

面向对象分析与设计

Object Oriented Analysis and Design

——需求分析

Requirement Analysis

邱明 博士

厦门大学信息学院

mingqiu@xmu.edu.cn

2023年秋季学期

目录

- 软件设计
- 编程范式
- 命令式编程
- 面向对象编程
- 函数式编程
- 统一建模语言UML



1. 系统开发的方法

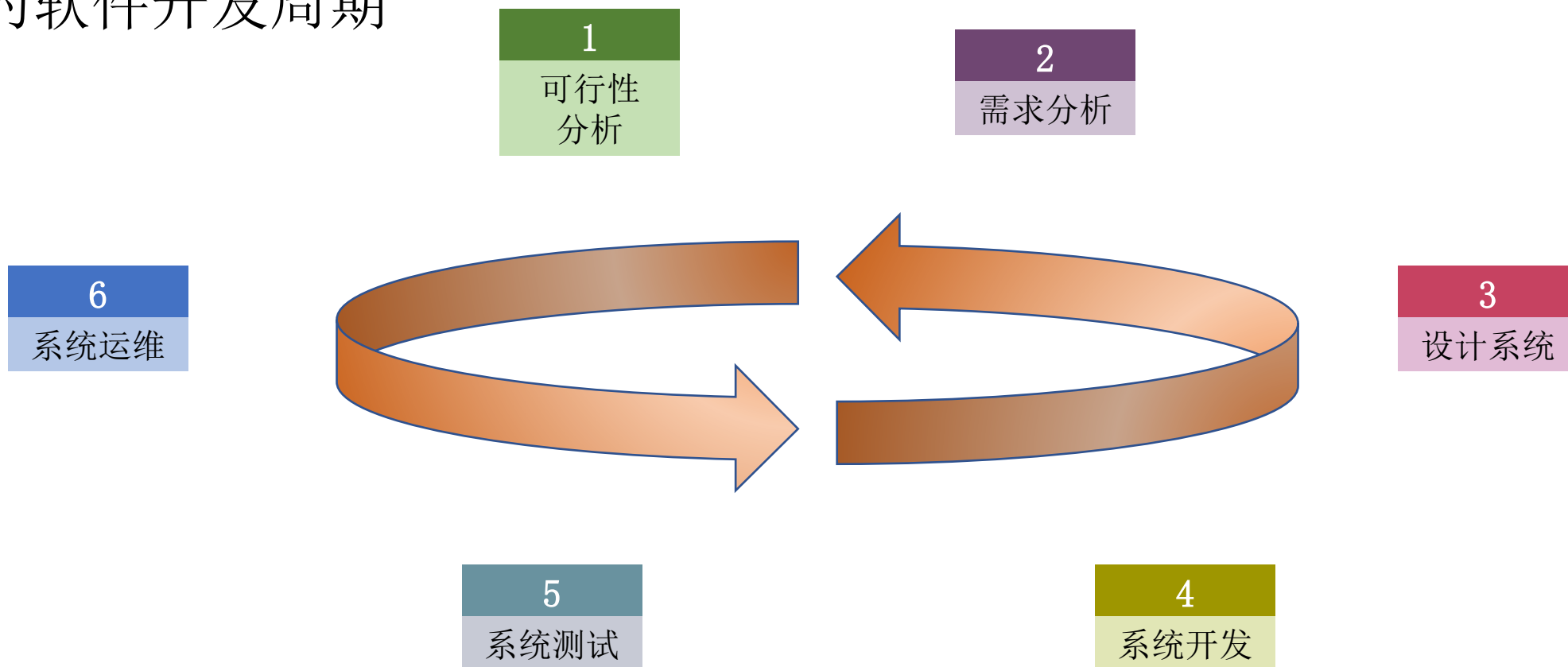
Development Methodologies



1.1 软件开发周期

Software Development Life Cycle (SDLC)

- 传统的软件开发周期



1.1 软件开发周期

Software Development Life Cycle (SDLC)

- 可行性分析——定位问题，找出机遇，确定目标
 - 分析现状，发现业务中存在的问题
 - 寻找信息技术所能带来的改变，抓住能确定竞争优势的机遇
 - 汇总信息，确定系统所需要解决的问题，划定系统的边界
 - 输出为包含问题定义和系统目标的可行性报告



1.1 软件开发周期

Software Development Life Cycle (SDLC)

- 需求分析
 - 确定用户信息需求
 - 分析现有业务（Who, What, Where, When, How）
 - 理解用户在执行业务时所依赖的信息
 - 分析与设计用户与系统的交互
 - 挖掘用户的需求
 - 分析系统需求
 - 用顺序图（Sequence Diagram）描述人机交互以及与集成系统的交互
 - 用活动图（Activity Diagram）描述业务的处理过程
 - 用数据字典描述系统中的数据项
 - 确定用户界面
 - 采用原型法（Prototype）设计界面，报表和交互信息



1.1 软件开发周期

Software Development Life Cycle (SDLC)

- 设计系统
 - 根据需求进行领域建模，用ER图描述数据库设计
 - 设计系统的体系结构，组件结构以及各组件间的接口
 - 在组件内部运用GRASP原则和设计模式，将组件的功能分解成职责分配给对象



1.1 软件开发周期

Software Development Life Cycle (SDLC)

- 系统开发
 - 根据设计分工合作用代码实现系统
 - 完成单元测试



1.1 软件开发周期

Software Development Life Cycle (SDLC)

- 系统测试
 - 集成测试和验收测试
 - 根据测试中的反馈，修改系统



1.1 软件开发周期

Software Development Life Cycle (SDLC)

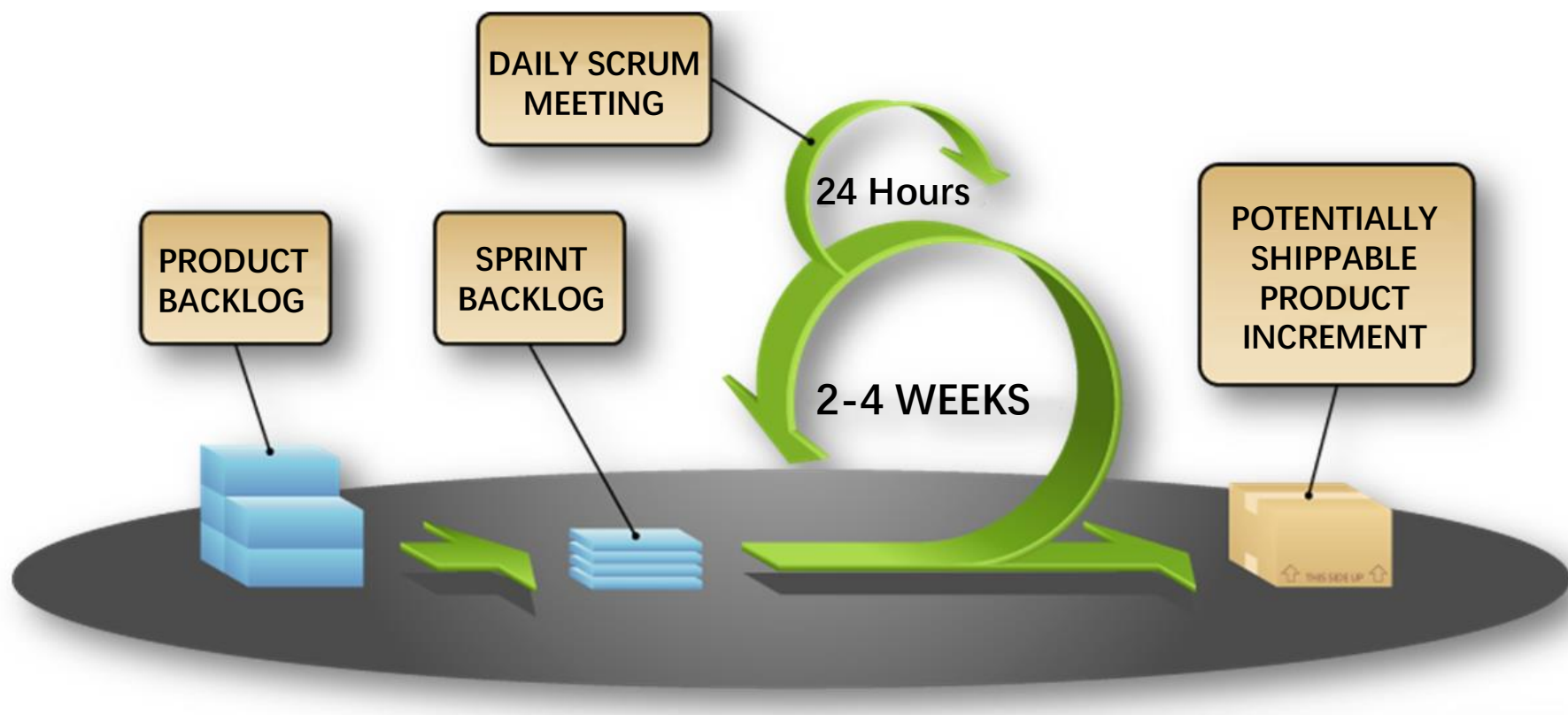
- 系统上线和运维
 - 验证系统是否满足需求
 - 部署系统，培训用户
 - 系统的运维



1.1 软件开发周期

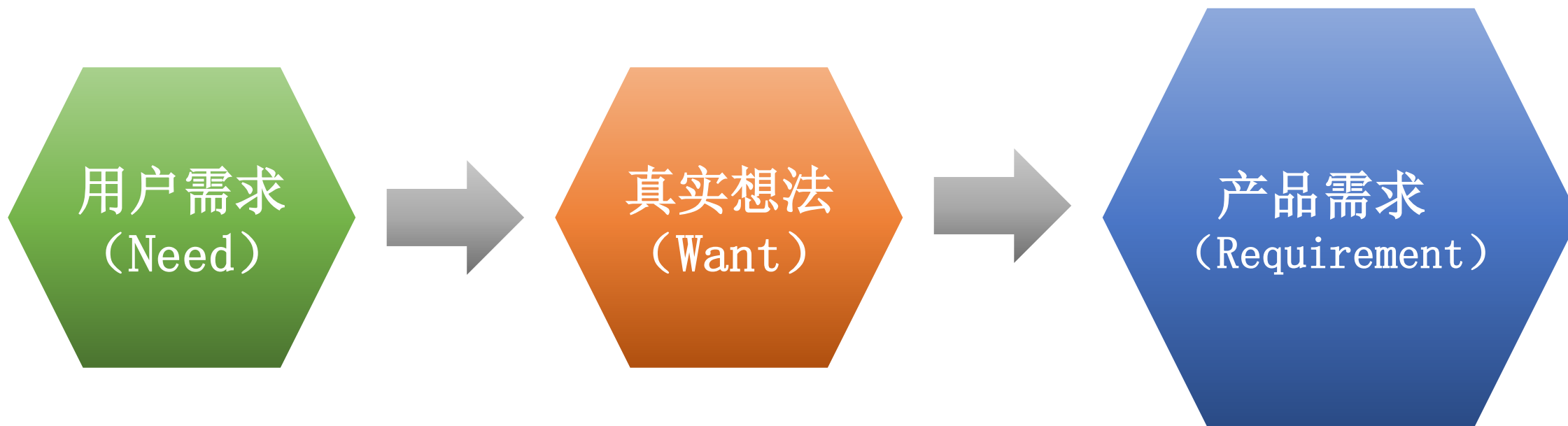
Software Development Life Cycle (SDLC)

