

软件需求设计UML全程实作

需求

潘加宇



建模 workflow

*业务建模

愿景

业务用例图

现状业务序列图

改进业务序列图

*需求

系统用例图

系统用例规约

*分析

分析类图

分析序列图

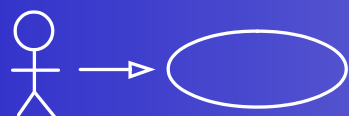
分析状态机图

*设计

建立数据层

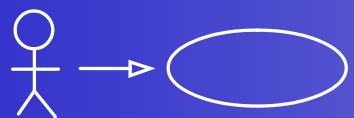
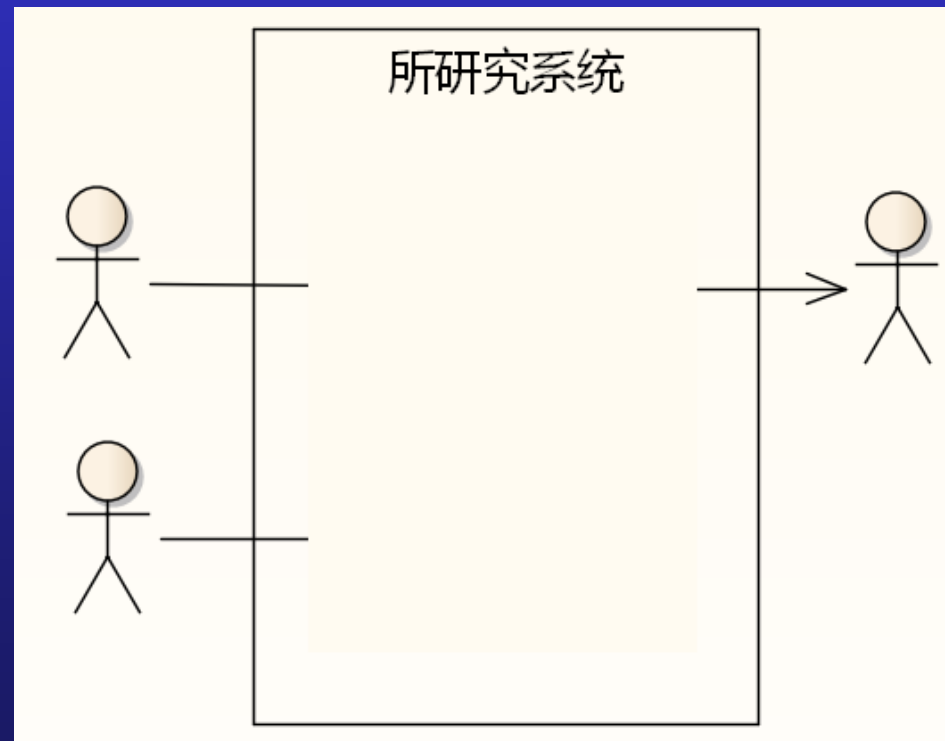
精化业务层

