，这样的模板即表达了构件化设计，又可以用于参数配置，而模板本身也可以起到另外一个作用，那就是服务的编排或者“粘接”。



- 参数化设计可以通过参数配置快速刷新产品。
- 在金融领域，参数化设计对于代理保险、实物贵金属、理财等标准化程度高的产品非常适用，因为产品之间几乎就是参数取值方面的差异，这点对于制造业来说也是一样的。对于数控机床之类的高精度工业机械而言，参数化能力比金融服务更强。
- 构件中的服务会涉及参数设计，参数的选择既可以定位在需要配置的参数，也可以更加宽泛地表达为构件包含的服务的外部输入报文集合和最终输出报文集合，这样做可以更方便地进行构件的组合设计。
- 严格来说，这些参数应该都是数据模型中的属性，可以在数据模型的属性上增加参数标识，以便在数据模型中可以很容易识别出来。

- 构件构件化开发要求更强的结构化，设计时应将设计对象切分成“零件”，通过组装“零件”、调整“零件”来快速实现新设计。
- 金融领域中的大部分产品与流程其实是“一体两面”的关系，将一段流程包装起来销售就成了产品，产品的“零件”自然也可以来自于流程。
- 流程在业务建模时就已经标准化成任务了，那么再进行分析设计，自然就是针对任务进行调整。将一个“任务”再细分成若干个“零件”，抑或将多个任务聚合成一个“零件”，这些打破了原有任务结构的“零件”就是可用于“组装”的“构件”。
- 构件的切分原则应当是每个构件都能够给出明确的业务含义，而不是单纯地为了给服务分分类，这种含义还应面向业务可“组装”的目标。

14.3 构建模型的设计方式

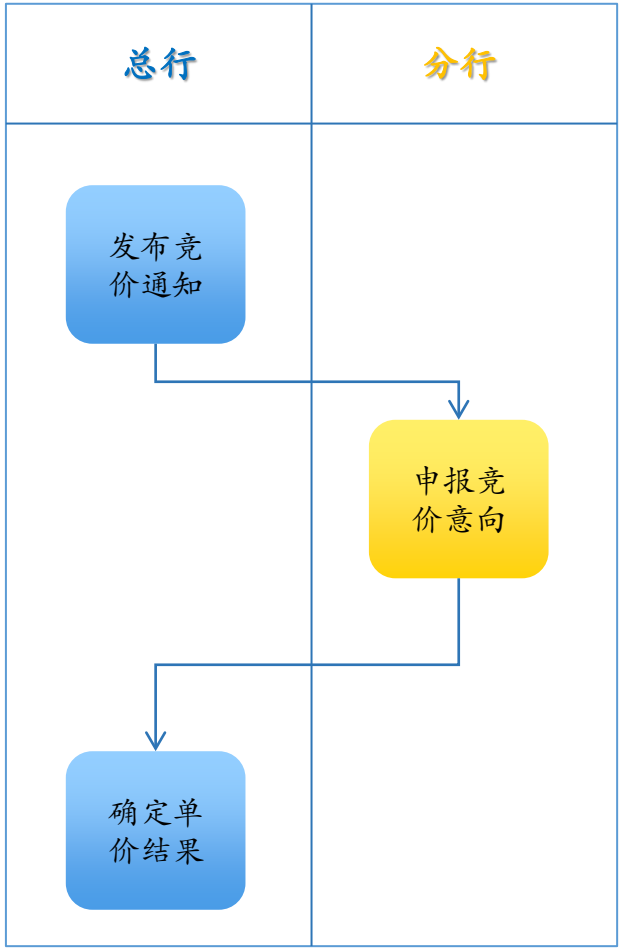


构件模型与业务模型、系统实现的关系示意图

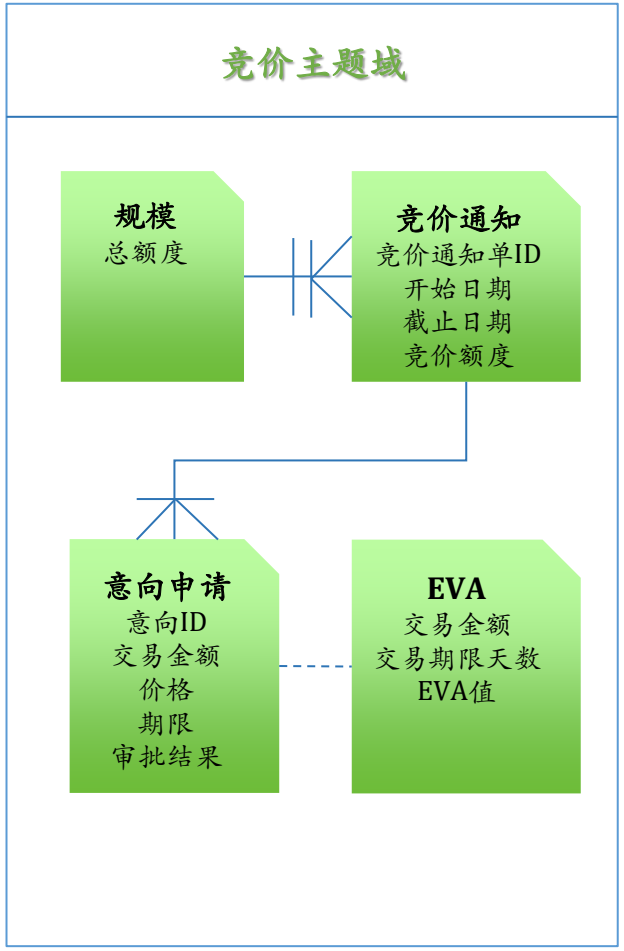
14.4 建立构件模型的虚拟案例

1、采用非构件模式实现的金融产品竞价功能

流程模型



数据模型



竞价过程的业务模型示意图

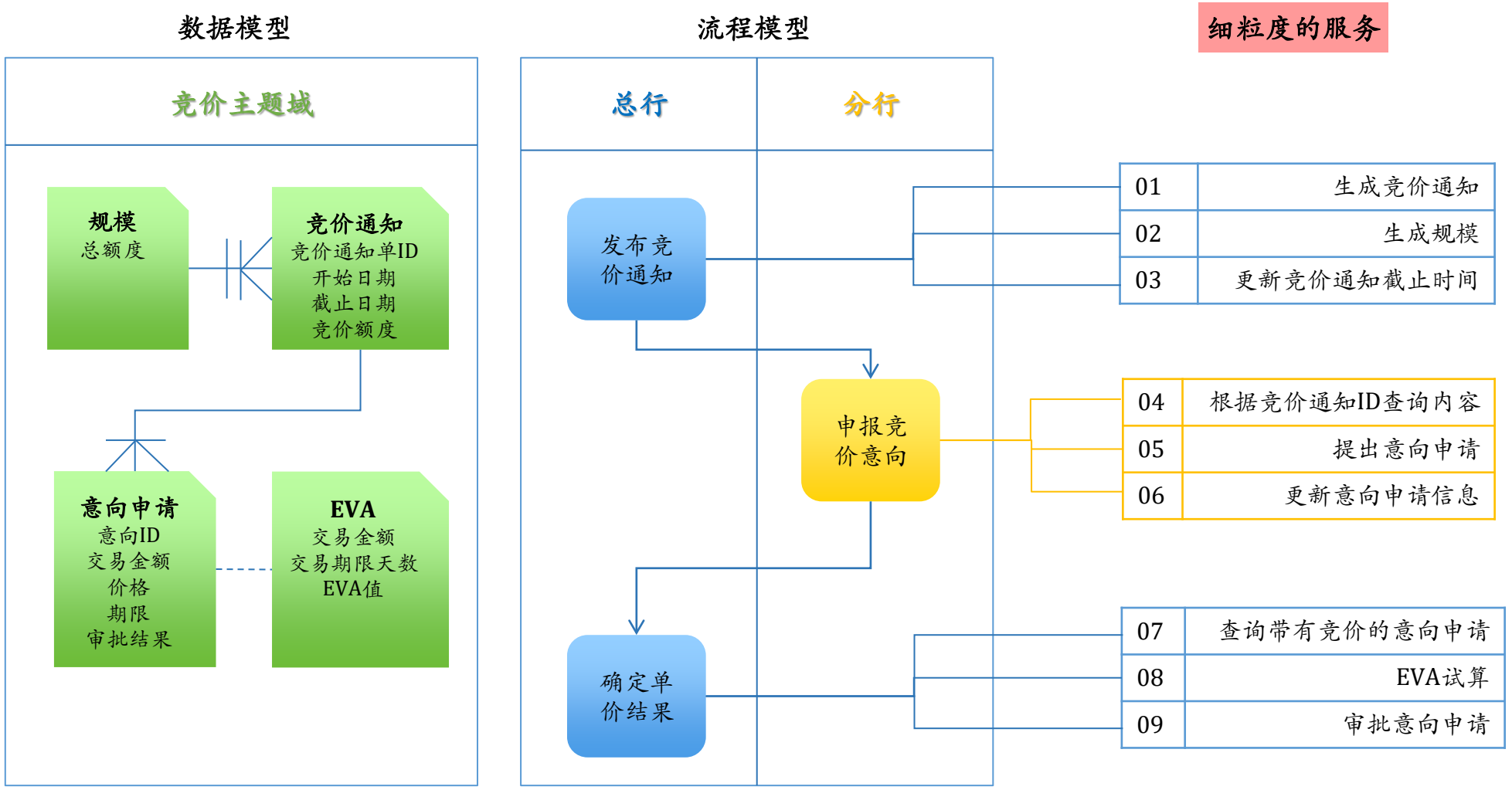
银行体系中的总行，总行具有较大的资金调度权力，可以通过对某笔闲置资金进行内部竞价，从而确定出价最高的分行可以运用该笔资金去经营业务以获取收益。因为通过竞价决定资金归谁使用其实是有机会成本的，因此竞价过程中可能还需要通过计算EVA（经济增加值）的方式，进一步衡量其经济贡献。