- 敏捷的软件开发方法——Scrum



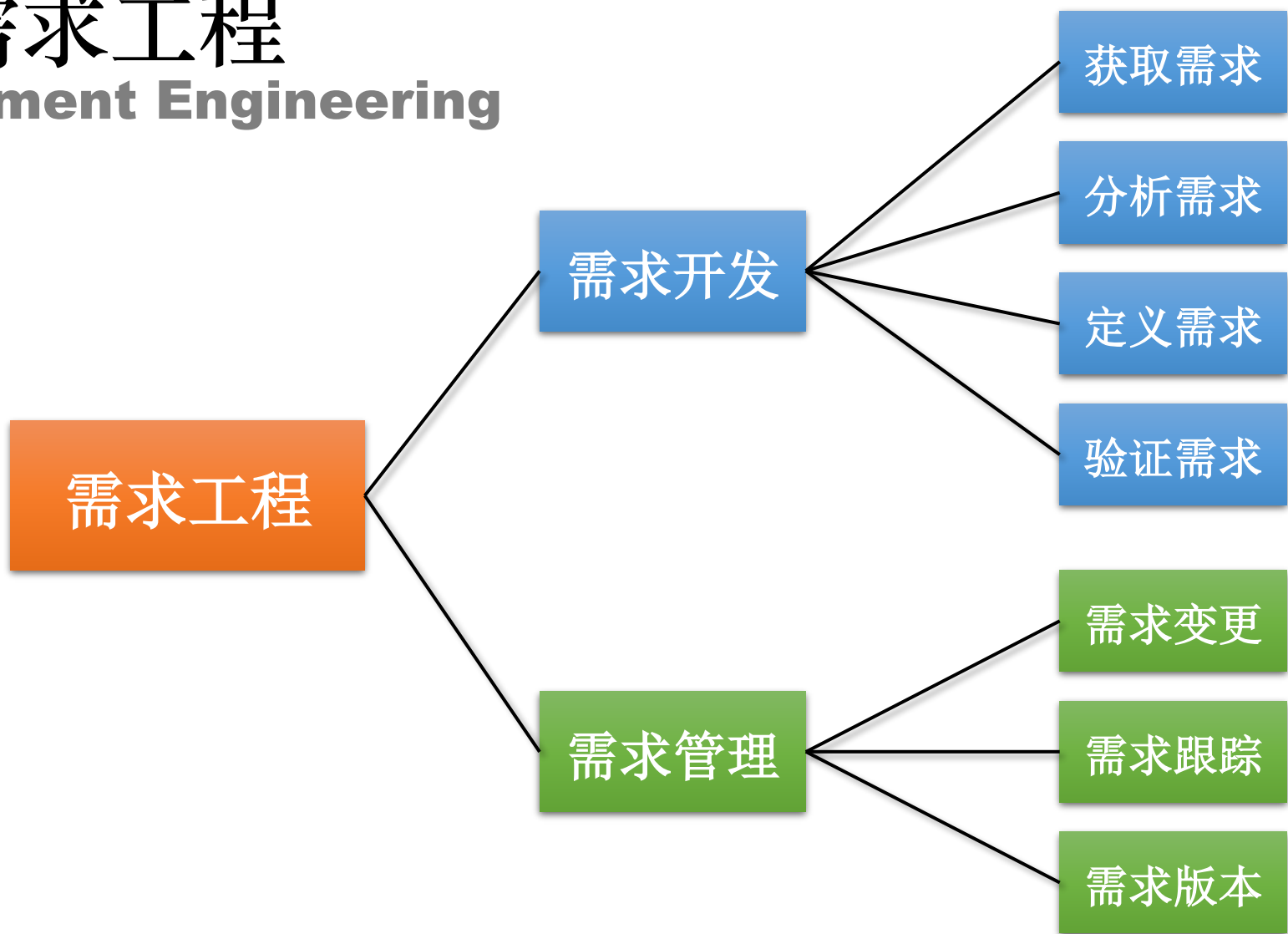
1.2 需求工程

Requirement Engineering



1.2 需求工程

Requirement Engineering



1.2 需求工程

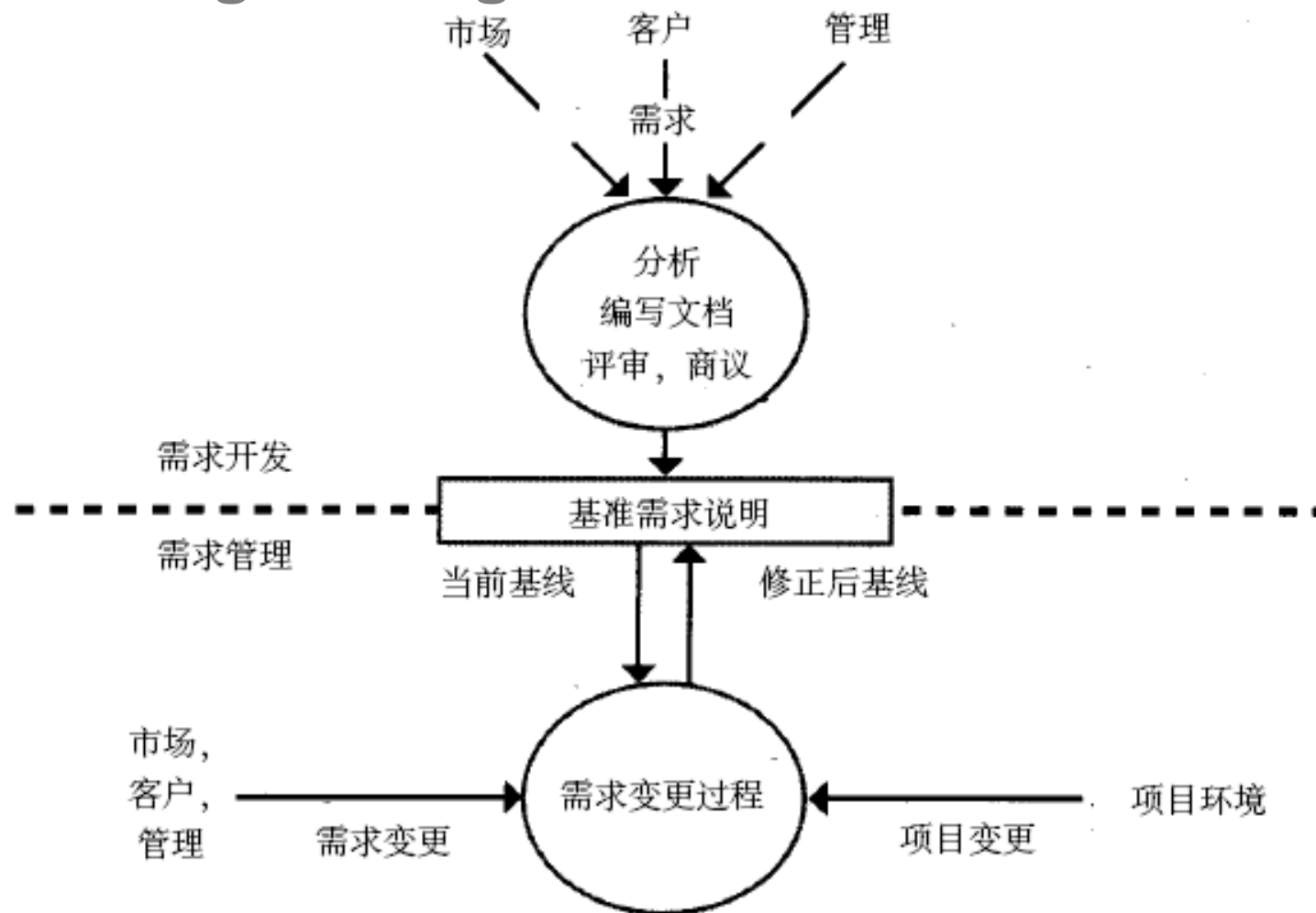
Requirement Engineering

- 需求开发结果—需求基线
 - 项目视图和范围文档、用例文档、软件需求规格说明及相关分析模型。
 - 经评审批准，这些文档就定义了开发工作的需求基线。
- 需求管理的目标
 - 建立软件需求基线，供软件工程和管理使用。
 - 软件计划、产品和活动同软件需求保持一致。



1.2 需求工程

Requirement Engineering



2. 需求获取

Requirement Gathering



2.1 FURPS+需求模型

FURPS+ Model

- 功能需求 (Functional)
 - 软件的功能，作用以及安全性要求
- 使用性需求 (Usability)
 - 人机交互，在线帮助，文档.
- 可靠性 (Reliability)
 - 错误率，可恢复性，可预计性.
- 性能 (Performance)
 - 响应时间，吞吐量，正确率，资源消耗.
- 支持 (Supportability)
 - 适配性，可维护性，国际化，可配置性
- 实现 (Implementation)
 - 资源限制，编程语言和工具，硬件，...
- 接口 (Interface)
 - 与外部系统的接口
- 法律 (Legal)
 - 软件的许可，软件的功能的合法性.



2.1 FURPS+需求模型

FURPS+ Model

- Use-Case 模型
 - 描述系统的使用场景 （功能需求）
- 辅助文件（Supplementary Specification ）
 - 非功能需求。
- 词汇表
 - 定义关键名词，数据词典。
- 业务规则
 - 描述业务规则的条款。



2.1 FURPS+需求模型

FURPS+ Model

- 需求的层次：
 - 目标性需求：是企业的高层管理人员所关注的整个系统需要达到的目标；
 - 功能性需求：是企业的中层管理人员所关注的整个系统必须完成的任务；
 - 操作性需求：是企业的具体操作人员所关注完成每个任务的具体的人机交互；



2.2 需求优先级划分

Determining Requirement Priority

- 从用户价值和公司价值这两个方面出发
 - 用户价值考虑频度、广度和强度。
 - 公司价值考虑用户增长，企业营收，用户粘性、产品护城河、用户体验和品牌价值等方面



2.2 需求优先级划分

Determining Requirement Priority

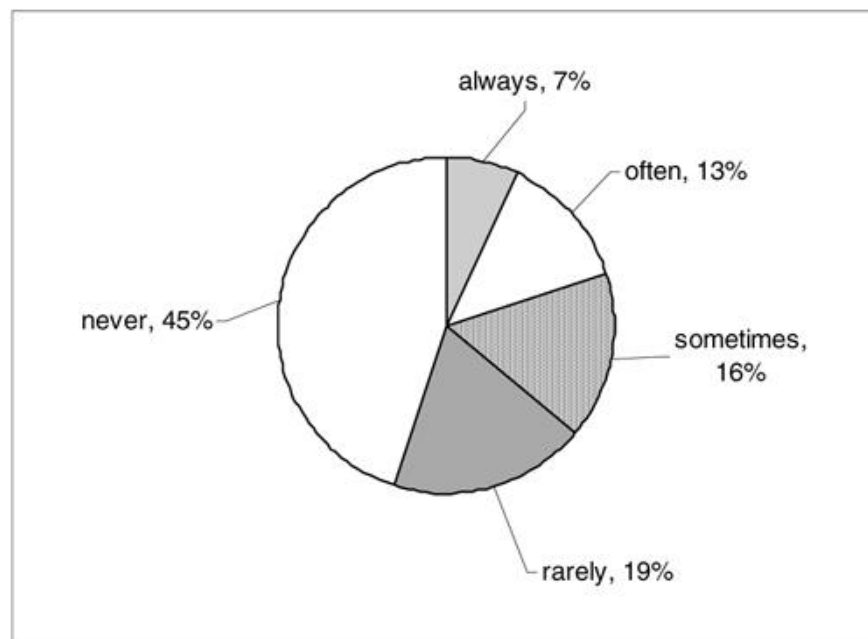
- 用户价值
 - 频度：用户多久会遇到这个问题一次；
 - 广度：有哪些用户会遇到这个问题。
 - 强度：利用KANO模型，通过不同需求对用户满意度的影响，把需求进行分类和优先级排序。



2.2 需求优先级划分

Determining Requirement Priority

- 系统功能的使用频率分析



2.2 需求优先级划分

Determining Requirement Priority

- KANO模型将需求分为：
 - 基本需求：用户使用你产品想要解决的核心需求，也是产品的必备功能，如果没有该功能，用户会极度不满，甚至放弃你的产品；即使有了该功能，用户也认为是理所应当的，用户不会因此对你的产品满意度增加，比如微信的聊天功能。
 - 期望需求：用户的痒点，产品满足了这类需求，用户满意度会因此增加，也可以成为自己产品的竞争优势；此类需求得不到满足或表现不好的话，客户的不满也会增加。



2.2 需求优先级划分

Determining Requirement Priority

- 兴奋需求：用户自己都没想到的需求，产品拥有此功能，即便表现的并不完善或完美，用户满意度也急剧提升；即便没有此功能，用户也并不会对产品对满意度降低。
- 无差异需求：用户感觉无所谓的需求，不管产品提供与否，用户的满意度都不会因此变化；虽然用户满意度不会变化，但是过多的无差异需求，浪费公司资源，使产品变的臃肿，因此也最好不要去做。
- 反向需求：该类需求提供对应的功能后，用户会对产品的满意度降低；



2.2 需求优先级划分

Determining Requirement Priority

- 公司价值分为：
 - 用户增长：这里的用户增长指的是净用户增长，
 - 净用户=新增用户+召回用户-流失用户。
 - 为用户生成个性化视频的功能，视频的炫酷效果可能带来用户的自发传播
 - 创造营收；
 - 增加信息流广告，
 - 优化商品详情页以提高订单转化率
 - 提高用户粘性，次日/七日留存，APP打开率，平均使用时长
 - 高频、刚需的需求是提升用户粘性的主要手段



2.2 需求优先级划分

Determining Requirement Priority

- 构建产品护城河，在满足用户基本需求后，做一些与自身产品内核相关的内容， 增加用户的更换成本；
 - 比如一款新的社交APP中的微信通讯录中的好友
- 改善用户体验，
 - 如修复Bug、改善产品外观、提高服务等；
 - 如优化流程、提高产品性能。
- 增加品牌价值，提高产品知名度，改善产品口碑，增加产品地位
 - 如美图秀秀在儿童节推出了秒变小学生的功能，就能很好的提高产品知名度。



2.2 需求优先级划分

Determining Requirement Priority

- 需求优先级公式：
 - 需求优先级=需求价值/成本
 - 需求价值=用户价值*（公司价值+战略契合度+老板意志）
 - 成本=开发周期+技术风险+政策风险



2.3 获取需求的方法

Requirement Gathering

- 获取需求的方法
 - 用户访谈 (Interview)
 - 问卷调查 (Questionnaires)
 - 实地观察 (Field Survey)



2.3 获取需求的方法

Requirement Gathering

- 用户访谈
 - 访谈为分析人员提供了与访谈对象自由沟通的机会；
 - 通过访谈可以挖掘更深层次的用户需求；
 - 访谈允许分析人员使用一些个性化的问题；
 - 成功的访谈在很大程度上取决于分析人员的经验与技巧



2.3 获取需求的方法

Requirement Gathering

- 访谈前的准备
 - 阅读背景材料
 - 企业组织结构图、公司战略规划、各部门的正式目标和职责、工艺流程和产品构成、政策手册、操作过程、工作指令、各种表格等
 - 确定访谈的目标
 - 管理方式和具体业务的管理方法、业务流程和工作形式、数据和数据流程、决策过程和决策方式、现存问题和改进意见
 - 决定访谈对象
 - 访谈前的准备
 - 可将访谈的提纲预先发给访谈对象



2.3 获取需求的方法

Requirement Gathering

- 访谈的问题类型 (Question Type)
 - 开放性问题
 - 您对当前公司的业务状态有什么看法?
 - 贵部门的关键目标是什么?
 - 当数据提交后, 如何处理?
 - 封闭性问题
 - 每月平均业务量有多少?
 - 报表是每月出一次吗?