精化表示层

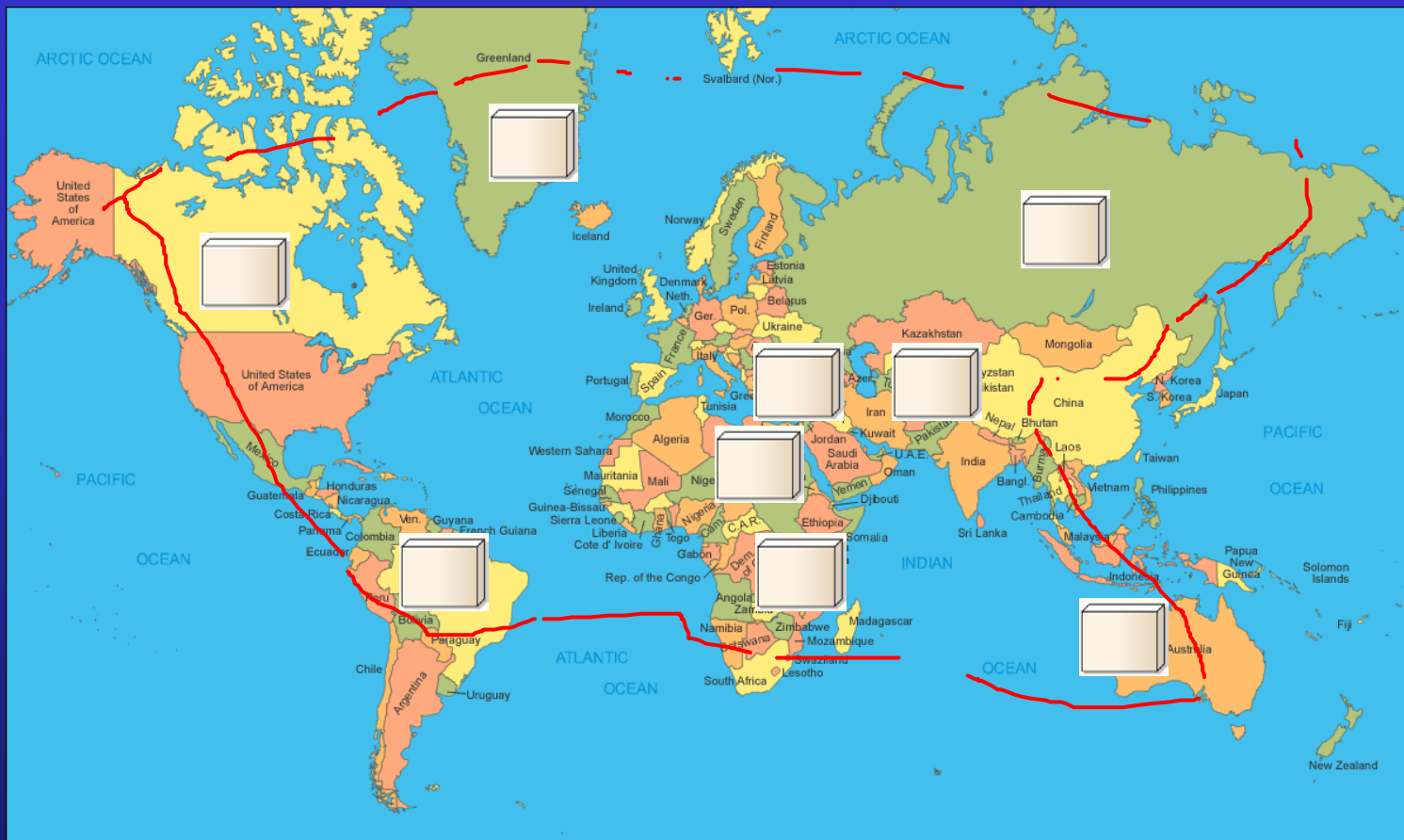


系统执行者

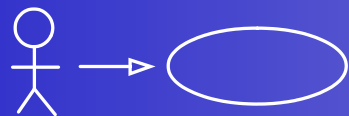
在所研究系统外，
与该系统发生功能性交互的其他系统



系统执行者



责任的边界，不是物理的边界



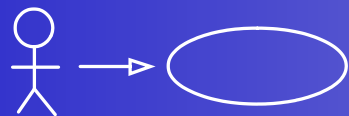