竞价过程的流程模型假定包含了3个任务，数据模型简化处理后设计了4个数据实体。总行发布竞价通知，这时可能会创立总额度，一个总额度可能拆分成多个竞价通知，每个竞价通知包含不同的额度；之后分行根据竞价通知申报竞价意向，报出自己的价格；总行将各分行的意向申请综合审核，计算并比对EVA值，然后确定最终的竞价结果。

3个任务可能会拆分成9个服务，这是很常见也很正常的拆分方式。通过对用例进行分析，做操作级的设计实现，即可能将任务拆分成很多细粒度的服务（如果考虑更多的业务细节，可能还要拆分得更细）。如同前文所述，对技术人员而言，这也的确是一种“组装”，具有“构件化”的特征，也符合SOA架构风格。

14.4 建立构件模型的虚拟案例

2、采用构件模式实现的金融产品竞价功能

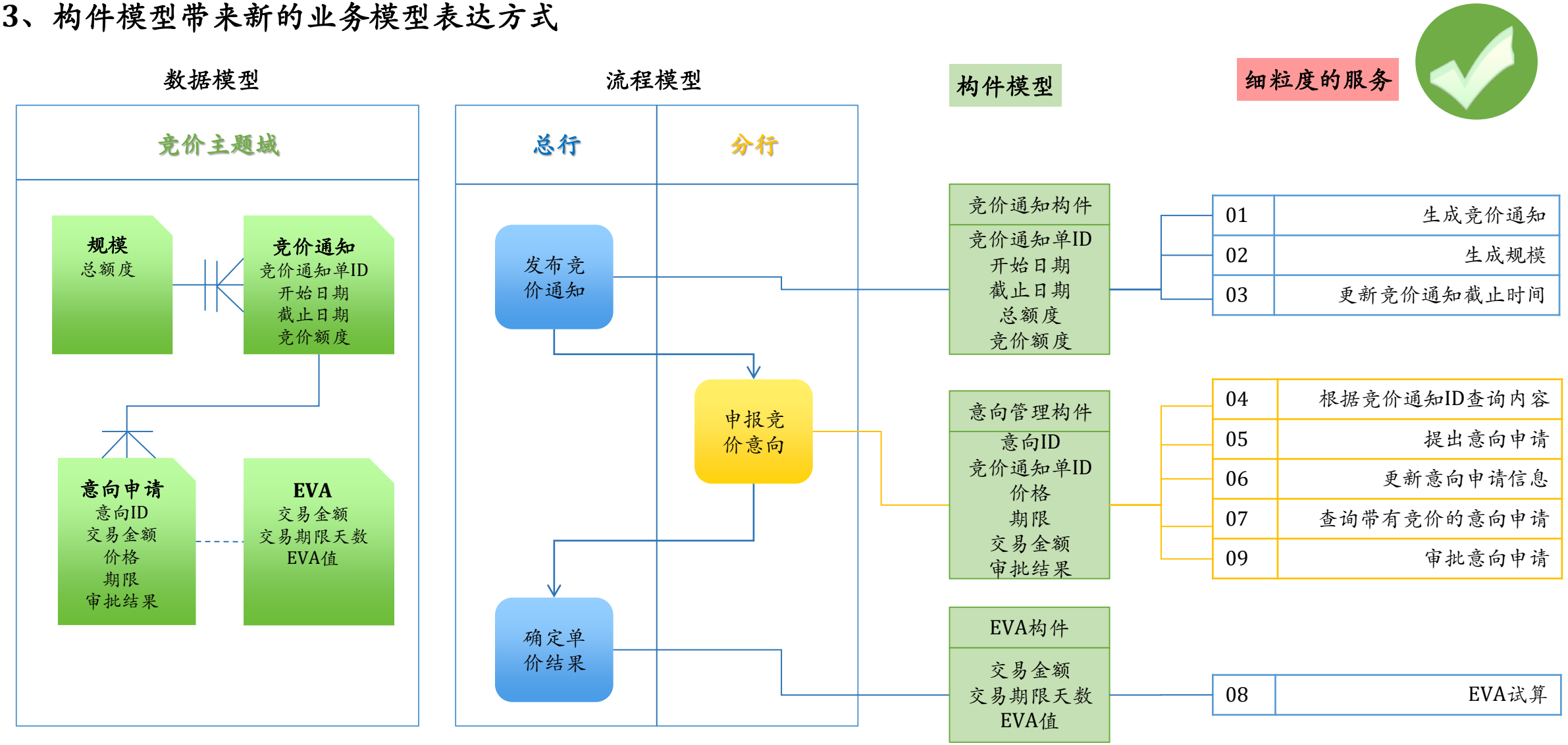


拆分过细的开发结果示意图

- 业务需要的不是面向操作步骤的过“细”的服务划分，而是面向如何更好地响应业务变化的服务划分方式。
- 基于已经完成标准化的业务架构模型，在对任务进行实现时，从企业级视角通盘考虑，如何才能设计出更有业务含义的“构件”。
- 业务只需要两个看起来较“粗”的流程构件和一个具有更强通用性的纯粹能力构件，这种划分思路更加面向业务对“组装”（或者说“复用”）的实际需求，而非常见的、面向操作实现的、较“细”
- 粒度的服务划分方式，不只是要满足技术视角的“组装”，同时也要满足业务视角的“组装”。