2.3 获取需求的方法

Requirement Gathering

开放问题

封闭问题

低

数据可靠性

高

低

时间效率

高

低

数据精确性

高

大

宽度和广度

小

大

访谈技巧要求

小

难

分析难度

易



2.3 获取需求的方法

Requirement Gathering

- 访谈的问题组织 (Arranging Questions)
 - 金字塔结构： 从封闭性问题到开放性问题
 - 您觉得贵公司的防火墙存在着哪些问题？
 - 您考虑过其他提升数据安全性的方法吗？
 - 您认为哪些方法可以包含公司的数据安全？
 - 您觉得如何权衡数据的安全性和互联网的访问便利性？
 - 漏斗结构： 从开放性问题到封闭性问题
 - 您对电商系统如何看？
 - 贵公司哪个部门涉及了电子商务系统？
 - 贵公司哪些商品是通过网络销售的？
 - 哪些商品不能通过网络销售？



2.3 获取需求的方法

Requirement Gathering

- 问卷调查
 - 自由格式
 - 为回答者提供了非常灵活的回答问题的方式。例如，“每天收到哪些报表和数据，如何使用或处理这些数据和报表？”；“这些数据是否适用？数据是否及时、准确？格式是否合理？”等等。
 - 固定格式
 - 需要事先设定选项或几种答案供用户选择。
 - 节省成本，短时间可获取大量反馈
 - 问卷不够灵活（内容局限）、信息质量难于保证。



2.3 获取需求的方法

Requirement Gathering

- 实地观察法
 - 直接、直观、带有一定的实验性。
 - 获取第一手数据、有助于弄清复杂流程（包括异常场景）
 - 获取多方面数据，可以证实收集的资料正确与否，更正不正确的概念，澄清模糊的概念。
 - 必须懂得业务、跟随进行实际工作、较花费时间。



3.用例建模

Use Case Modeling



3.1 用例模型

Use Case Model

- 用例模型包含用例图和用例两部分，描述系统的使用场景（功能需求）
 - 用例使用文字来描述的，一个好的用例应该具有良好可读性。通常由多个句子构成，按照一定的格式进行组织。每个句子描述一个简单的操作步骤。
 - 用例图用于描述用例和参与者以及它们之间的关系



3.1用例模型

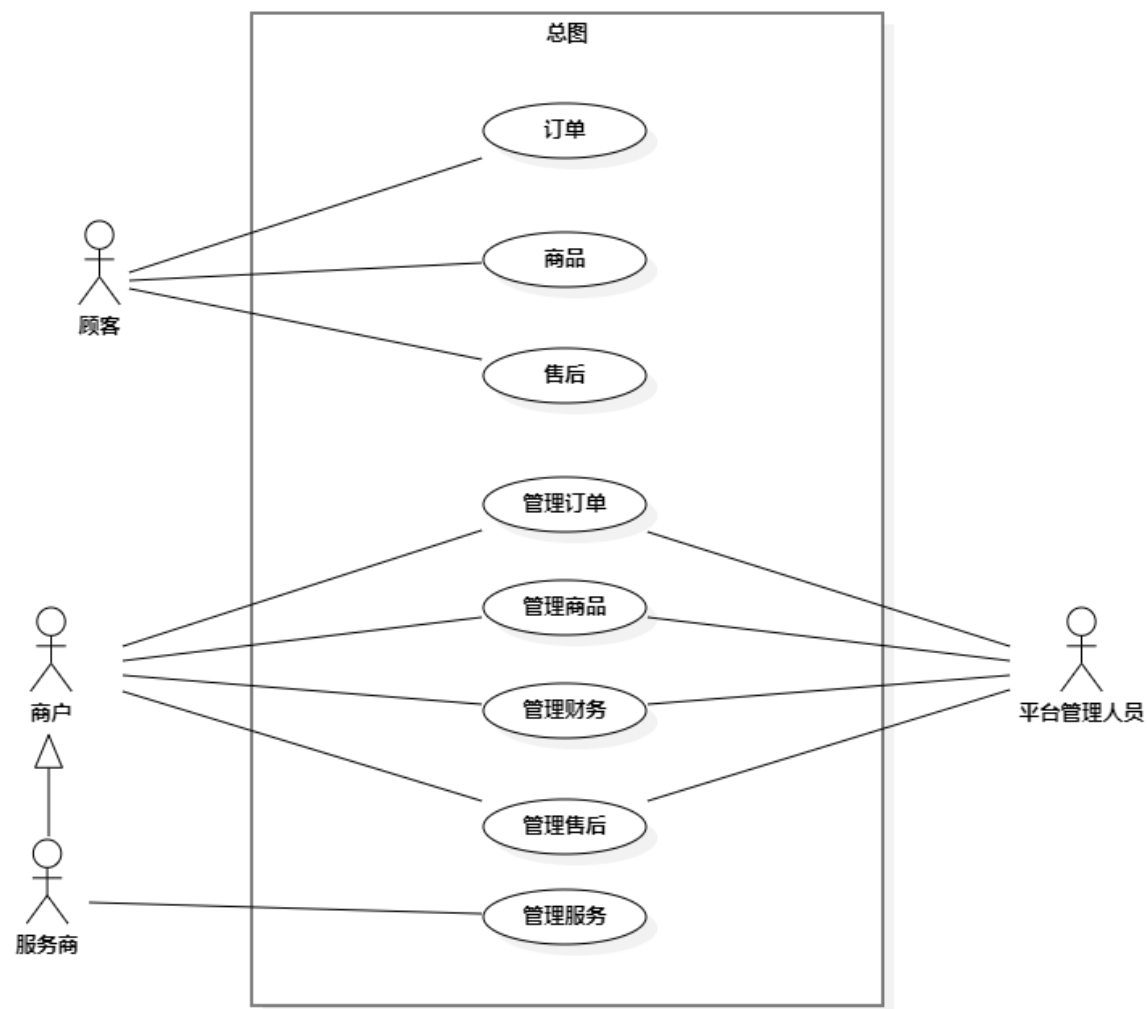
Use Case Model

用例编号: MALL-004		用例名称: 购买商品
级别: 用户目标		
包: 商品		
参与者: 顾客		
描述: 顾客购买商品,		
触发事件: 顾客购买商品		
主成功场景		信息
1. 顾客购买商品。		商品id, 数量
2. 系统计算总价、运费, 推荐的优惠卷, 选择默认的开票方式。		总价, 运费, 推荐的优惠卷, 默认开票方式
3. 顾客提交订单		
4. 系统进入支付子用例 (MALL-PAY-001)		
扩展场景		信息
2a. 商品售罄		
1. 系统显示商品售罄, 终止后续步骤		
3a. 顾客选择其他优惠卷		
1. 系统显示所有适用的优惠卷		适用优惠卷
2. 顾客选择其中一张优惠卷		优惠卷
3. 返回第2步重新计算总价		总价
3b. 顾客选择其他开票方式		
1. 系统显示顾客的所有开票方式		所有开票方式
2. 顾客选择其中一种开票方式		开票方式
3. 返回第2步重新显示新选的开票方式		
4a. 商品售罄		
1. 系统显示商品售罄, 终止后续步骤		
其他: 若是第三方商户的交易, 需通过分账模式完成		



3.1 用例模型

Use Case Model



电子商城用例图



3.1 用例模型

Use Case Model

- 用例模型将其他需求项的连接在一起



3.2 用例的构成

Use Case Symbols

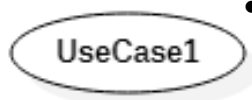
- 参与者 (Actor)：与系统产生交互的人、系统或者组织



- 主参与者 (Primary actor) - 需要通过使用系统来达成某些目标
- 辅助参与者 (Supporting actor) - 为系统提供服务
- 幕后参与者 (Offstage actor) - 在系统中有相关利益，但是不直接参与系统的使用

- 场景 (Scenario)：参与者与系统之间产生的一系列动作和交互，是使用系统的一种操作顺序

- 用例 (Use Case)：



- 是以用户的角度，描述系统完成客户目标的动作序列，这一序列对于特定参与者产生了一个有价值的可见结果。



3.2 用例的构成

Use Case Symbols

- 场景与用例（Scenarios and Use Cases）

退货主场景：一位顾客带着要退货的商品和收据来到收银台。收银员使用POS系统查询收据。收银员检查核对每件退回的物品。收银员记录退货的商品。收银员将退货的钱款从原支付渠道退回。

其他场景：如果顾客的商品是捆绑销售商品，需要将捆绑的商品一并退货。

其他场景：如果客户支付时使用了优惠券，退款同时给用户退回优惠券。

其他场景：如果客户开具了发票，需要将原发票退回，再重新开具发票。



3.2 用例的构成

Use Case Symbols

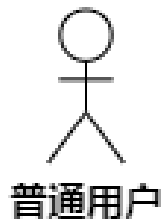
- 参与者 (Actor)



主参与者：收银员



辅助参与者：信用卡交易系统，税务开票系统



浏览商品



3.3 定义用例

Defining Use Cases

- Step 1: 选择系统的边界
 - 首先确定系统中的所有参与者，这会有助于厘清系统的边界
- Step 2: 寻找其中的主参与者
- Step 3: 确定主参与者的目标
- Step 4: 定义用例
 - 通常一个目标可能是一个用例，每个用例都会创造价值



3.3 定义用例

Defining Use Cases

- 电子商城系统中的边界
 - 不包含支付内容，依托第三方支付平台完成支付（中国人民银行规定）
 - 不包括物流的内容
 - 不包括仓库管理的内容



3.3 定义用例

Defining Use Case

- 电子商城系统中的主参与者
 - 顾客
 - 销售商品的商户
 - 平台管理人员
 - 提供售前和售后服务商
 - 快递员？
 - 仓库管理员？



3.3 定义用例

Defining Use Cases

- 电子商城系统中的辅助参与者
 - 仓库系统
 - 第三方支付平台
 - 物流系统
 - 税务系统



3.3 定义用例

Defining Use Cases

- 电子商城系统中的幕后参与者
 - 中国人民银行
 - 国家税务总局
 - 国家市场监督管理总局



3.3 定义用例

Defining Use Cases

- 主参与者的目标

主参与者	目标	主参与者	目标
顾客	获得商品信息	商户	新增商品
	查找商品		上架商品销售
	按照分类查看商品		下架商品
	购买商品		增加团购活动
	查看订单信息		中止团购活动
	取消订单		接收订单
	提出售后申请		订单发货



3.3 定义用例



Defining Use Cases

- 电子商城系统中的边界
 - 不包含支付内容，依托第三方支付平台完成支付（中国人民银行规定）
 - 不包括物流的内容
 - 不包括仓库管理的内容



3.3 定义用例

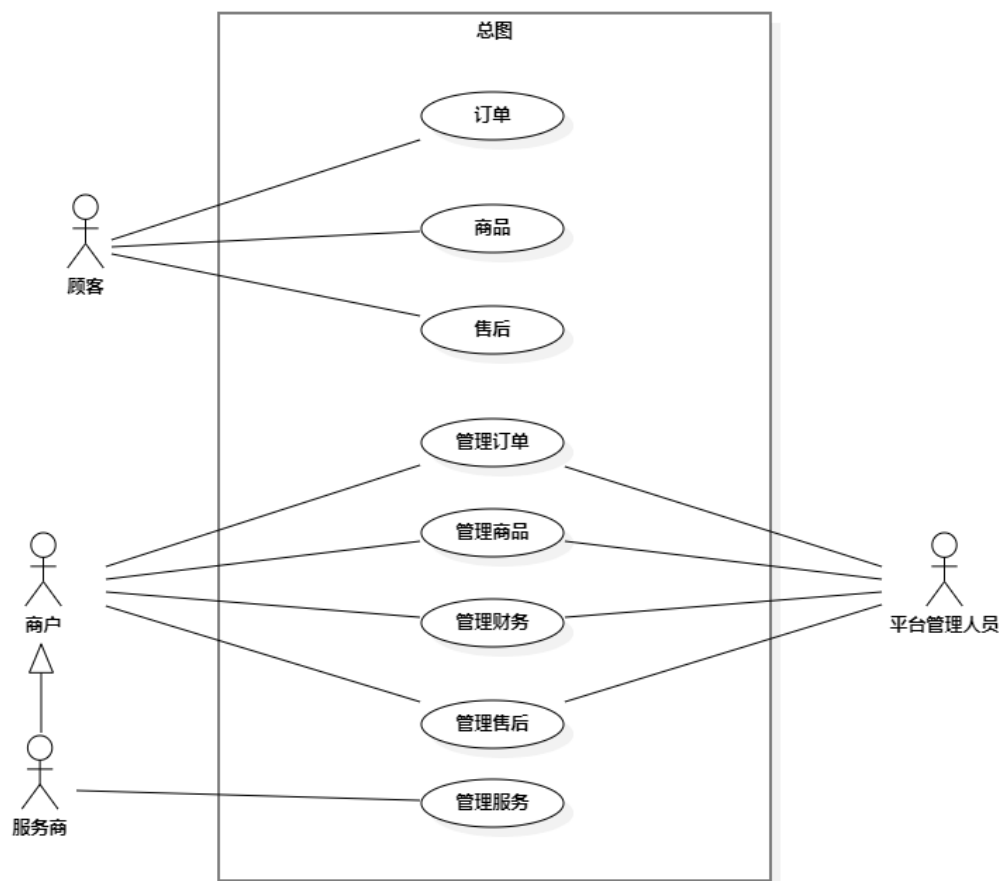
Defining Use Cases

- 用例的级别
 - 用户级别 (User-Goals)
 - 表述主参与者的意图
 - 需要满足EBP原则
 - 标记为蓝色 (海平面) 
 - 汇总级别 (Summary Level)
 - 包含多个用户级别的用户
 - 主要用于描述用户目标的上下文
 - 也可以用于描述多个用例的执行顺序
 - 无需要满足EBP原则
 - 标记为白色 (海平面上)
 - 子功能级别 (Subfunctions)
 - 通常是多个用例中共同的部分
 - 标记为深蓝色 (海平面以下) 