系统执行者



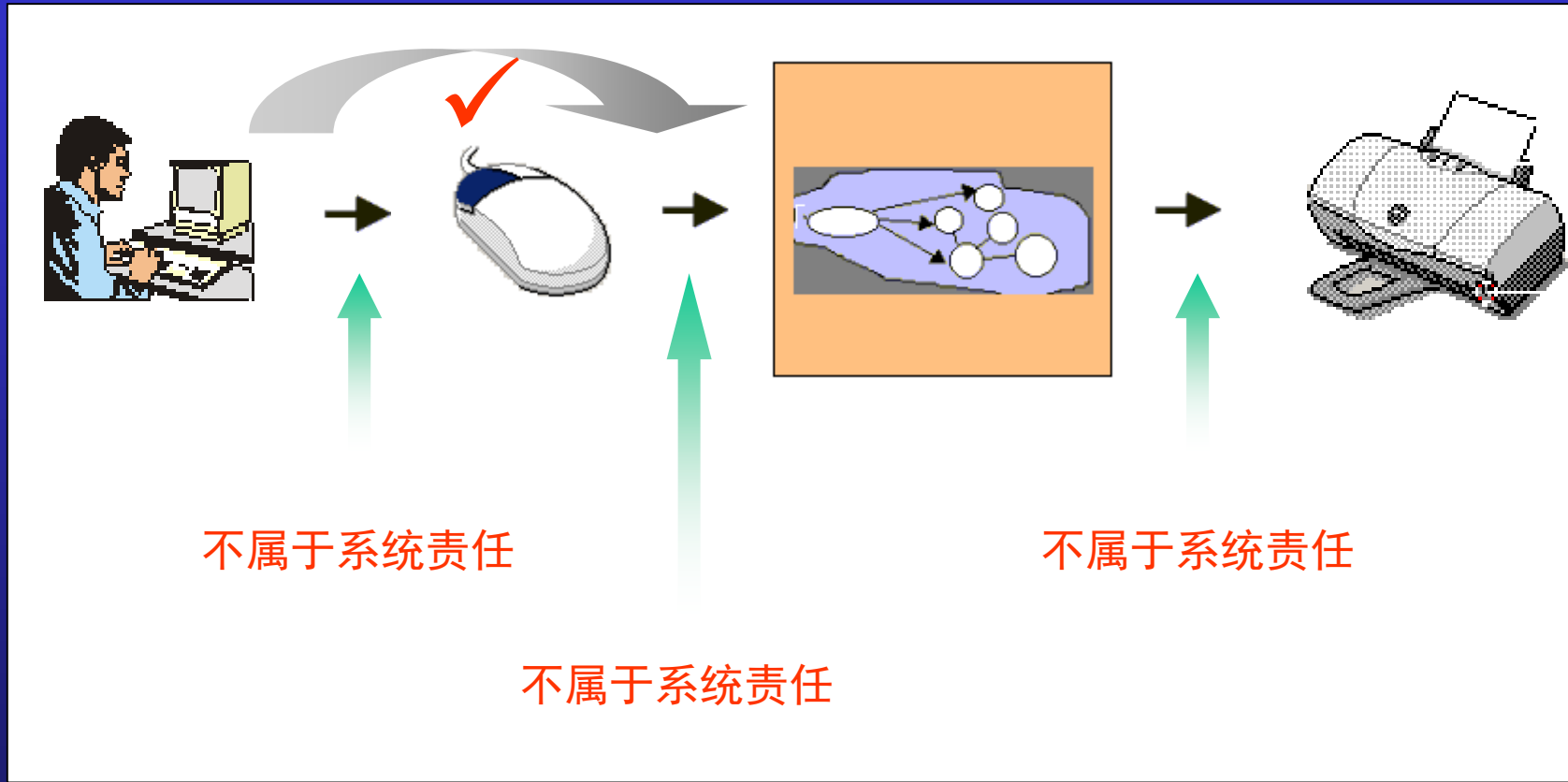
涉众

执行者

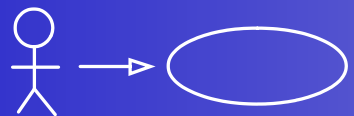
交互——与重要无关



系统执行者



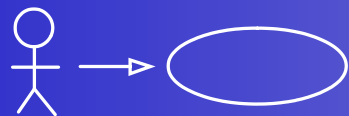