14.4 建立构件模型的虚拟案例

3、构件模型带来新的业务模型表达方式

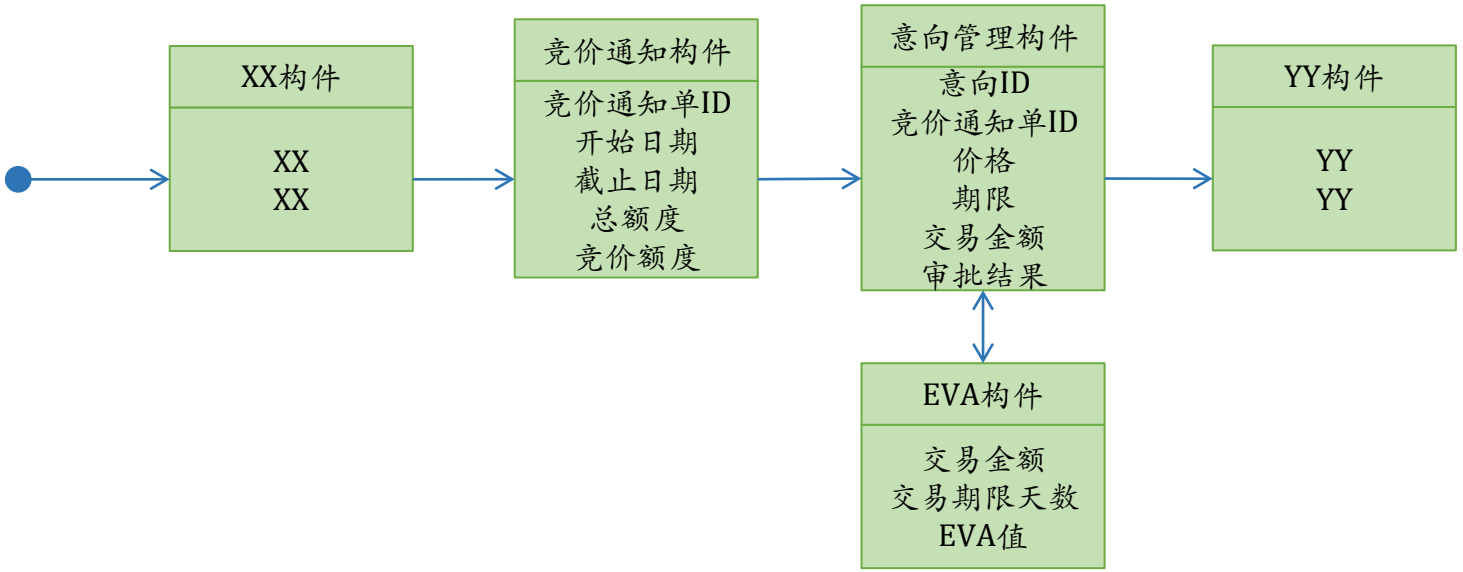


构件模型与业务模型、服务之间的关系示意图

14.4 建立构件模型的虚拟案例

3、构件模型带来新的业务模型表达方式

- 构件模型的思路非常符合企业级业务架构的设计目标，其本身相较于流程模型来说更贴近IT设计，因此构件模型在业务架构向IT架构过渡方面可能会发挥更大的作用。
- 这种模型打破了原来业务模型较为单纯的业务属性，将业务含义与具体实现直接结合在了一起，而且是基于构件串起来的执行流程，其是一种可以在一定程度上满足流程与能力解耦的表达方式。
- 构件模型也会因此打破原来流程模型中的任务边界，产生新的模型表述方式，这种情况下，可以选择是否将其视为流程模型的一次新视角的迭代，替换掉原有的流程模型。
- 构件模型带来新的业务模型表达方式构件模型的思路非常符合企业级业务架构的设计目标，其本身相较于流程模型来说更贴近IT设计，因此构件模型在业务架构向IT架构。



通过构件模型展示的流程图

14.5 构件模型的技术设计建议

- **1.参数的实现**

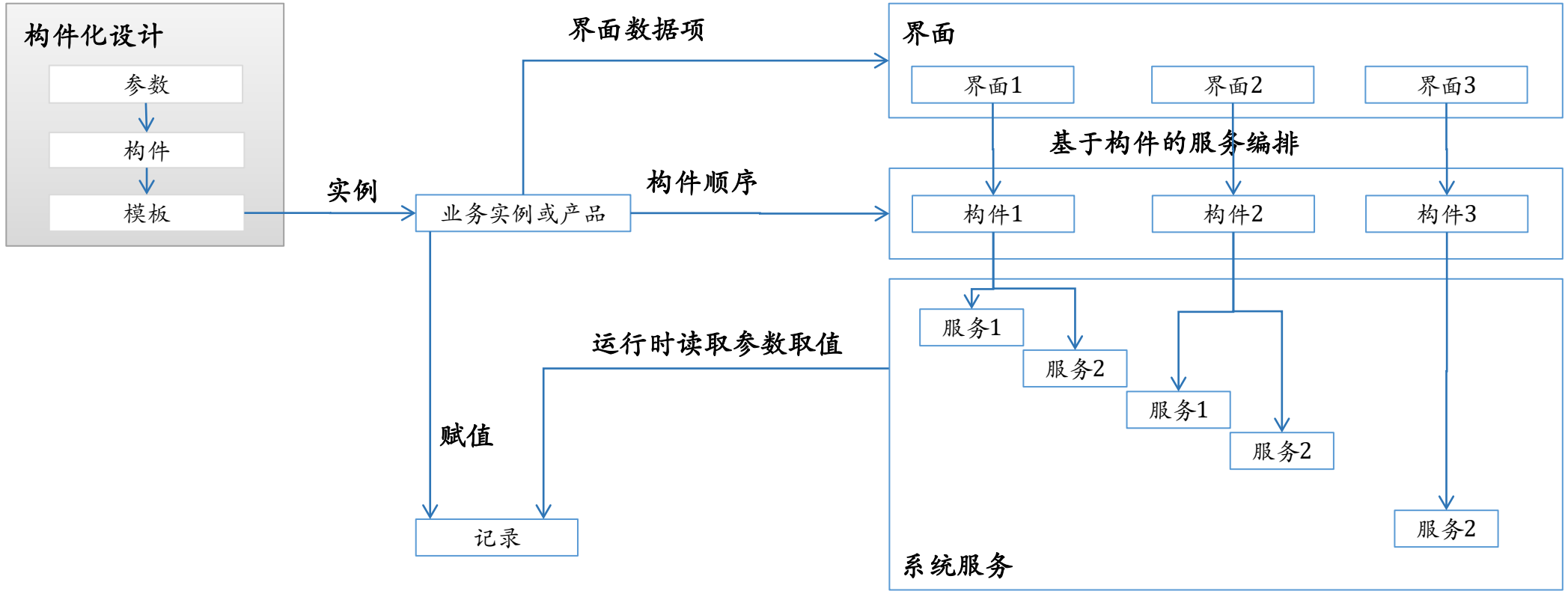
实例化的结果中包含参数清单，其中可配置的参数可以基于业务操作形成的赋值结果生成参数记录，用于服务被调用时提供参数，参数中包含的界面数据项可用于界面的动态生成。

- **2.构件的实现**

实例化的结果中还包含构件清单，构件清单可以用于服务编排。构件的执行顺序可以串成流程，也可以在服务编排中用于确定构件的执行顺序，也可以基于界面进行触发。构件根据其与服务的关联关系，可以在其被执行时确定服务的调用顺序。