3.3 定义用例

Defining Use Case

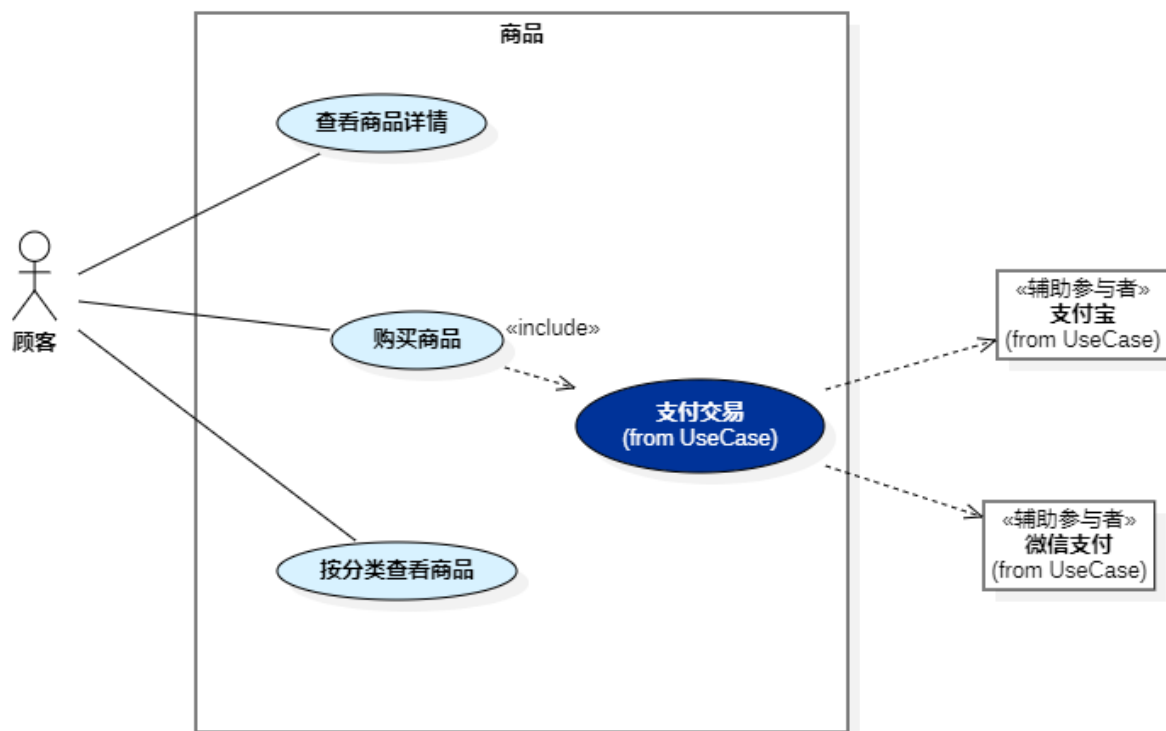


电子商城用例总图



3.3 定义用例

Defining Use Case



商品用例图



3.3 定义用例

Defining Use Case

- 有效用例的标准
 - 老板测试：每一个用例都应该有明确的目标，在完成后应该有可量化的价值
 - EBP（Elementary Business Process）测试：一个人于某个时间某个地点，为响应业务事件所执行的任务，如果能增加可量化的业务价值，并且以持久状态保留下数据，则可以通过EBP测试
 - 规模测试：一个用例必须达到一定的规模才能算有效。必须达到什么样的规模呢？通常必须包含多个步骤，在详述形式的用例说明通常会持续数页。不能通过规模测试的一个典型错误，就是将一系统步骤中的一个作为用例。



3.4 用例格式

Defining Use Case

- 简洁格式
 - 用一段话描述用例，主要描述其中的主成功场景
- 普通格式
 - 分为多段，每一段描述一个场景
- 用例规约
 - 遵循严谨的格式规范



3.4 用例格式

Defining Use Case

- 用例规约
 - 遵循严谨的格式规范

用例编号:	用例名称:
包:	
参与者:	
描述:	
触发事件:	
主成功场景	
扩展场景	
其他:	



3.4 用例格式

Defining Use Case



用例编号：MALL-004		用例名称：购买商品
级别：用户目标		
包：商品		
参与者：顾客		
描述：顾客购买商品，		
触发事件：顾客购买商品		
主成功场景		信息
1. 顾客购买商品。		商品id，数量
2. 系统计算总价、配送地址、运费，推荐的优惠卷，选择默认的开票方式。		总价，配送地址、运费，推荐的优惠卷，默认开票方式
3. 顾客提交订单		
4. 系统进入支付子用例（MALL-PAY-001）		
扩展场景		信息
2a. 商品售罄		
1. 系统显示商品售罄，终止后续步骤		
3a. 顾客选择其他优惠卷		
1. 系统显示所有适用的优惠卷		适用优惠卷
2. 顾客选择其中一张优惠卷		选择的优惠卷
3. 返回第2步重新计算总价		总价
3b. 顾客选择其他开票方式		
1. 系统显示顾客的所有开票方式		所有开票方式
2. 顾客选择其中一种开票方式		选择的开票方式
3. 返回第2步重新显示新选的开票方式		
3c. 顾客选择其他配送地址		
1. 系统显示顾客的其他配送地址		所有配送地址
2. 顾客选择其中一个配送地址		选择的配送地址
3. 返回第2步重新显示新的配送地址		
4a. 商品售罄		
1. 系统显示商品售罄，终止后续步骤		
其他：若是第三方商户的交易，需通过分账模式完成；支付失败需要重新改回可销售数目		



3.4 用例格式

Defining Use Case

用例编号：MALL-PAY-001	用例名称：支付交易
级别：子功能	
包： 交易	
参与者：顾客	
描述： 顾客通过网络支付，	
触发事件： 界面操作	
主成功场景：	信息
1. 系统显示支付的金额以及支持的支付渠道。	支付金额，支付渠道
2. 顾客选择支付渠道，提交。	
3. 系统调用选择的支付渠道处理	实际支付金额（可能有支付渠道优惠）
扩展场景：	信息
3a. 系统调用支付系统失败	
1. 返回第1步重新选择支付渠道	
其他：	



3.4 用例格式

Defining Use Case

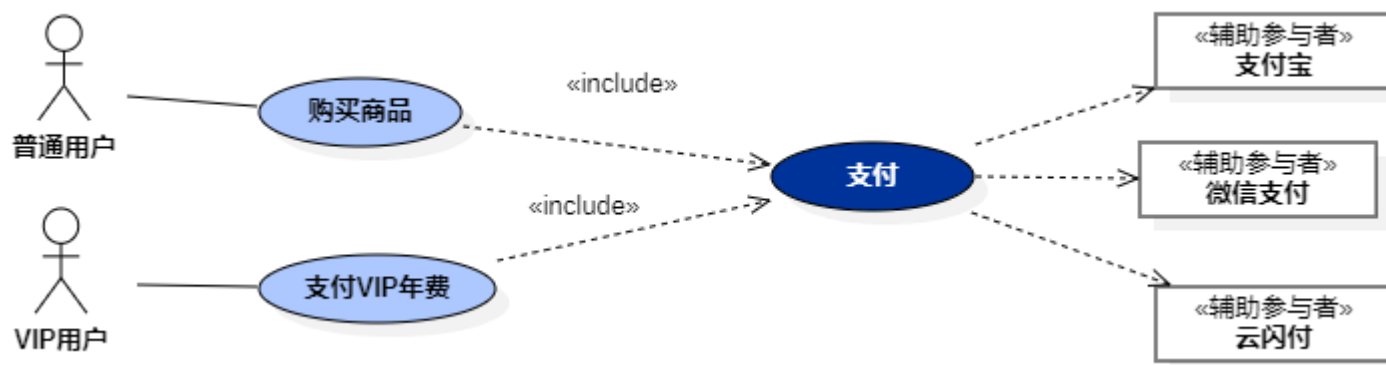
- 用例是短文。
- 用例里不要有系统设计、界面设计、合法性检查、特性列表、测试等。
- 用例描述行为需求和信息需求。
- 用例不描述系统内部如何实现。
- 用例的主场景不要超过九步。可以在适当的层次上得到子目标和移除设计说明。
- 用例的最大价值不在于主场景，而是在于备选行为。主场景可能只占用例长度的四分之一到十分之一。



3.5 用例的关系

Linking Use Case

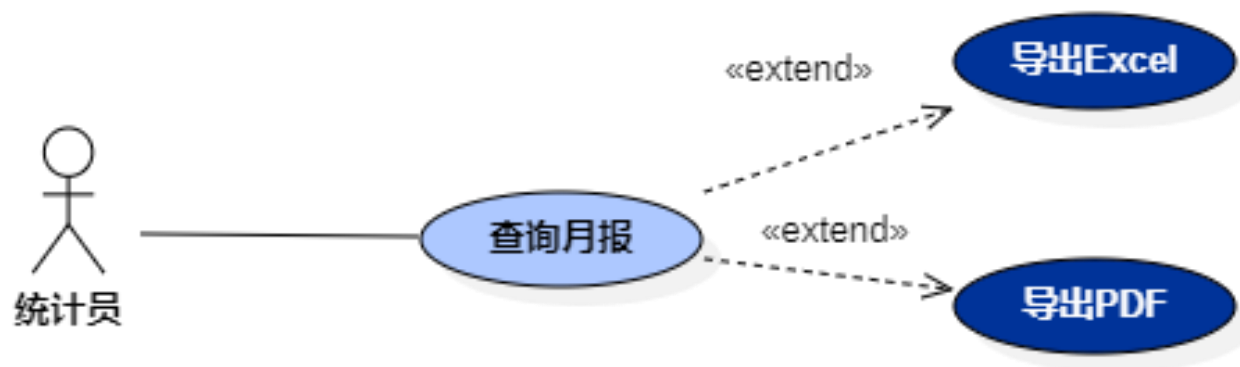
- 包含关系（Include）：将共性的内容放在在一个子用例中，共性内容会在多个用例中多次出现。



3.5 用例的关系

Linking Use Case

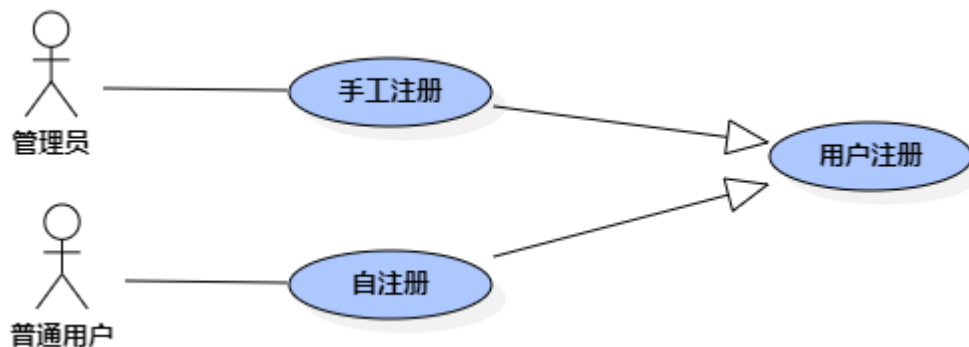
- 扩展关系（Extend）：表示一个用例扩展另一个用例的内容。



3.5 用例的关系

Linking Use Case

- 泛化关系（Generalizes）：当多个用例共同拥有一种类似的结构和行为时，可以将它们的共性抽象成为父用例，其他的用例作为泛化关系的子用例。在用例的泛化关系中，子用例是父用例的一种特殊形式，它继承了父用例的所有结构、行为、关系。
 - 泛化关系也可以描述参与者之间的关系



4. 系统交互与流程建模

System Interaction Analysis and Business Process Modeling



4.1 系统交互

System Interaction

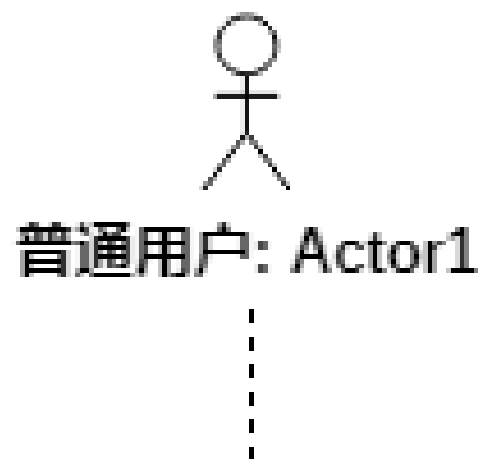
- 在特定的用例场景下，参与者、系统、以及外部集成之间存在着信息的流动。
- 系统交互分析把系统都看作黑盒，不关心其内部实现，只关心从参与者到系统，从系统到集成系统之间那些跨越系统边界的信息流动。