功能性交互



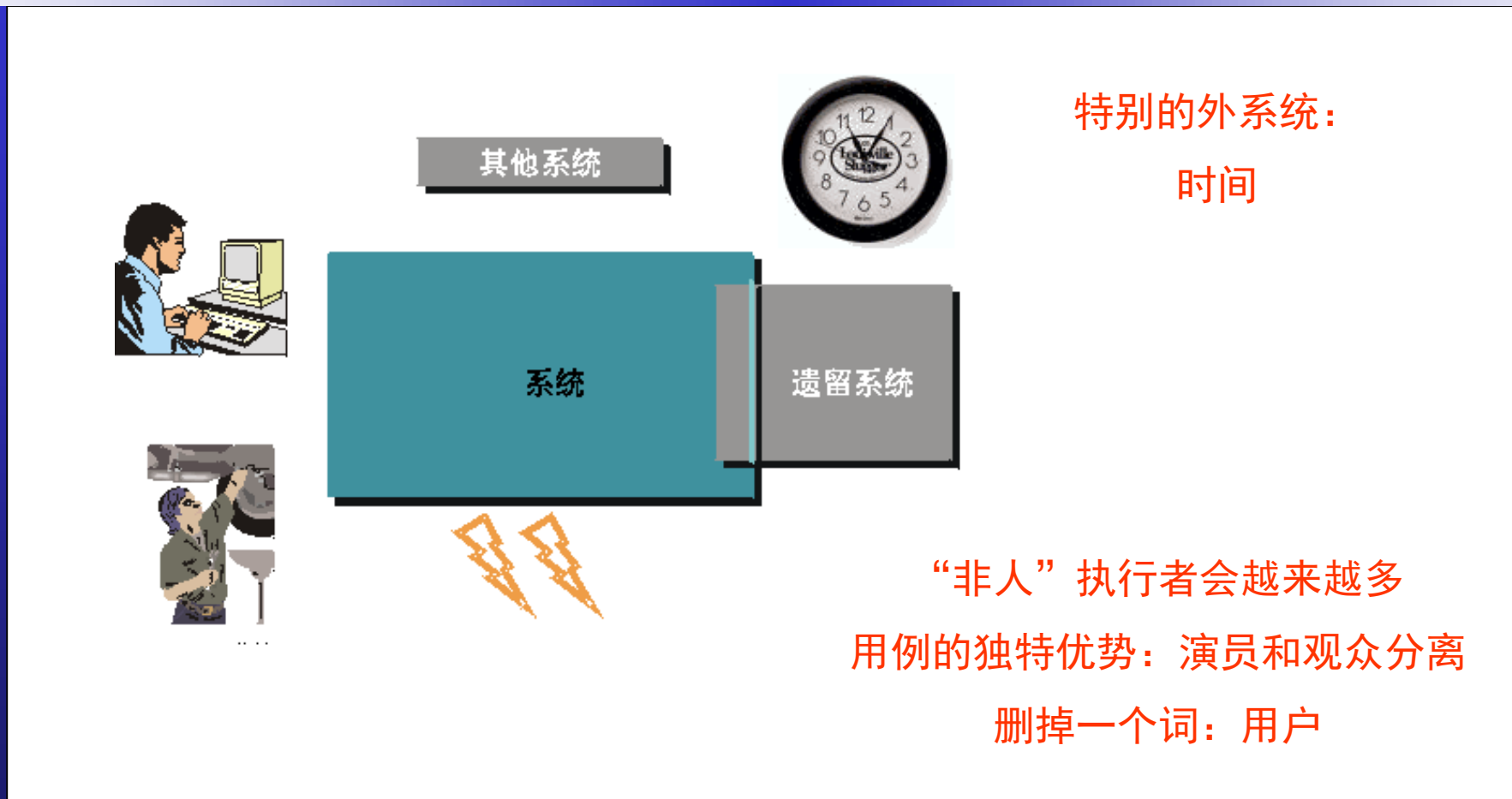
系统执行者



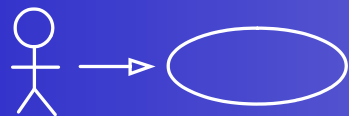
执行者代表功能需求



系统执行者

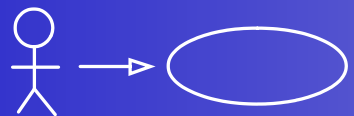


任何系统