对于流程明确的情况，但其实流程是否发生变化并没有很大的影响，实例化之前流程能够确定即可。

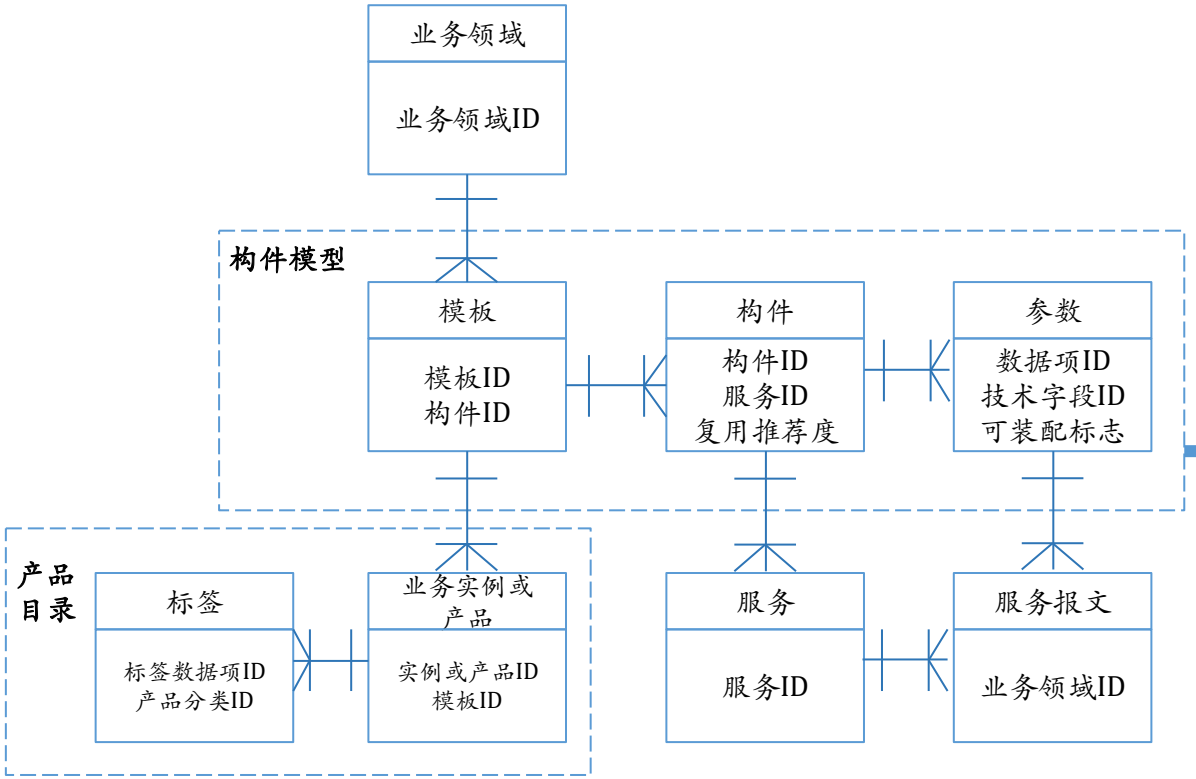
而对于实例化之前流程不能确定的情况，则可能无法实现构件顺序的预定，只能单个触发构件执行，而构件执行时服务的执行顺序相对来说还是比较稳定的。



构件模型的技术实现示意图

第15章 构件轻量级架构管理工具

15.1 构件模型的抽象要素及逻辑关系



构件模板的抽象结构

1.模板与构件

- 每个业务领域下都可能有一到多个模板用于设计业务实例或产品；模板可以包含若干个构件，组装式开发可以表达为业务实例或产品与模板、模板与构件间的对应关系。

2.构件与参数

- 构件中可以记录复用推荐度，以方便业务人员后续做设计时使用；构件中还会包含多个参数，参数应尽量使用数据模型中的数据项，但是实际操作中也可能需要列入一些与业务无关的技术字段。此外，应该为每个参数注明是否可供业务人员直接配置，若是不可直接配置的参数则不提供面向业务人员的配置界面。

3.构件与服务

- 一个构件可以对应一到多个实现上的服务，构件此时代表的是一个服务集合。对构件的复用并不是指去任意复用SOA中的服务，而是构件对应的服务整体对外提供一种能力，这才是“零件”的含义，否则，构件就不是一个真实的存在。如果原有构件中的一部分服务又被集合成了新的能力，则应再增加一个构件与之进行对应，这样的构件才真正具有部署的含义。

4.服务与报文

- 由于服务可以被调用，因此服务会对应报文，报文既包括请求报文，也包括响应报文。报文中又会包含与构件对应的参数，二者既可以合二为一，也可以分开表达。

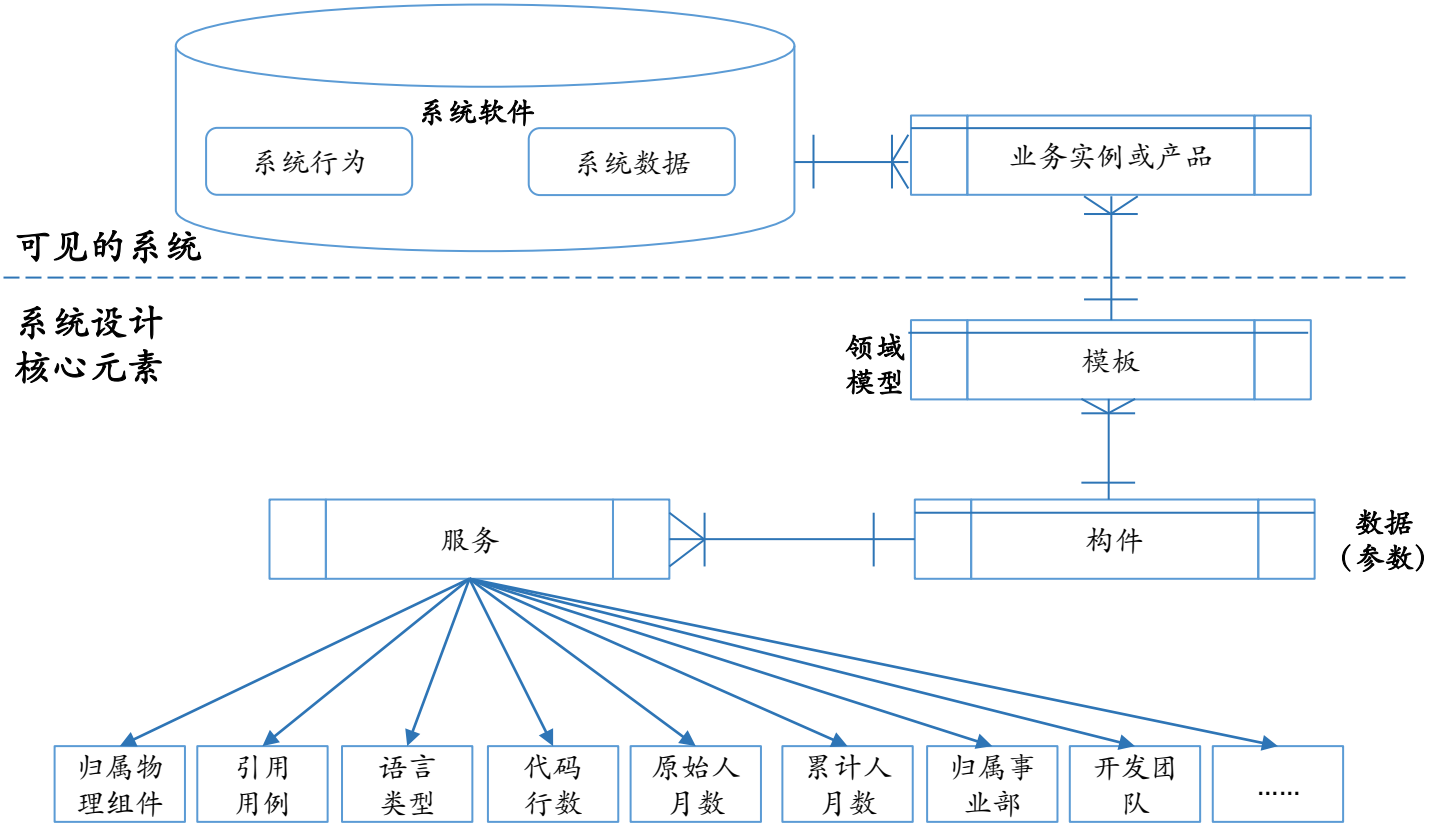
5.实例化

- 模板在生产中会派生为业务实例或产品，在产品销售前及销售过程中，所有的参数都将完成赋值。

6.产品目录

- 每个业务实例或产品都会有描述性的标签信息，这些信息通过集合形成产品目录。

15.2 轻量级架构管理工具的设计原理



轻量级架构管理工具的逻辑图

- 业务实例或产品由模板配置而成，模板实际上是一种领域模型，不同领域的产品可能差异较大。模板之下是构件，构件代表了一个领域的具体组成部分，是“零件”。构件中包含数据，也就是参数。
- 构件又可以进一步分解为服务，服务实际上就包含了行为，如果是微服务设计，则也会包含对应的数据。服务作为设计上的底层核心元素，可以从统计角度包含服务归属的物理组件、引用该服务的用例、语言类型、代码行数、初始开发或累积的人月数、归属的开发团队等可用于项目管理的信息。其中一些信息可以通过工具或配置信息来获得，另一些则需要人工维护。