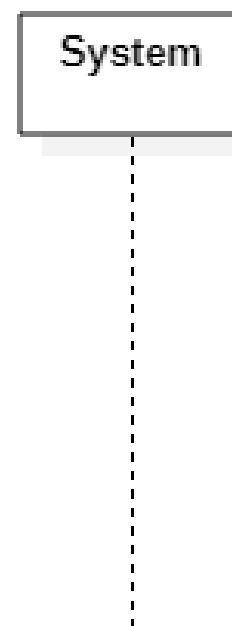
4.2 系统顺序图的符号

System Sequence Diagram Symbols

参与者：与用例相同

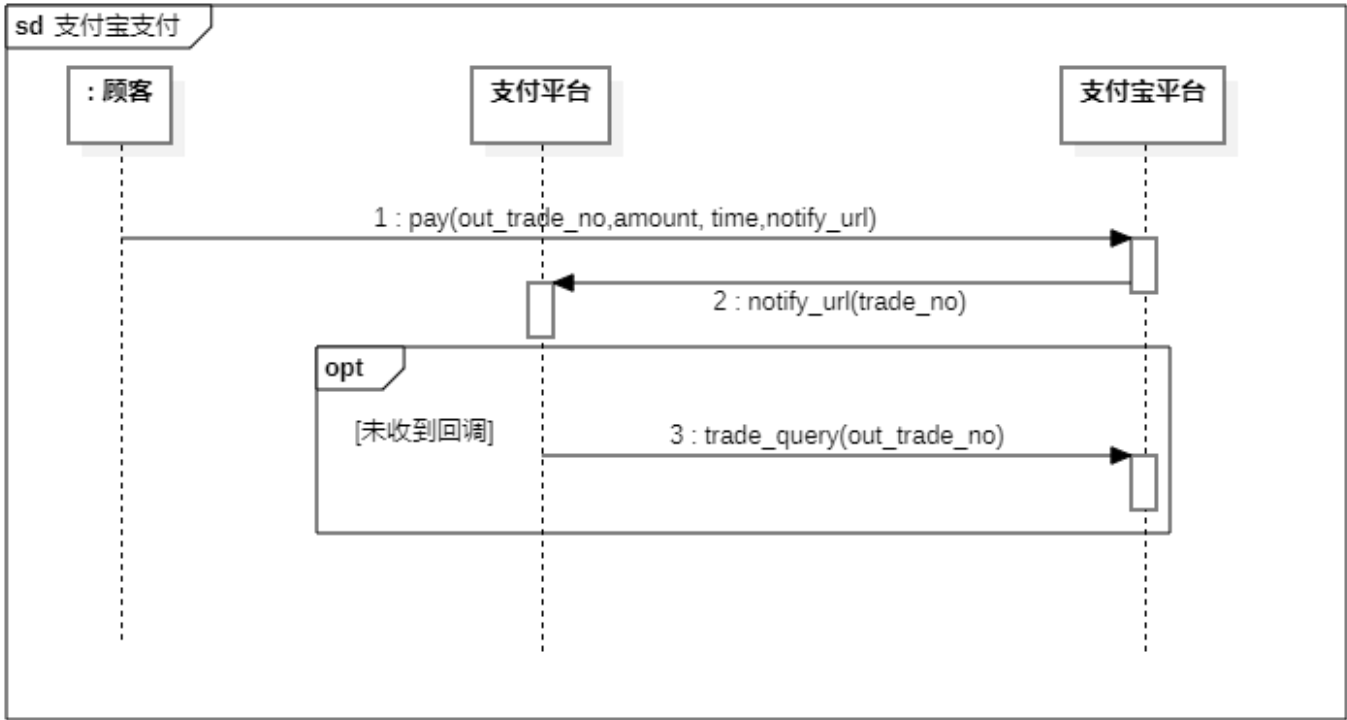


系统：分析的目标系统
(System Under Design)



4.3 系统顺序图范例

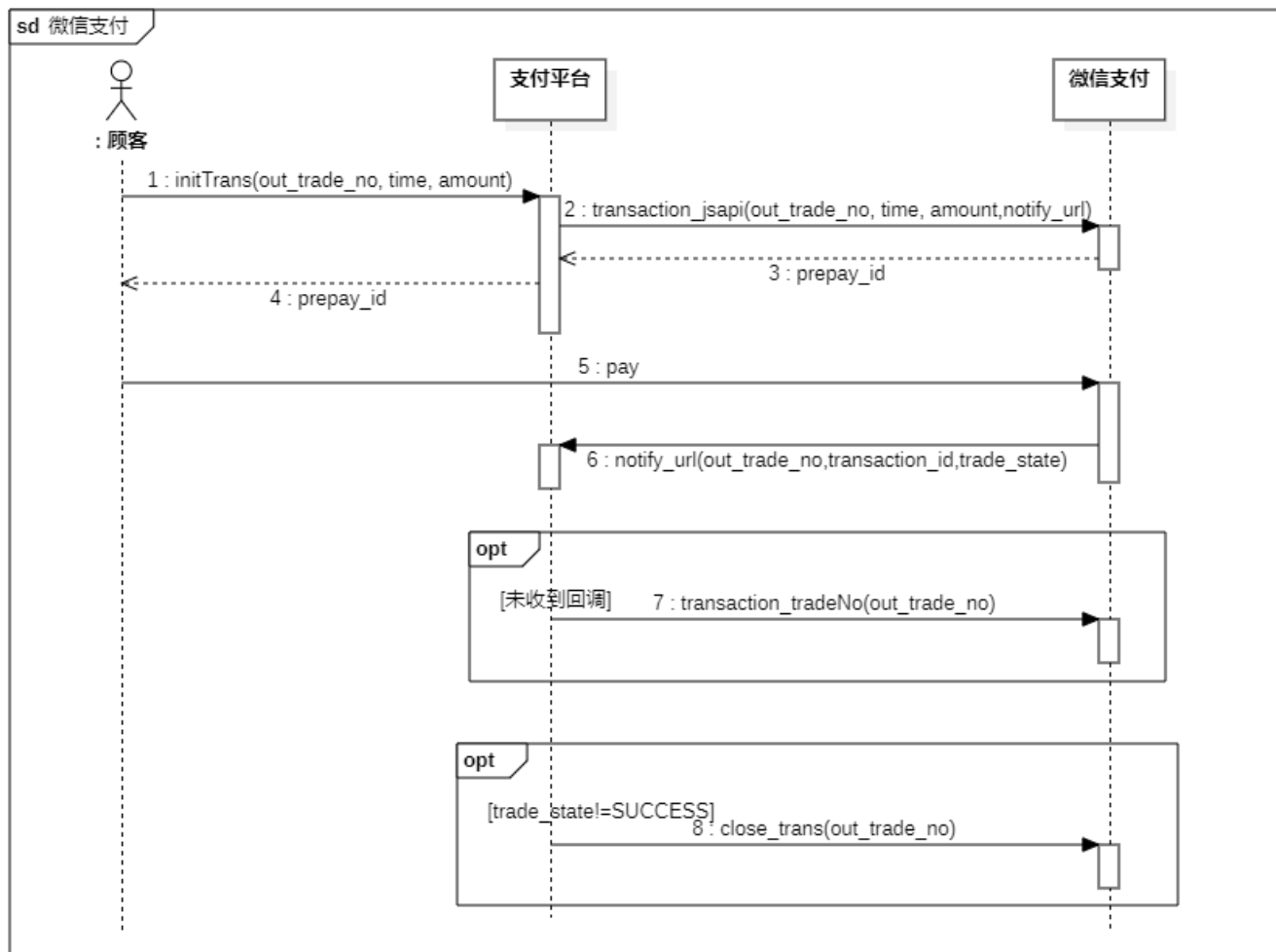
Examples of System Sequence Diagram



图中方法名	调用说明URL	功能
pay	https://opendocs.alipay.com/open/028xr6	下单
notify_url	为 下 单 时 传 过 去 的 notify_url, 说 明 见 https://opendocs.alipay.com/open/270/105902?ref=api	回调
trade query	https://opendocs.alipay.com/open/02irgs , 以out trade no查询	查单

4.3 系统顺序图范例

Examples of System Sequence Diagram



4.3 系统顺序图范例

Examples of System Sequence Diagram

图中方法名	调用说明URL	功能
transaction_jsapi	https://pay.weixin.qq.com/wiki/doc/apiv3_partner/apis/chapter4_1_1.shtml	下单
notify_url	为 下 单 时 传 过 去 的 notify_url, https://pay.weixin.qq.com/wiki/doc/apiv3_partner/apis/chapter4_1_5.shtml	回调
transaction_tradeNo	https://pay.weixin.qq.com/wiki/doc/apiv3_partner/apis/chapter4_1_2.shtml , 以 out_trade_no查询	查询订单
close_trans	https://pay.weixin.qq.com/wiki/doc/apiv3_partner/apis/chapter4_1_3.shtml	关闭订单

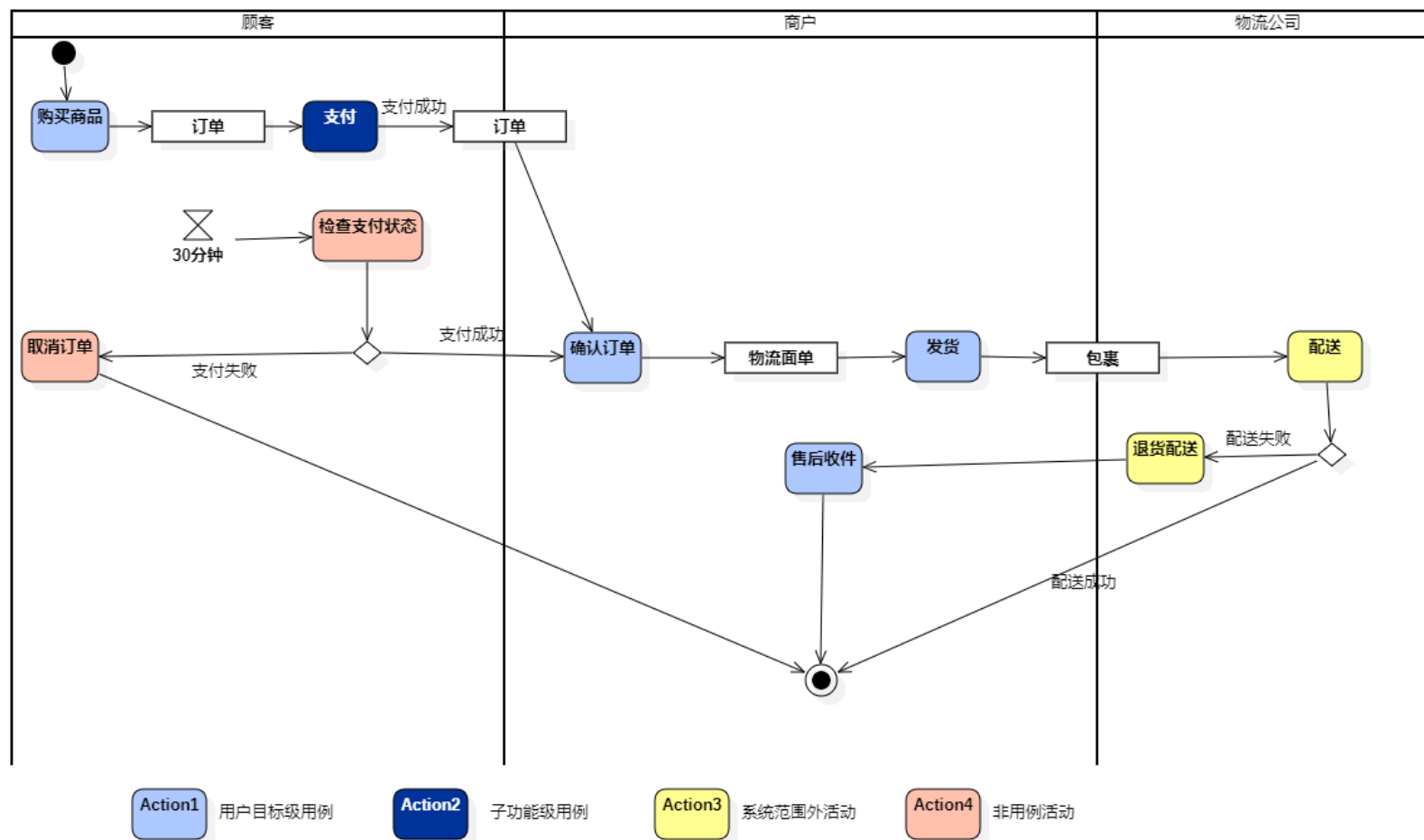
4.4 业务流程

Business Process

- 业务流程是一组活动，这组活动有一个或多个输入，输出一个或多个结果，这些结果对客户来说是一种增值。



4.4 业务流程 Business Process



5. 支付模块案例

The Example of Payment Module



5.1 支付模块的需求

Payment Module Requirement

- 连接消费者、商家（或平台）和金融机构的桥梁，管理支付数据，调用第三方支付平台接口，记录支付信息（对应订单号，支付金额等），金额对账等功能
- 支付平台应该具备产品的通用性，接入主流的第三方支付（微信支付，支付宝支付），屏蔽不同支付平台的差异，
- 支持多阶段支付能力（定金支付），多种类退款（全部退款，部分退款）