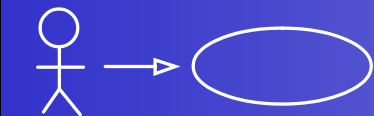


系统执行者

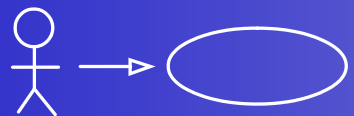
——讨论与练习、项目实作



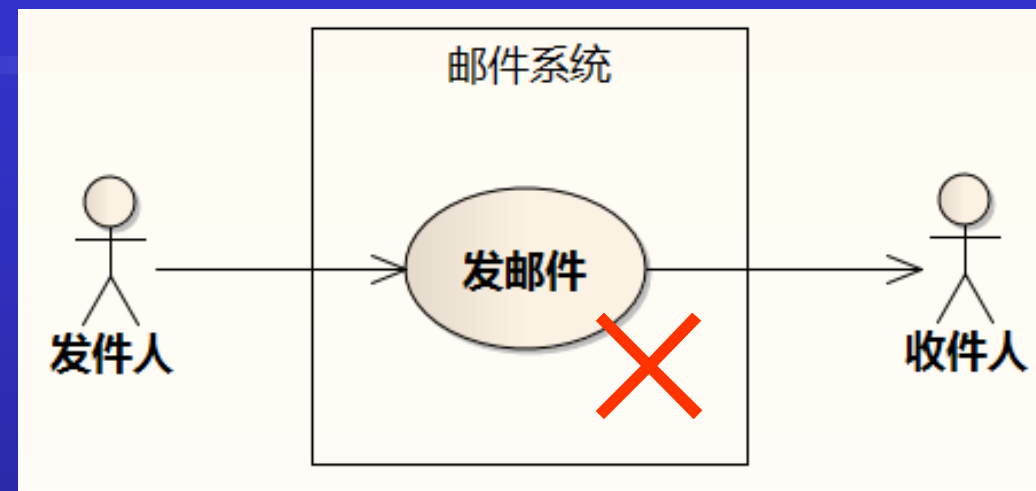
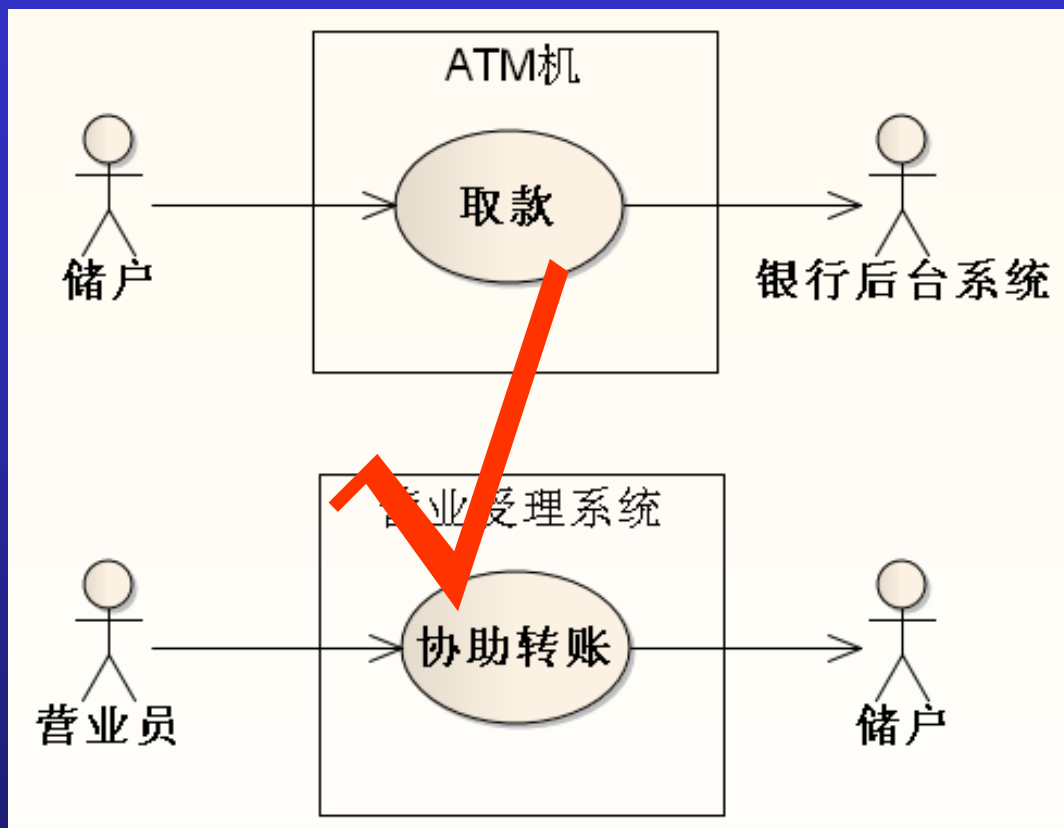
系统执行