15.3 采集项目信息的价值

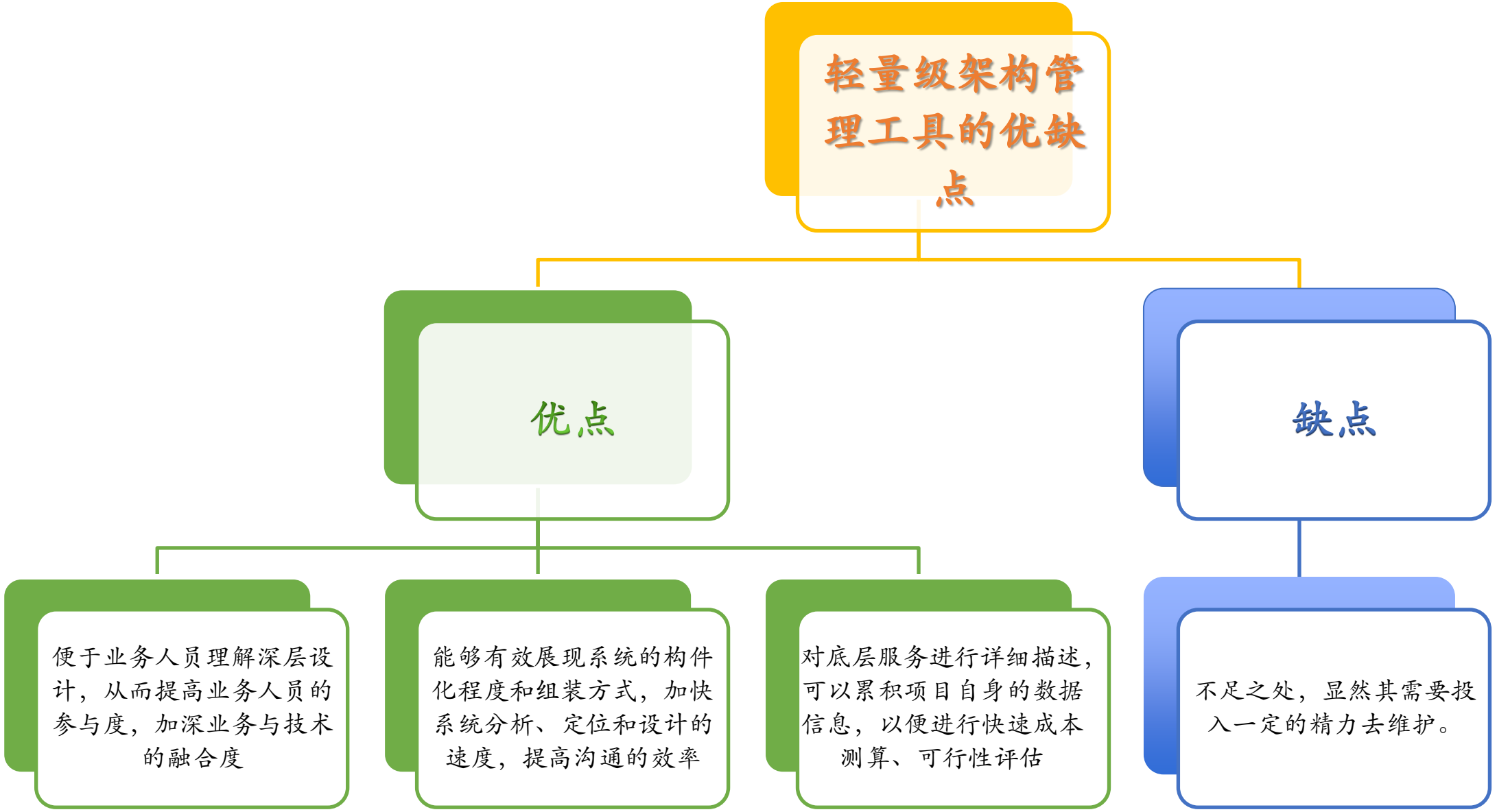
信息采集会带来一定的工作量

- 信息采集会带来一定的工作量，但是，其所累积起的项目信息库对于大型企业的项目管理而言是非常有价值的。毕竟目前的情况是，一个企业可能做了不少项目，其中不乏大型项目，但是积累起来的、可以用于项目快速决策的管理信息却少得可怜。只知道项目最终花了多少钱，却不知道钱都花到哪里去了，也不知道系统中的核心部分到底花费了多少成本。系统再次进行更新改造、新功能上线时，预算基本上还是采用功能点估工作量的粗略估算方法。

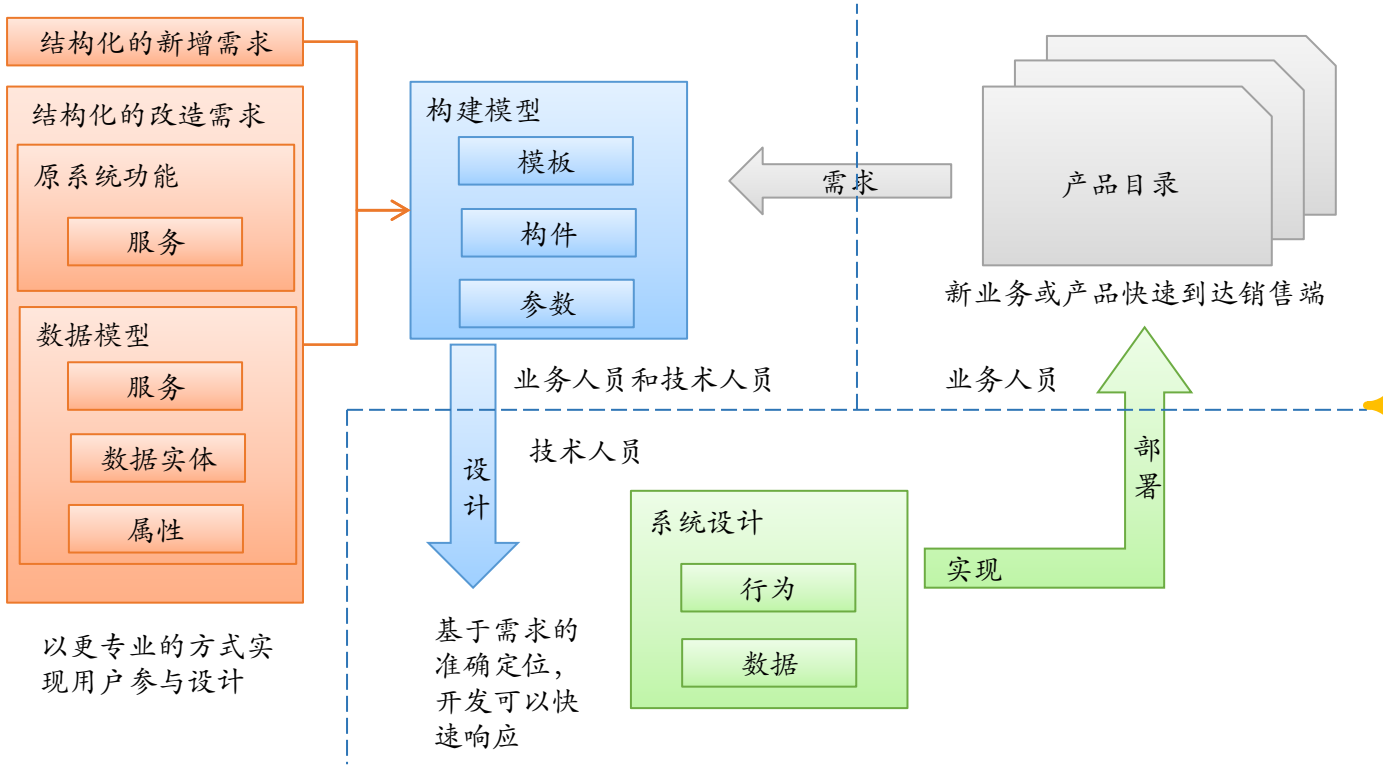
考虑设计一套逻辑进行项目信息的有效收集和分析

- 如果信息采集过程可靠，那么以这个逻辑建立起来的平台不仅可以用于支持快速的架构设计，还可以将项目成本分解至服务层面，甚至基于这一点来比较团队的开发效能。总之，需要考虑设计一套逻辑进行项目信息的有效收集和分析，否则项目开发永远无法向“精益”方向靠拢。