5.1 支付模块的需求

Payment Module Requirement

- 主参与者及其目标
 - 顾客：通过平台支付和退款
 - 平台管理人员：负责分账、对账、检查每日流水，管理支付渠道
 - 财务总监：处理挂账、
 - 商户：登记自己的账户，查看流水



5.1 支付模块的需求

Payment Module Requirement

- 辅助参与者
 - 支付宝
 - 微信支付
 - 其他第三方支付



5.1 支付模块的需求

Payment Module Requirement

- 潜在参与者 – 中国人民银行
 - 中国人民银行办公厅. 关于进一步加强无证经验支付业务整治工作的通知. 银办发[2017]217号文
 - 中国人民银行. 关于规范支付创新业务通知. 银办发[2017]281号文
 - 中国人民银行. 关于印发〈条码支付业务规范（试行）〉的通知. 银办发[2017]296号文



5.1 支付模块的需求

Payment Module Requirement

中国人民银行办公厅文件

银办发〔2017〕217号

中国人民银行办公厅关于进一步加强 无证经营支付业务整治工作的通知

中国人民银行上海总部，各分行、营业管理部，各省会（首府）城市中心支行，各副省级城市中心支行：

为贯彻落实第五次全国金融工作会议精神，保障党的决策部署在支付结算领域全面执行，主动防范系统性支付风险，严肃支付结算纪律，落实支付服务市场主体责任，强化监管问责，促进支付服务良性循环和市场健康有序发展，根据《互联网金融风险专项整治工作实施方案》（国办发〔2016〕21号文印发）和《非

— 1 —



中华人民共和国中央人民政府

www.gov.cn



首页 | 繁体 | 英文EN | 登录 | 邮箱

首页 > 国务院公报 > 2018年第17号

字号：默认 大 超大 | 打印 | 分享 | 收藏

人民银行关于规范支付创新业务的通知

银发〔2017〕281号

中国人民银行上海总部，各分行、营业管理部，各省会（首府）城市中心支行，各副省级城市中心支行；各国有商业银行、股份制商业银行，中国邮政储蓄银行；各非银行支付机构；中国银联股份有限公司、中国支付清算协会、城市商业银行资金清算中心、农信银资金清算中心、网联清算有限公司：

近年来，我国支付业务创新不断发展，支付服务环境日趋完善，对提高支付效率、便利社会生产生活发挥了积极作用。为进一步加强支付业务管理，促进支付创新，推动支付服务市场持续健康发展，现就有关事项通知如下：

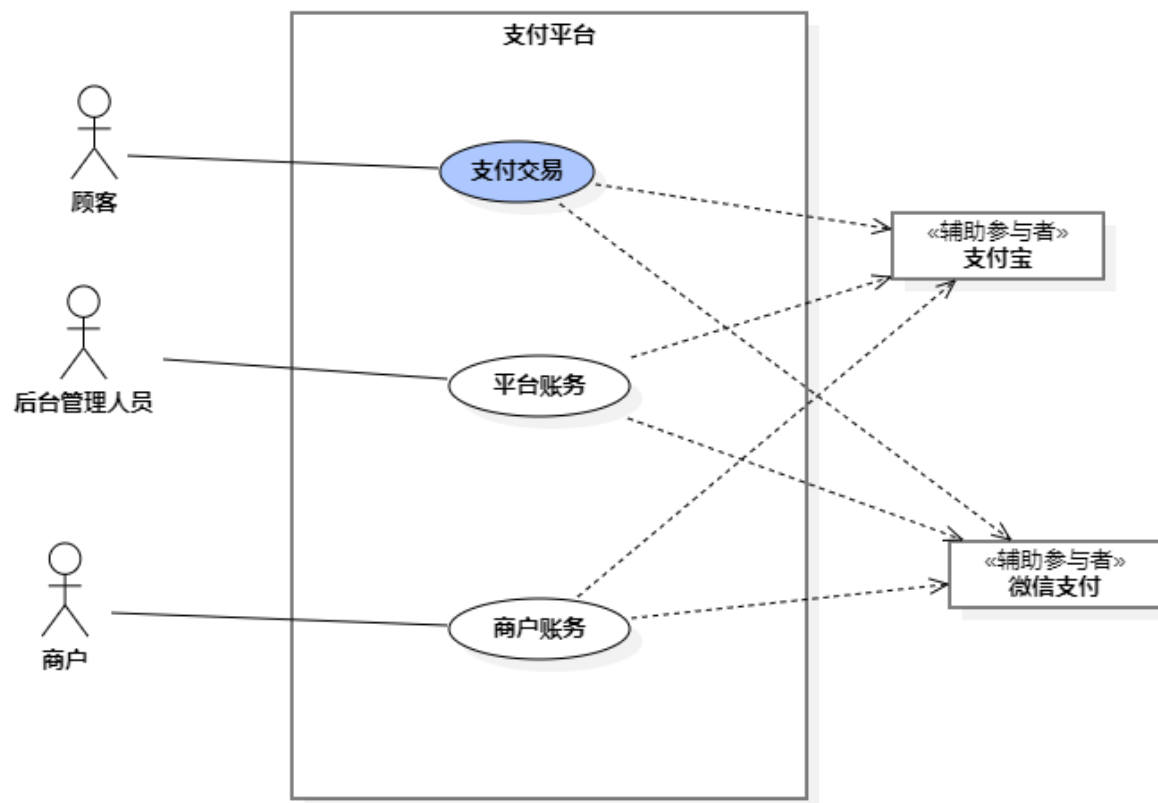
一、开展支付创新业务事前报告

各银行业金融机构（以下简称银行）、非银行支付机构（以下简称支付机构）提供支付创新产品或者服务、与境外机构合作开展跨境支付业务、与其他机构开展重大业务合作的，应当对相关业务的合规性和安全性进行全面评估，并于业务开展前30日书面报告中国人民银行及其分支机构。全国性银行报告中国人民银行；其他银行、支付机构按属地管理原则，报告法人所在地中国人民银行分支



5.2 支付模块用例分析

Payment Module User Cases

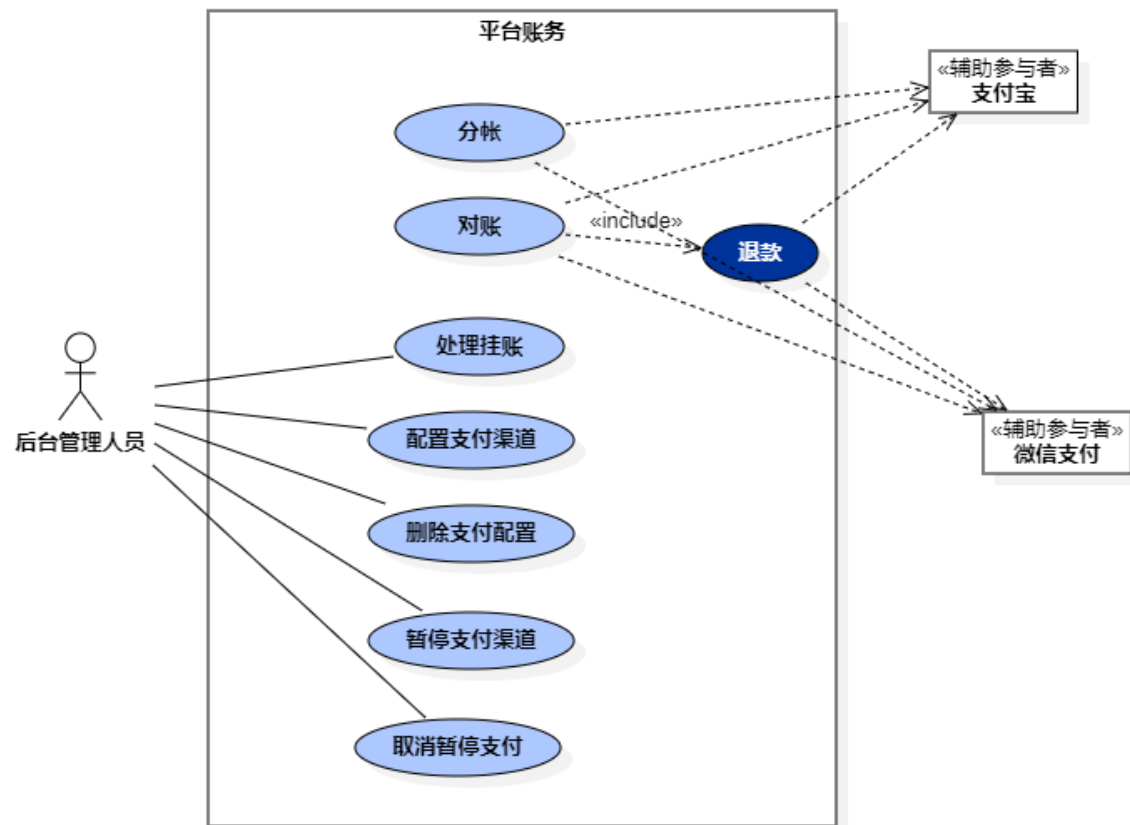


用例总图



5.2 支付模块用例分析

Payment Module User Cases

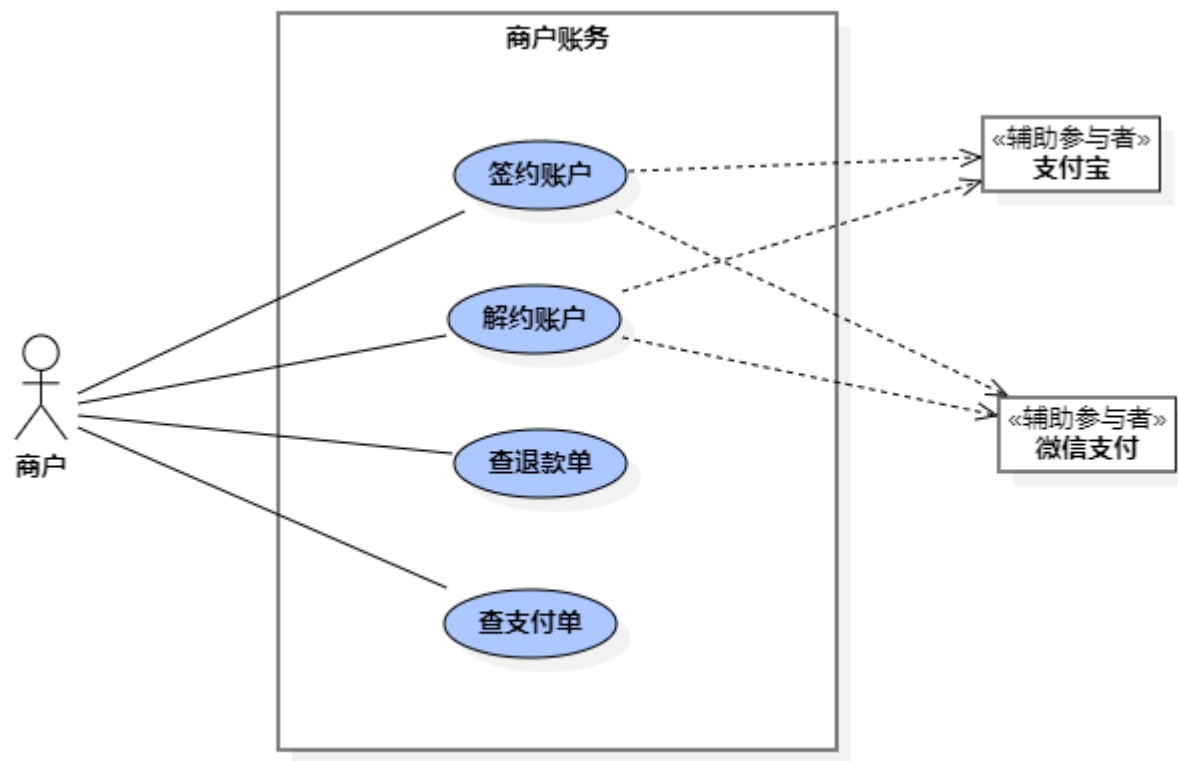


平台账务用例图



5.2 支付模块用例分析

Payment Module User Cases



商户账务用例图



5.2 支付模块用例分析

Payment Module User Cases

用例编号: MALL-PAY-002	用例名称: 退款
级别: 子功能	
包: 平台账务	
参与者:	
描述: 根据交易单和退款金额, 通过第三方支付平台完成退款交易, 积点和优惠券退回原账号	
触发事件: 被其他模块调用	
主成功场景:	信息
1. 其他模块提供交易单id、退款金额和退款分账金额, 提交退款。	交易单号, 退款金额
2. 系统提示退款成功。	
扩展场景:	信息
2a. 退款失败	
1. 系统返回失败原因	失败原因
其他: 需控制退款金额不能超过付款金额, 退款分账金额不超过付款分账金额	