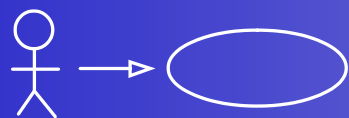
系统执行者



系统执行者

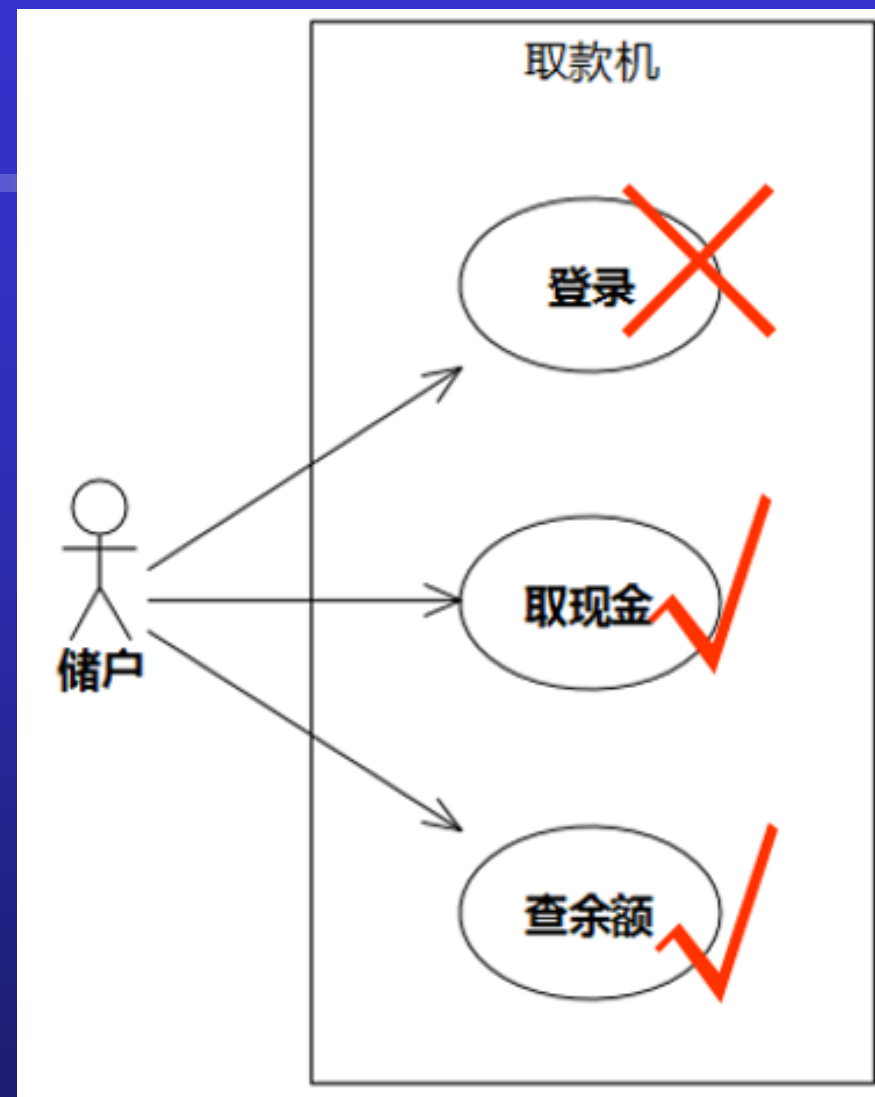


主执行者、辅执行者



系统用例

- 期望和承诺的平衡点
- 买和卖的平衡点
- 我去系统那里什么一下
- 更难！

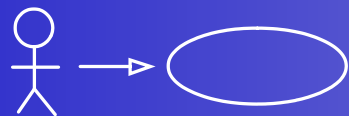


系统能够为执行者提供的，涉众可以接受的价值

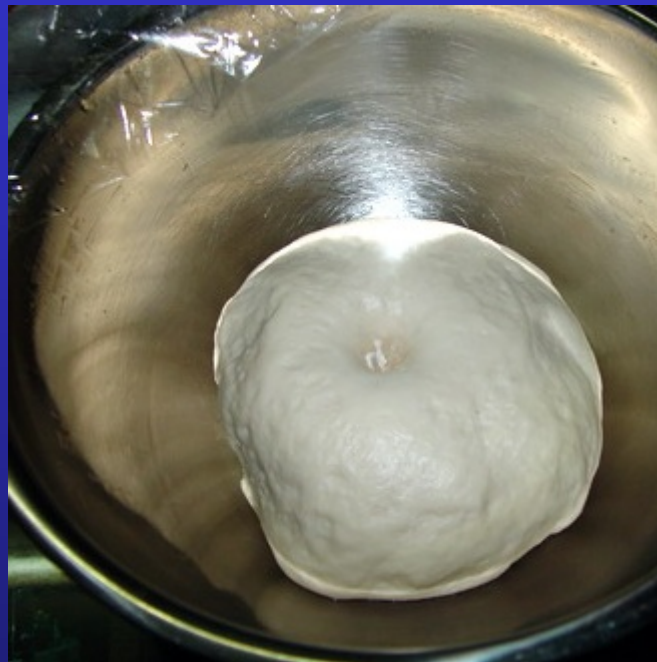


系统用例

- 针对什么样的系统来说，“登录”是合适的用例？
- 针对什么样的系统来说，“输入密码”是合适的用例？
- 有人说：有时我就是喜欢登录取款机看看我的密码对不对，然后就觉得达到目的退出了，登录难道不是取款机的用例吗？
- 小崔经受着失眠的痛苦，经常精神恍惚。有一天他在取款机取钱时，居然把银行卡往取款机一插就走开了。回家之后，发现自己可能会丢钱，心里居然生出一种舒适感，过一会就安然入睡了。小崔尝到这个甜头后，干脆睡不着时就跑到楼下取款机插一张卡，回家后果然酣然入睡。在小崔看来，千金难买安稳觉，这样做是划算的。请问：如果世界上确实有小崔这样的人，那么插卡是取款机的用例吗？