15.4 轻量级架构管理工具的优缺点



15.5 应用轻量级架构管理工具管理新需求



应用轻量级架构管理工具管理新需求

构件模型是基于系统结构和关系的，其通过一条顺序流“串”起构件，形成完整的业务处理过程。

1.原有功能改造需求
业务人员在产品销售或服务提供过程中产生新需求，通过产品与模板的对应关系，找到实现产品的构件模型，与技术人员共同基于构件模型分析产品需求的实现位置。如果是对原有产品进行改造，则可以根据构件的切分，快速找到需求的实现位置，进而定位到需要改造、新增的服务及数据。

2.新增功能需求
如果是原先没有的业务环节或者是全新产品（这种情况其实较少），则会产生构件级别的新增。但是基于原有的业务环节，可以很快定位出新增环节与原有环节的关系，设计前后构件间的数据关系、构件接口。
在这种分析模式下，可以让业务用户以更为专业的方式高效地参加到业务或产品的设计过程中，将更加精准的需求传导到开发环节，从而提升开发效率。

业务架构的设计成果要想具有生命力，最重要的莫过于经常使用，这一点对任何架构设计模式来说都是一样的。使用该工具管理新需求，其目的就是将业务架构变成连接业务与技术的“通用语言”，使用越多则沟通越容易，也就越容易被各方接受以用于沟通，这是一种正向循环。

第16章 基于构件模型谈谈传统企业的产品创新

创新管理三要素

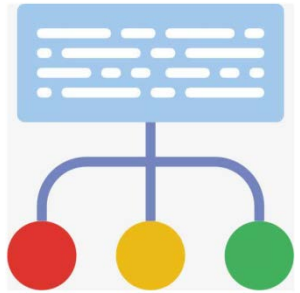
创新管理三要素

信息传导

- 由内向外传递
- 由外向内传递
- 通过产品目录构建起“产品信息传递高速公路”

信息分析

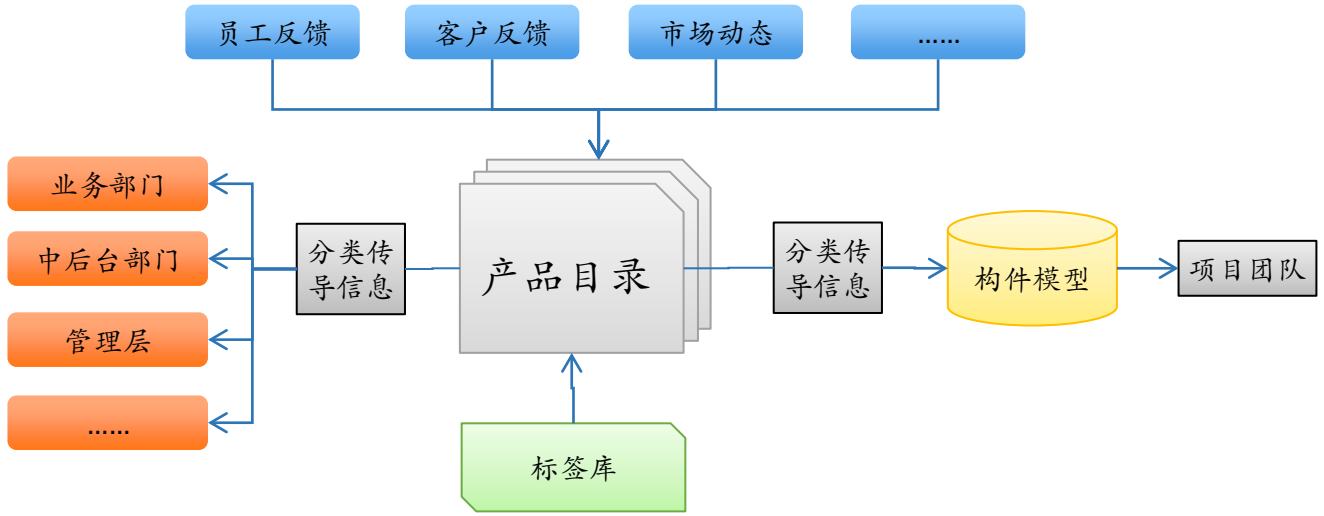
- 用标签打造高维数据
- 标签比分类更容易做企业级推广
- 构建业务与技术兼顾的产品目录



创新平台

- 核心域
- 支持域
  - 1.产品创意域
  - 2.产品评价域
  - 3.产品运行效率监测域

16.1 信息传导：打造信息传递高速公路



通过产品目录传导产品信息示意图

可以包含设计者、管理者、销售方、合作方、产品特征、适用客户群体等各类描述信息，也就是“标签”。

产品目录能够连通公司内外，连通业务和技术，将市场信息、客户信息、销售信息、伙伴信息等各种信息通过高维数据结构组织起来，让海量信息分门别类地流向信息需求方，成为一个高效的信息传输渠道

16.2 信息分析：创造高维数据

用标签打造高维数据

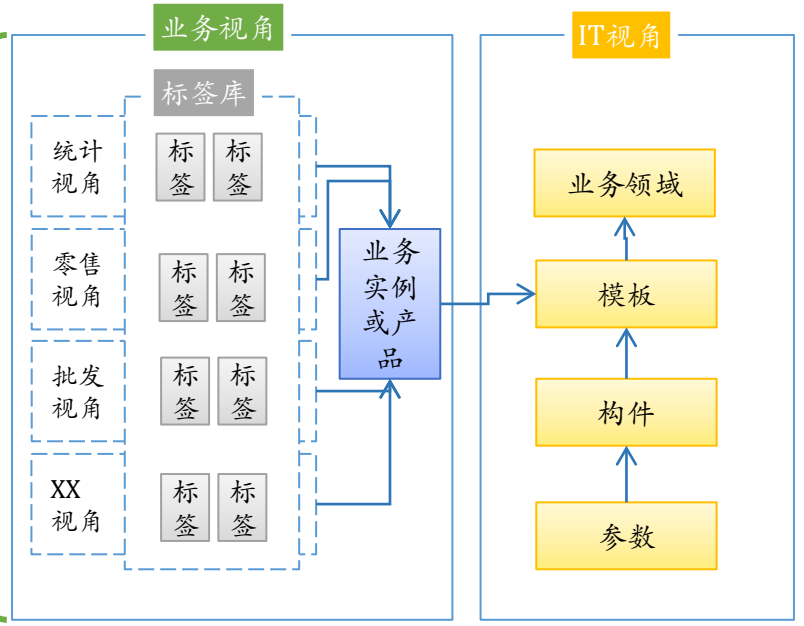
- 具备高维度特征的产品信息库才具备构成产品大数据能力的基本条件。
- 一个产品能够具有多少有效的标签，反映了产品适用范围的宽窄、承载信息能力的大小、对象描述能力的强弱。
- 标签不仅用于分类，与分类相比，标签应尽可能满足各种视角的灵活展现
- 标签数据包含标签、产品，以及标签与产品的关系三个部分。一般分为用户打的标签和系统打的标签两类。

标签分类更容易做企业级推广

- 部门级标签分类，难度不大；如果企业级的，则可能就不太容易了
- 每一类不同的需求都可以转化成不同的标签集合。为产品赋予大量的标签，可以满足在不同视角展示和应用需要，这也就减少了部门对分类这种相对固定又带有一定权威象征的资源的关注。