5.2 支付模块用例分析

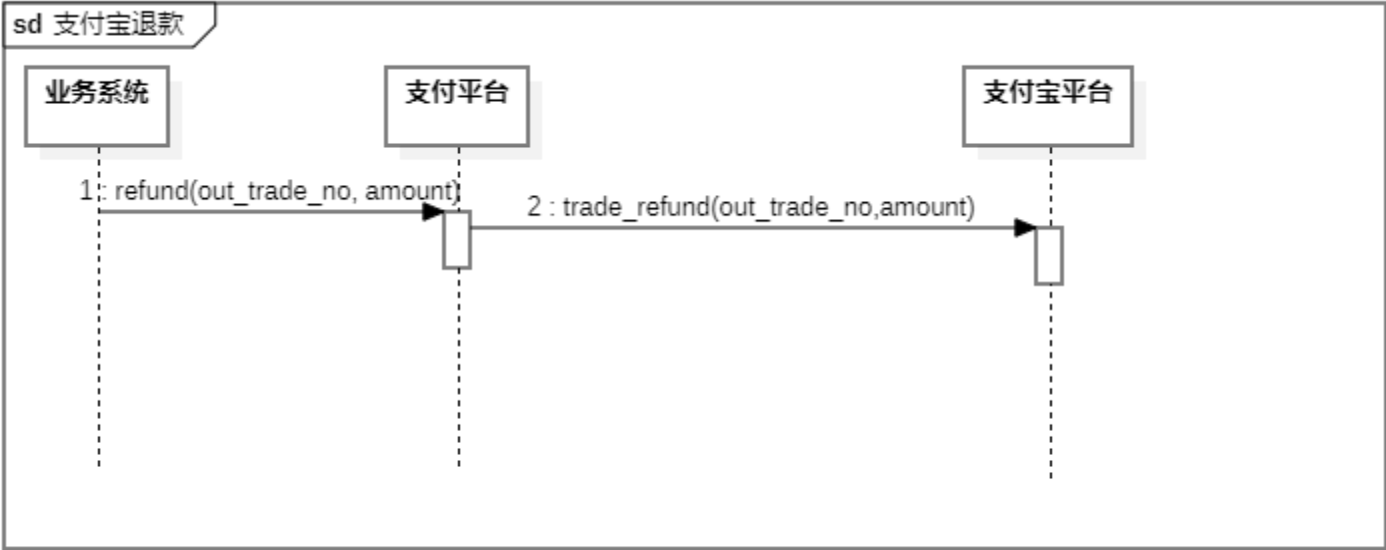
Payment Module User Cases

用例编号: MALL-PAY-006	用例名称: 配置支付渠道
级别: 用户目标	
包: 平台账务	
参与者: 后台管理人员	
描述: 后台管理员增加支付渠道的相关信息和渠道参数 (包括密钥, 商户ID等关键参数信息),	
触发事件: 界面操作	
主成功场景:	信息
1. 系统显示支持的支付渠道, 以及目前已经配置的支付渠道	支付渠道名称
2. 用户从支持的支付渠道中, 选择需配置的支付渠道, 输入渠道参数 (密钥, 平台在渠道申请的商户id)、配置启用时间等	渠道参数 (密钥, 平台在渠道申请的商户id), 各个平台不同
3. 系统提示配置成功	
扩展场景:	信息
2a. 用户从已经配置的支付渠道中选择删除已有配置 (只有启用时间未到的配置才可以删除)	
1. 系统返回第1步, 重新显示已经配置的支付渠道	
其他: 只能配置系统支持的支付渠道,	



5.3 支付模块系统交互

Payment Module SSD

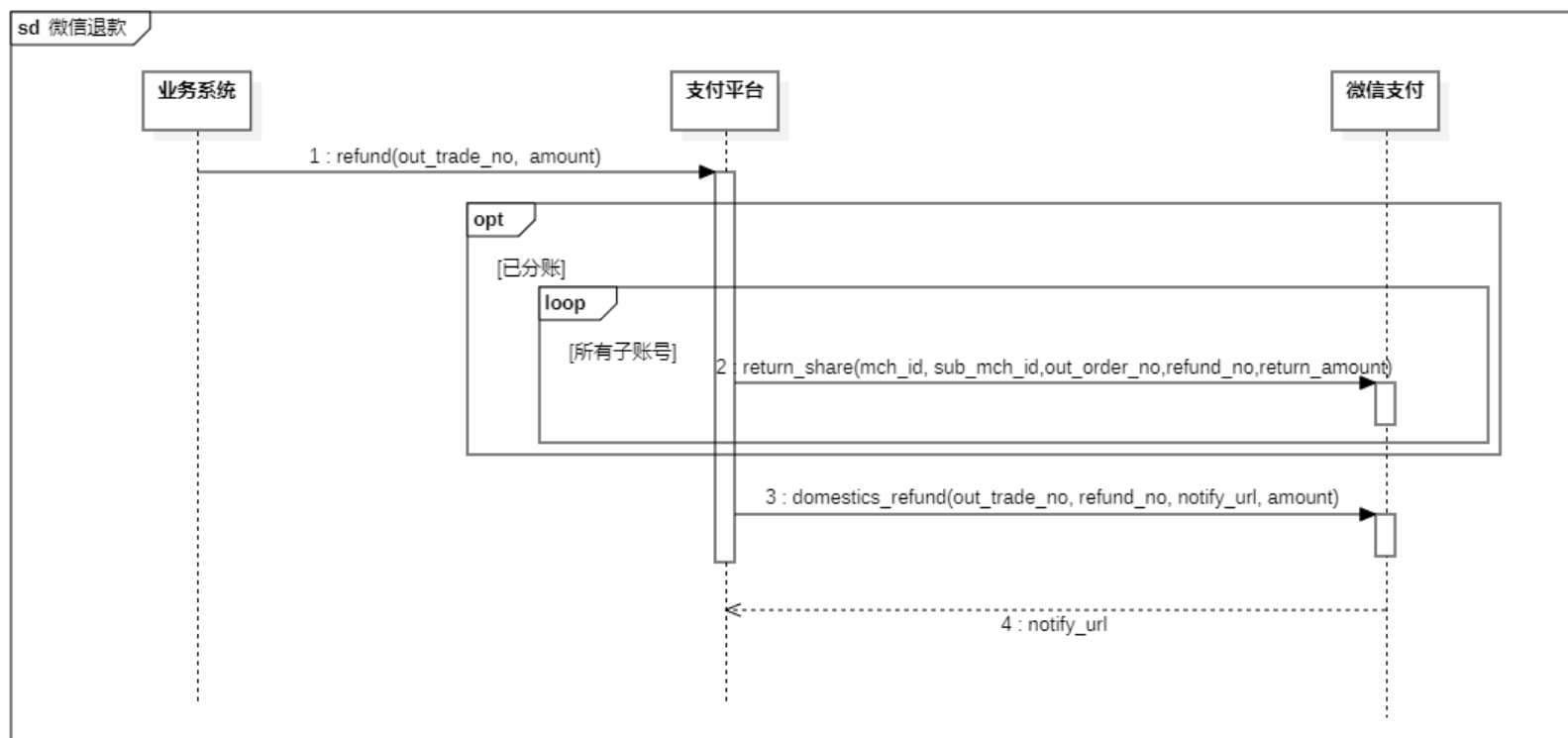


图中方法名	调用说明URL	功能
return share	https://pay.weixin.qq.com/wiki/doc/apiv3_partner/apis/chapter8_1_3.shtml	分账回退
domestics refund	https://pay.weixin.qq.com/wiki/doc/apiv3_partner/apis/chapter4_1_9.shtml	退款



5.3 支付模块系统交互

Payment Module SSD

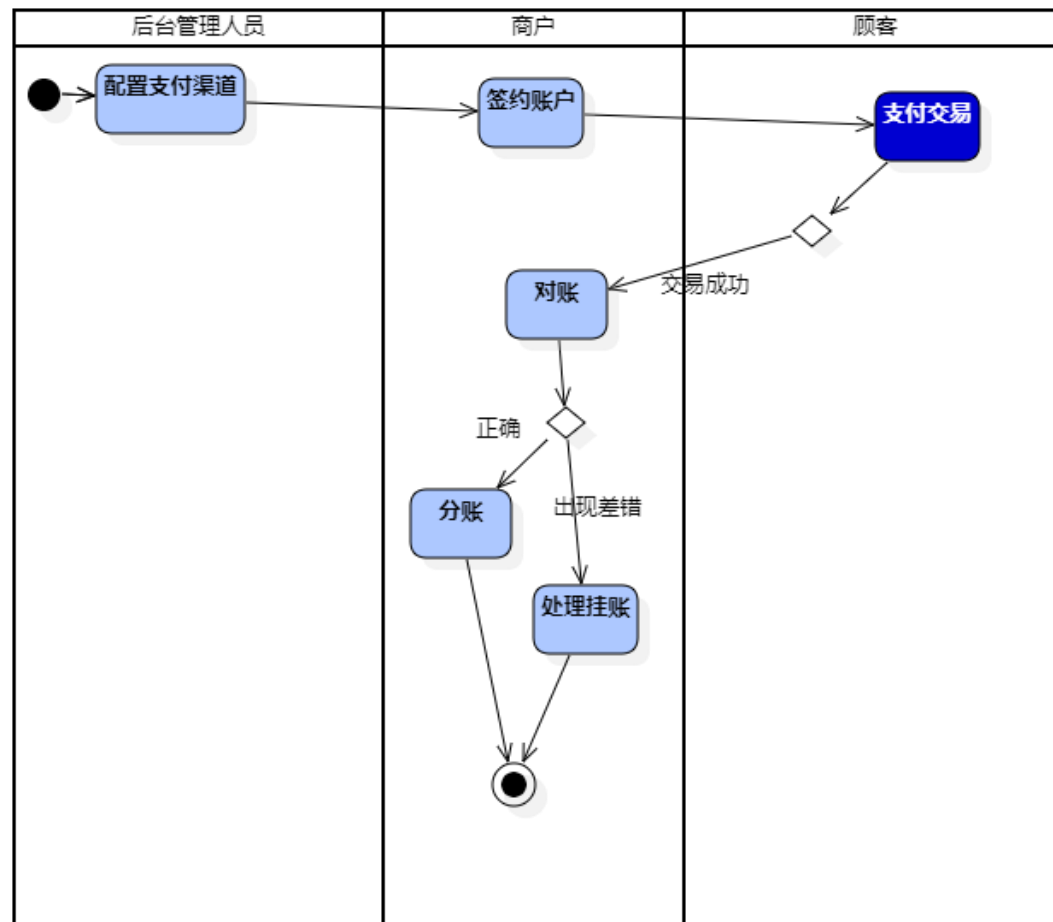


图中方法名	调用说明URL	功能
return_share	https://pay.weixin.qq.com/wiki/doc/apiv3_partner/apis/chapter8_1_3.shtml	分账回退
domestics_refund	https://pay.weixin.qq.com/wiki/doc/apiv3_partner/apis/chapter4_1_9.shtml	退款



5.4 支付模块业务流程

Payment Module Business Process



6. 需求管理

Requirement Management



常见的场景

- 产品提的期望是A，结果做出来演示的是B，实际出现很大的偏差，从期望到真正可落地的需求千差万别；
- 比如老板和产品负责人对需求的理解不一致，经常出现已经定好的目标，产品负责人要变更，但老板觉得没必要，项目经理受夹板气；
- 老板经常性的插入需求，导致变更率居高不下，需求变更流程形同虚设；
- 需求落地执行的过程中，一切都看上去挺顺利的，都在按计划执行，结果到验收阶段，各种问题频出：不是功能的缺失，就是进度的延迟，功能看着像是做完了，但各种小问题一堆；
- 多方不断地插入需求，但时间又不允许增加，还要求按原计划完成，导致团队怨声载道，满意度很低。



6.1 需求管理的过程

Requirement Management Process

- 定义需求基线（迅速制定需求文档的主体）。
- 评审提出的需求变更、评估每项变更的可能影响从而决定是否实施它。
- 估计变更需求所产生影响，并在此基础上协商新的承诺（约定）。
- 使当前的项目计划与需求一致。
- 让每项需求都能与其对应的设计、源代码和测试用例联系起来以实现跟踪。
- 在整个项目过程中，跟踪需求状态及其变更情况。



6.1 需求管理的过程

Requirement Management Process

- 采用什么需求管理工具和技术
- 定义需求变更处理过程
- 定义需求状态跟踪与发布过程
- 定义需求变更影响分析
- 需求变更对项目计划的影响度



6.2 需求变更

Requirement Change

- 需求变更原因
 - 单纯用户的因素
 - 市场形势的变化
 - 系统因素
 - 工作环境和要求的变化
 - 需求开发的缺陷
 - 需求分析、定义和评审不充分
 - 与用户沟通不畅



6.2 需求变更

Requirement Change

- 需求变更对软件开发的影响
 - 使变更前开发工作和成果失效
 - 工作量及资源投入的增加
 - 项目完成时间的延后



6.2 需求变更

Requirement Change

- 降低需求变更风险的策略
 - 与用户充分沟通，明确确定需求的意义
 - 与用户共同确定需求，作为合同附件，签字生效
 - 合同中包含对需求变更的条款
 - 采用原型法开发，或螺旋模型开发
 - 在项目计划中预先留有余地
 - 严格实施变更控制