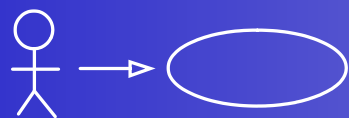


系统用例

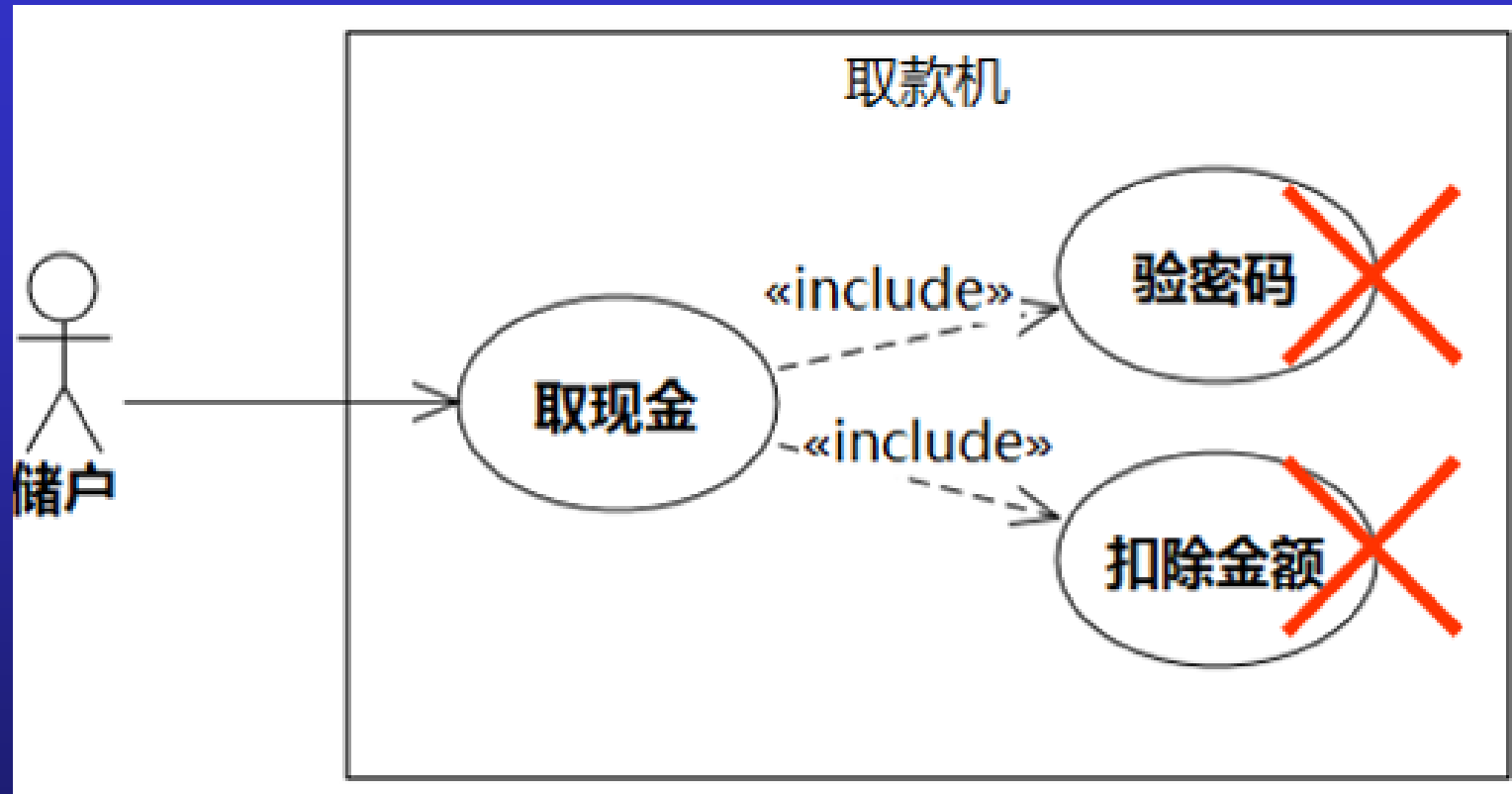


用例不是面团
由开发人员乱捏

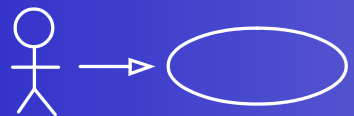