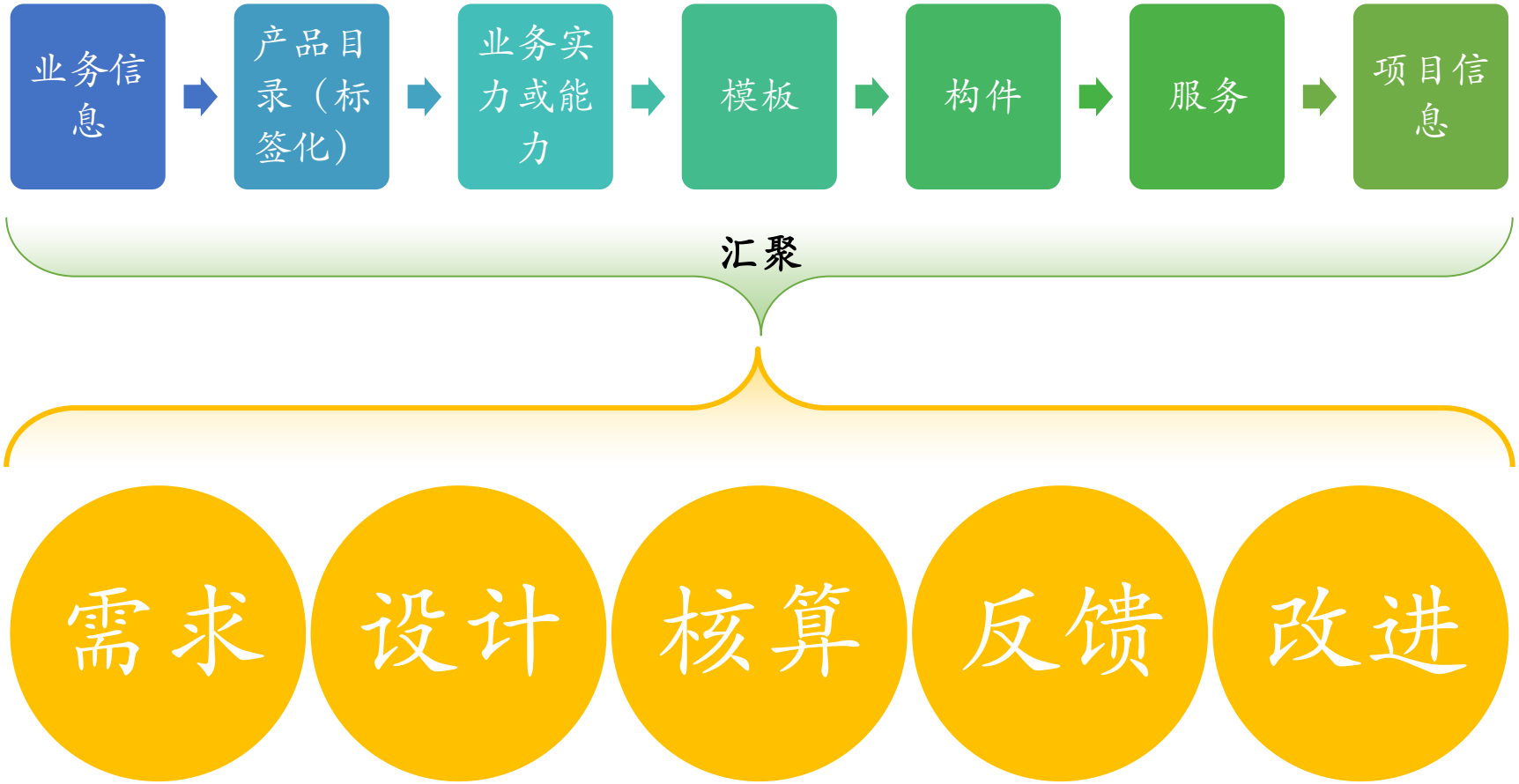
构建业务与技术兼顾的产品目录

- 从设计的角度来讲，基于构件模型，设计人员应该关注的主要是产品和模板的映射关系
- IT人员多看待产品时并不需要那么维度，构件模型本身就是IT侧需要看到的“产品目录”