6.2 需求变更

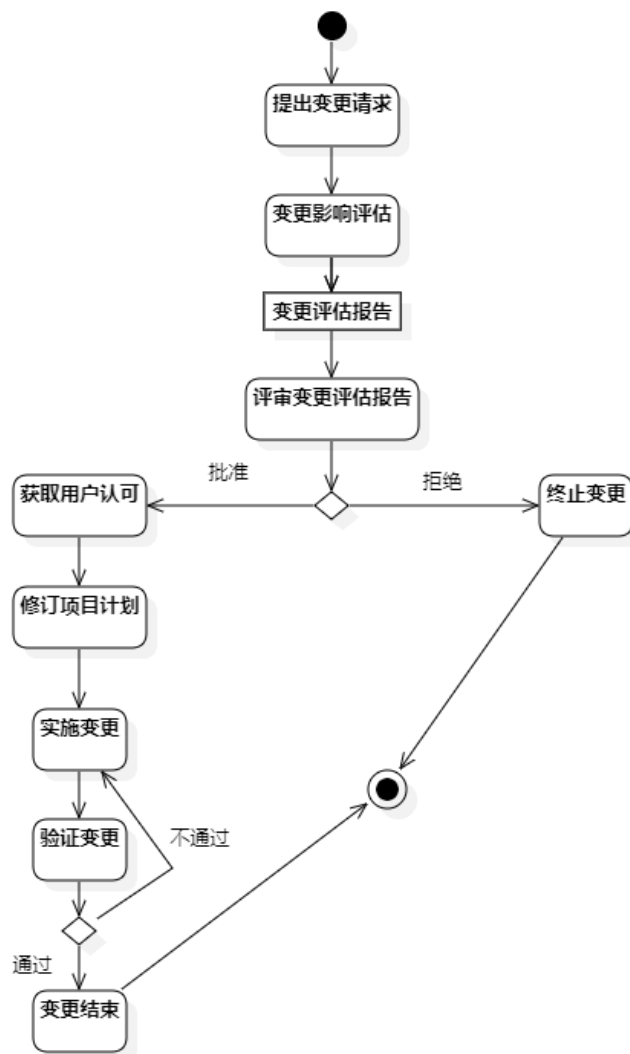
Requirement Change

- 需求变更控制的策略
 - 所有需求变更必须遵循需求变更控制规程实施变更
 - 需求变更提出后是否被接受，应由专门的组织-变更控制委员会审查决定
 - 不得以任何理由删除和修改需求变更的原始文件
 - 应将已接受的需求变更通知到所有相关人员
 - 已接受的需求变更应能追溯到批准的变更请求
 - 对项目的需求赋予状态属性，以利于需求变更的控制



6.2 需求变更 Requirement Change

- 需求变更的规程



变更要求 在财务模块中增加计算XX类产品成本及相关报表		
(系统名称) XX软件		RFC序号:
申请人: 张三 日期(日/月/年): 09年10月11日		
申请变更内容及状态: 系统自动核算XX类产品成本, 生成相应的报表。。。, 目前该模块已经完成需要分析与设计。		
申请变更原因: XX用户XX工作需要。。。		
变更的影响分析: 在质量上。。。在时间上。。。在费用上。。。		
变更类别(标明A、B或C): <u>A</u> 功能方面 B 运行性能方面 C 文档方面		
授权人签字:		日期:



6.2 需求变更

Requirement Change

- 变更影响分析
 - 影响分析确保了对变更的准确理解，为变更决策提供数据依据。
 - 影响分析的完备性和准确性依赖于需求跟踪的完整性和质量。
 - 变更影响的软件要素包括：其它相关需求、设计、代码、测试、培训文档、帮助文档、系统软件等。



6.2 需求变更

Requirement Change

- 变更影响评估的内容
 - 分析需求变更对工作、产品的影响（质量等）
 - 估计变更请求所需的工作量，重新估计交付成果的进度（延后或提前多少？）
 - 估计增加或减少的成本
 - 得出评审结果（通过否？）
 - 若评审通过，则更改相应的工作产品（如软件），使其与变更的需求保持一致
 - 若评审未通过，将需求变更请求表及相应文档存档



6.2 需求变更

Requirement Change

- 需求变更控制的难点在于：
 - 变更控制流程是否存在于项目组每个成员心中。
 - 当需求变更大量涌现时，CCB是否有足够的人力、精力、能力和耐心，认真处理每一个变更



6.3 需求追踪

Requirement Traceability

- 确保每一个需求被实现、被测试。
- 当需求变更发生时，确保影响分析的完备性。
- 跟踪需求的状态，了解进度情况。
- 复用：系统升级时，可借助需求跟踪矩阵复用旧系统的资产，如功能需求、设计、代码和测试用例。
- 在测试出错时，可借助联系链，有效分析相关的代码。
- 借助联系链，将相关文档、代码关联起来。



6.3 需求追踪

Requirement Traceability

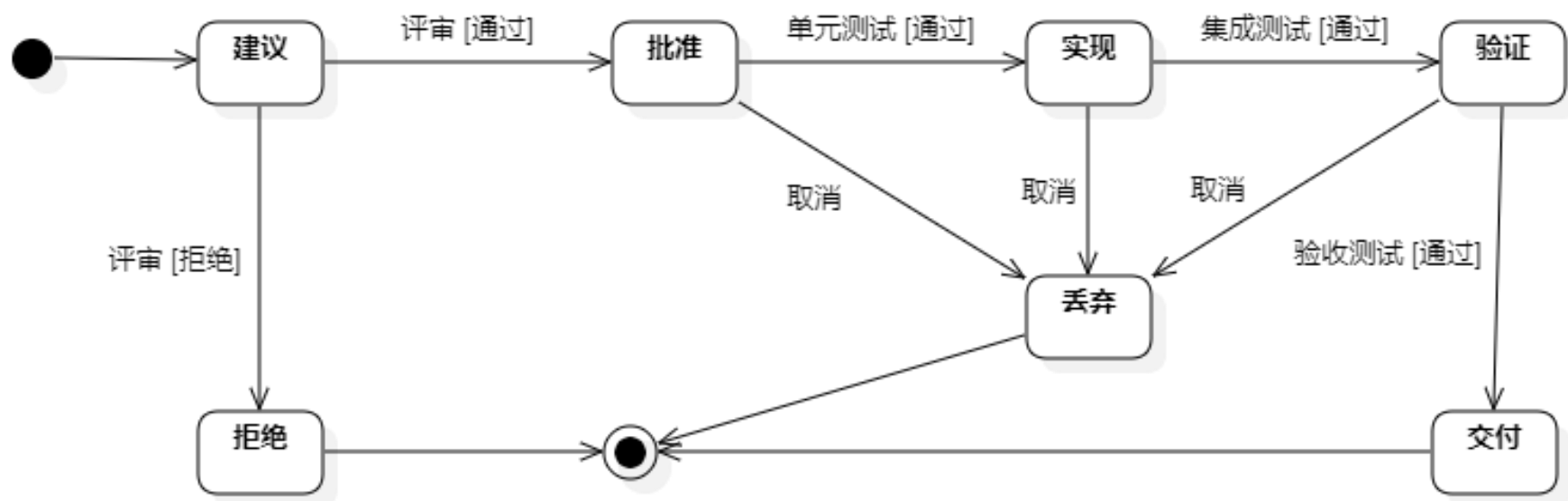
- 需求的属性
 - 需求创建时间
 - 版本号
 - 作者
 - 需求来源
 - 确认需求的客户代表
 - 需求涉及的子系统
 - 需求对应的产品版本号
 - 需求状态
 - 需求优先级
 - 测试标准
 - 需求的稳定性



6.3 需求追踪

Requirement Traceability

- 需求的状态



6.3 需求追踪

Requirement Traceability

- 需求追踪的目标
 - 建立与维护“需求—设计—编程—测试”之间的一致性，确保所有的工作成果符合用户需求。
 - 各个开发阶段的工作产品之间存在着关联关系，形成可追踪矩阵
 - 编制每个需求同系统元素之间的联系文档。这些元素包括别的需求、体系结构、其他设计部件、源代码模块、测试、帮助文件、文档等。



6.3 需求追踪

Requirement Traceability

- 需求跟踪有两种方式：
 - 正向跟踪：检查《产品需求规格说明书》中的每个需求是否都能在后继工作成果中找到对应点。
 - 逆向跟踪：检查设计文档、代码、测试用例等工作成果是否都能在《产品需求规格说明书》中找到出处。



项目需求跟踪矩阵							
说明:							
1 需求跟踪矩阵的主要目的是跟踪，查找对应关系。							
2 公共的模块需要在设计文档中体现出来。							
项目名称				文档版本			
作者				日期			
跟踪矩阵							
业务需求	需求规格	概要设计	详细设计	源代码	单元测试用例范围	集成测试用例范围	系统测试用例范围
BR001	SRS001	HLD001	LLD001	CODE001	UT001-UT010	IT001-IT012	ST001-ST005
				CODE002			
			LLD002	CODE003			
				CODE004			
		HLD002	LLD003	CODE005			
				CODE006			
			LLD004	CODE007			
				CODE008			
BR002	SR002						
业务需求							
大类（如子系统）	业务需求编号	业务需求名称	优先级	状态	说明		
需求规格							
大类（如子系统）	需求规格编号	需求规格名称	优先级	状态	说明		
概要设计							
大类（如子系统）	概要设计编号	概要设计名称	优先级	状态	说明		



6.3 需求追踪

Requirement Traceability

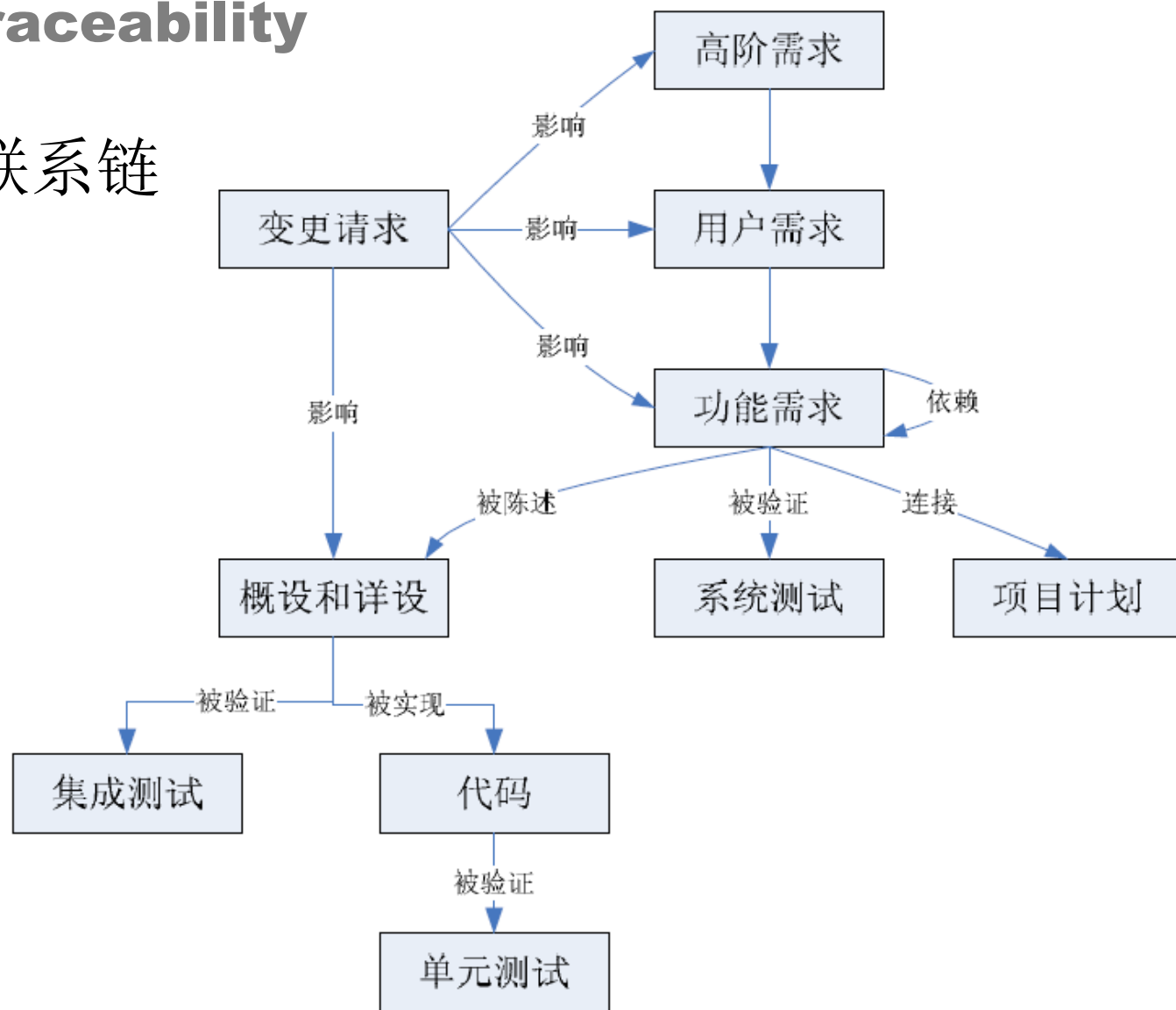
- 跟踪矩阵的作用
 - 考察需求是否有遗漏的情况
 - 有无冗余代码
 - 检查所有的性能需求已被测试用例测试
 - 对集成测试和系统测试进行交互检查



6.3 需求追踪

Requirement Traceability

- 需求跟踪能力联系链



6.3 需求追踪

Requirement Traceability

- 需求管理工具的作用
 - 建立功能需求与设计、代码、测试用例之间的联系。
 - 更好跟踪每项需求属性和状态。
 - 更好管理需求的版本。
 - 需求变更时不需要手工通知受影响的人员。