16.3 创新平台：扩展构建模型

由业务延伸到技术的链条

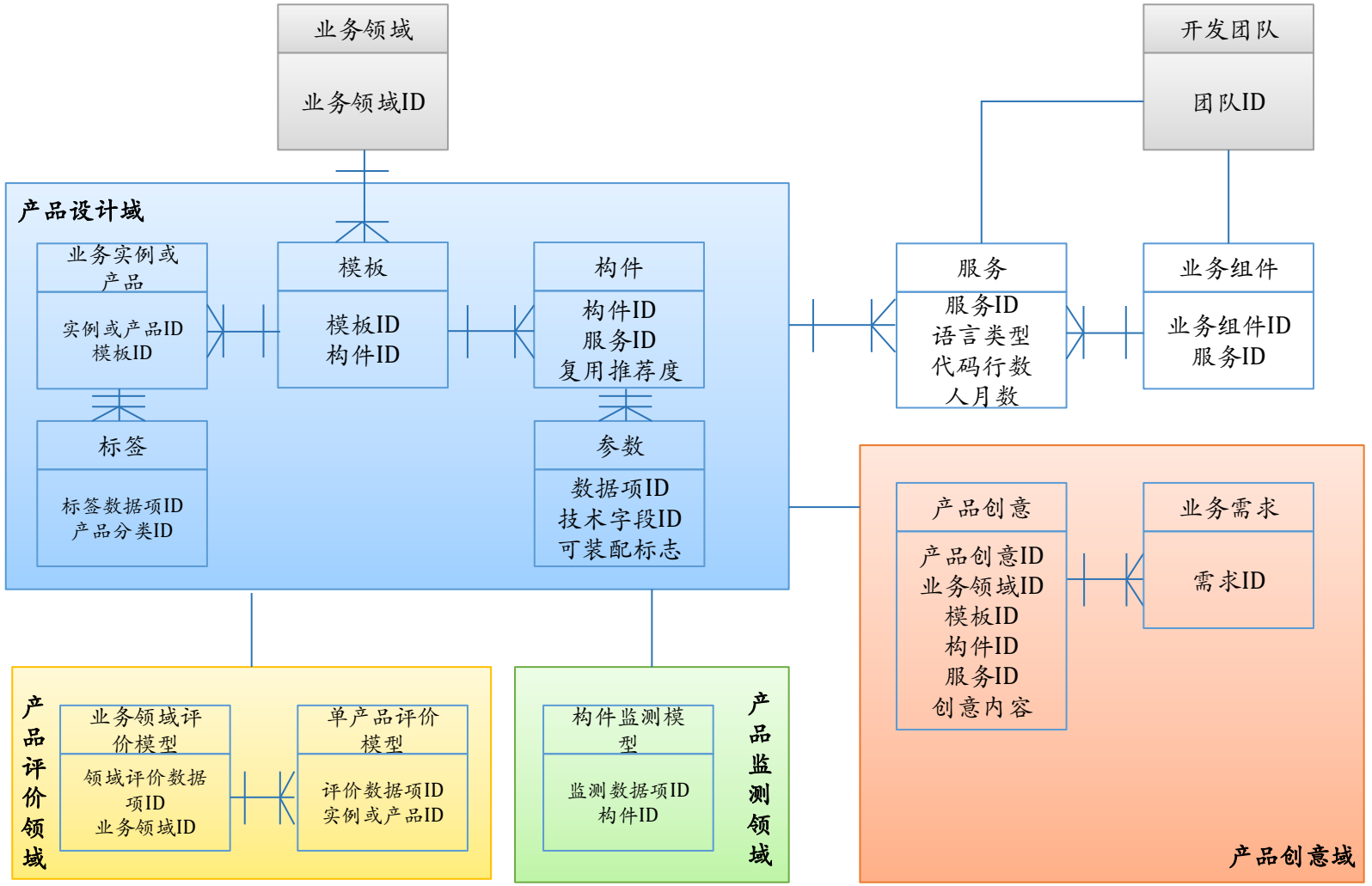


以这些信息为基础，搭建一个产品创新管理平台

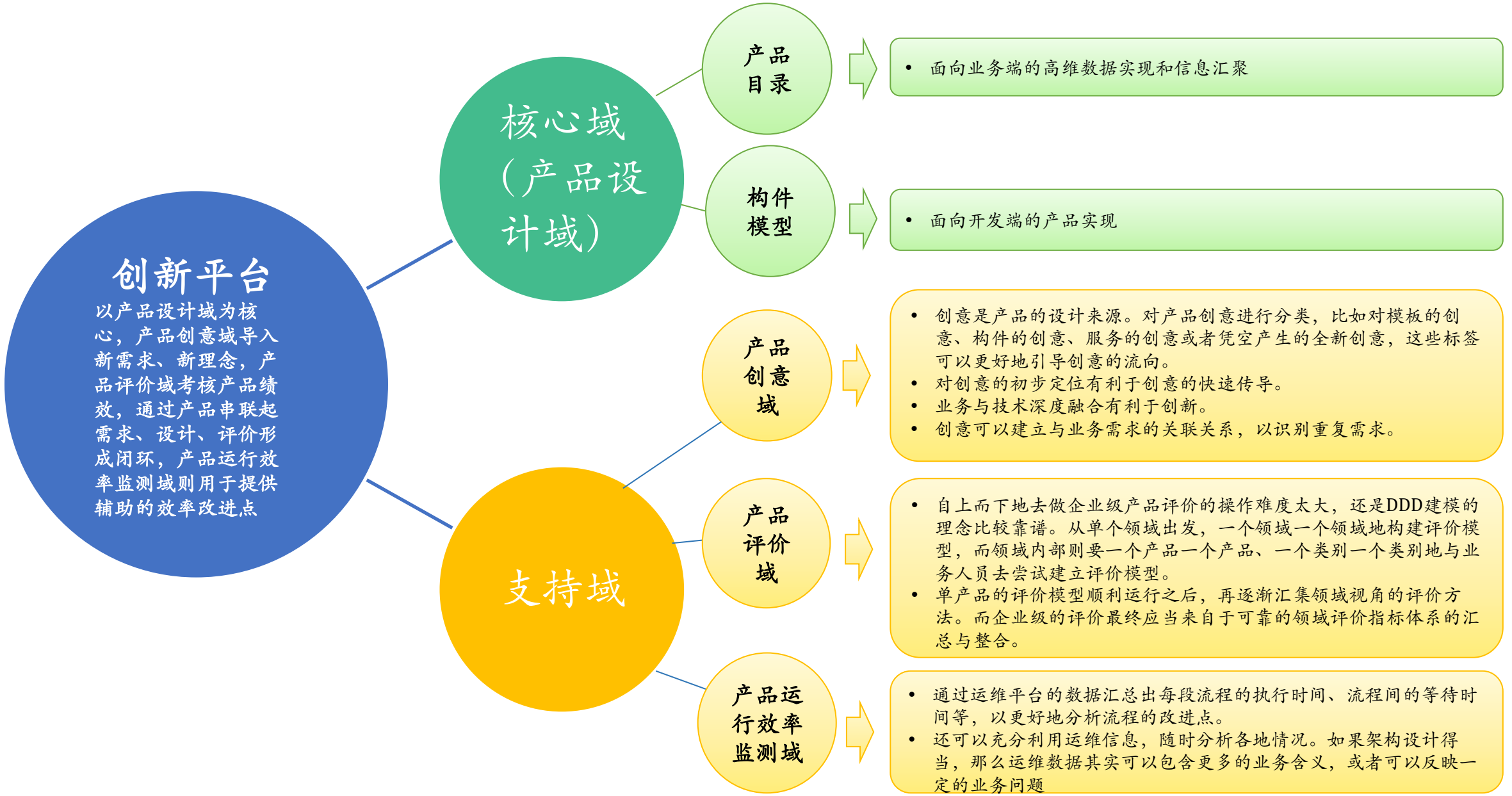
产品的生命周期

16.3 创新平台：扩展构建模型

产品创新管理的平台设计逻辑图



16.3 创新平台：扩展构建模型



16.4 构件模型及其应用设想的不足

一

- 不易于在企业级项目初期产生相对成熟的设计，而是需要经过反复精炼。

二

- 架构设计的模型形态与业务人员较容易理解的流程模型形态之间存在较大的差异，因此需要一定的学习成本。

三

- 构件类型作为架构管理工具，需要经过一定时间的数据积累才能实现，且成本统计可能会由于数据录入结果等人为因素而不准确。

四

- 产品目录的信息传导能力也需要较长时间的建设才能形成。

- 在企业级转型项目结束之际，形成自己的方法论，培养出自己的方法论专家，这是知识沉淀与升华的过程，不能只关注“行线”的完成，而不关注“知线”的建设。
- 这是对社会的最大回馈，在架构领域尤其如此。这才是数字领袖最难得的境界。

第17章 中台之上

- 17.1 阿里中台简介
- 17.2 企业文化的作用
- 17.3 由业务架构方法可以推导出中台设计吗?

17.1 阿里中台简介

是一个累积的过程

- 从2009年建立共享事业部开始，2015年才正式发展成中台战略。
- 大型架构、好的架构都不是一蹴而就的设计，而是根据实践不断磨合、调整得来的。

包含了十几个共享业务单元

- 包括用户中心、商品中心、交易中心等。淘宝、天猫、聚划算等25个大型业务应用都是由中台的共享业务单元支持的。
- 共享业务单元的划分原则其实不是简简单单就可以掌握的，而是要综合考量设计、运营和工程因素，尽可能遵循“高内聚、低耦合”“数据完整”“业务可运营”和“渐进”的原则。
- 阿里在划分共享业务单元时非常重视其业务价值和基于业务的设计，而且设有业务架构岗位，每个共享单元都有业务架构师，并将其视为非常稀缺的“复合型人才”
- 总体来讲，其业务架构仍然是领域性的。

要解决的核心问题是高并发、可扩展

- 采用去中心化（也就是去ESB）的HSF分布式服务框架，以支持服务的点对点调用，用于解决ESB可能产生的瓶颈问题
- 采用微服务设计方式，提高变化响应
- 研究DDD（领域驱动开发）等设计模式，以提升设计效率
- 自研设计了分布式数据层框架TDDL（Taobao DistributedData Layer，又称“头都大了”）以及分布式数据库DRDS
- 研发了支持分布式事务处理的AliWare TXC
- 支持高效故障定位和运维监控的鹰眼平台
- 实现了限流和优雅降级设计，以及做保障的全链路压测平台、业务一致性平台
- 去IOE化

实现的目标

目标一

实现了以“厚重”的共享中心支持“灵活”的前端应用

目标二

实现了业务与技术的充分融合

17.2 企业文化的作用

- 阿里集团的优秀也不是单纯学学中台技术就可以复制过来的。
- 阿里集团技术上的成功离不开其滴水穿石般逐渐形成的企业文化。

1.明确底线

对诚信问题的“零容忍”

带有末位淘汰性质的考核机制

“底线”将员工“逼”到了一个必须有较强自律性、自我负责的状态。

2.开放上限

在人员晋升方面拥有一个开放的“上限”，晋升主要是拼个人实力

有多种序列可供员工选择，前中后台，不同条线的员工大致都可以达到相同的晋升高度，在某一序列的发展中如果遇到瓶颈就可以很方便地进行转岗

3.鼓励好奇

鼓励员工不断探索，自我驱动，挑战极限。

17.3 由业务架构方法可以推导出中台设计吗？

业务架构方法可以推导出中台设计

- 阿里集团对共享中心的设计，与业务组件的设计实际上是异曲同工的。每个共享中心在前端支持上，都是既包括单独服务的调用，也包括整段流程的封装，这与企业级业务架构设计，包括作为改良建议的面向构件的设计，在逻辑上并无太大差别。
- 共享中心设计中对工程上的考虑，其实质是从实现的角度对业务组件或者构件进行的分组。所以，完全可以基于企业级业务架构设计得出中台规划方案。
- 中台模式与组件化一样，都是实现快速响应的方式，企业级业务架构产生的组件化模式，核心优势还在于有效连接战略与开发，实现上下贯通的一体化设计，这是中台模式考虑较少的地方。

业务架构方法具有更大的优势

- 企业级业务架构是连接企业顶层战略和技术实现的桥梁，是业务人员和技术人员互相沟通理解的桥梁。业务架构基于企业目标进行业务能力和流程的整体规划，对业务能力进行标准化、组件化。实际上，遵循业务架构设计方法，不断基于自身的实践进行积累和调整，任何企业都能发现适合自己的架构，包括适合自己的中台规划，之后再根据企业的业务规模和发展预期选择合适的技术栈。
- 企业级业务架构设计的真正威力还体现在对企业的整体认知、规划与改变上。深入开展企业级业务架构设计，足以将一个企业完整、深刻地联结在一起，这不是领域级设计可以解决的问题。从这个角度来讲，企业级业务架构设计堪称“中台之上”。相对自下而上的“生长”方式而言，企业级业务架构设计更适合于传统企业的数字化转型实践。

“巨大的成果只有经过巨大的努力方能获得”