用例不存在“粒度问题”



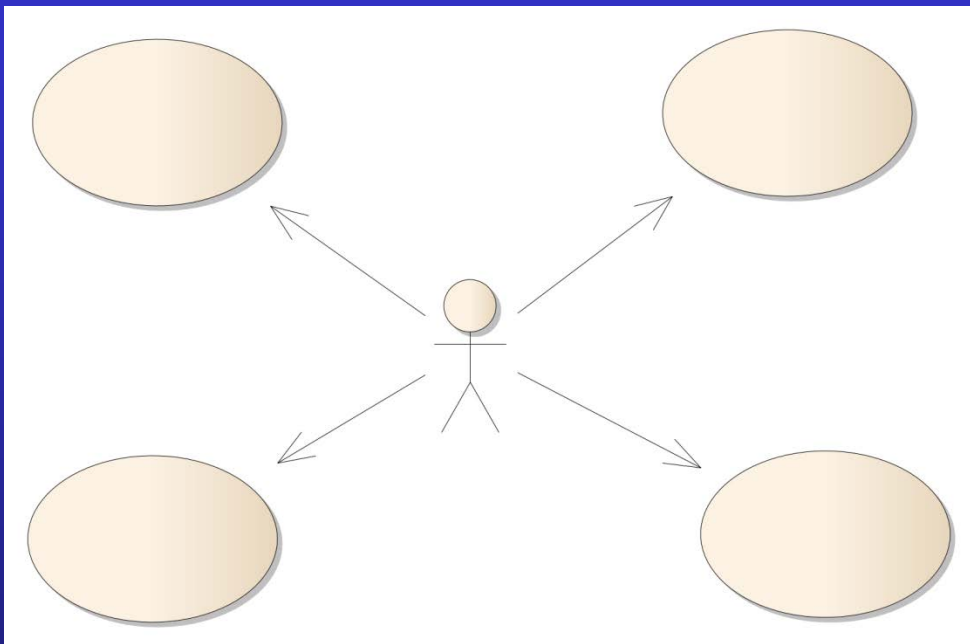
系统用例



最常犯“粒度”错误：把步骤当作用例



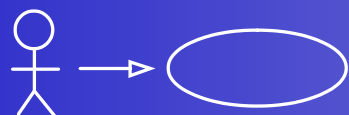
系统用例



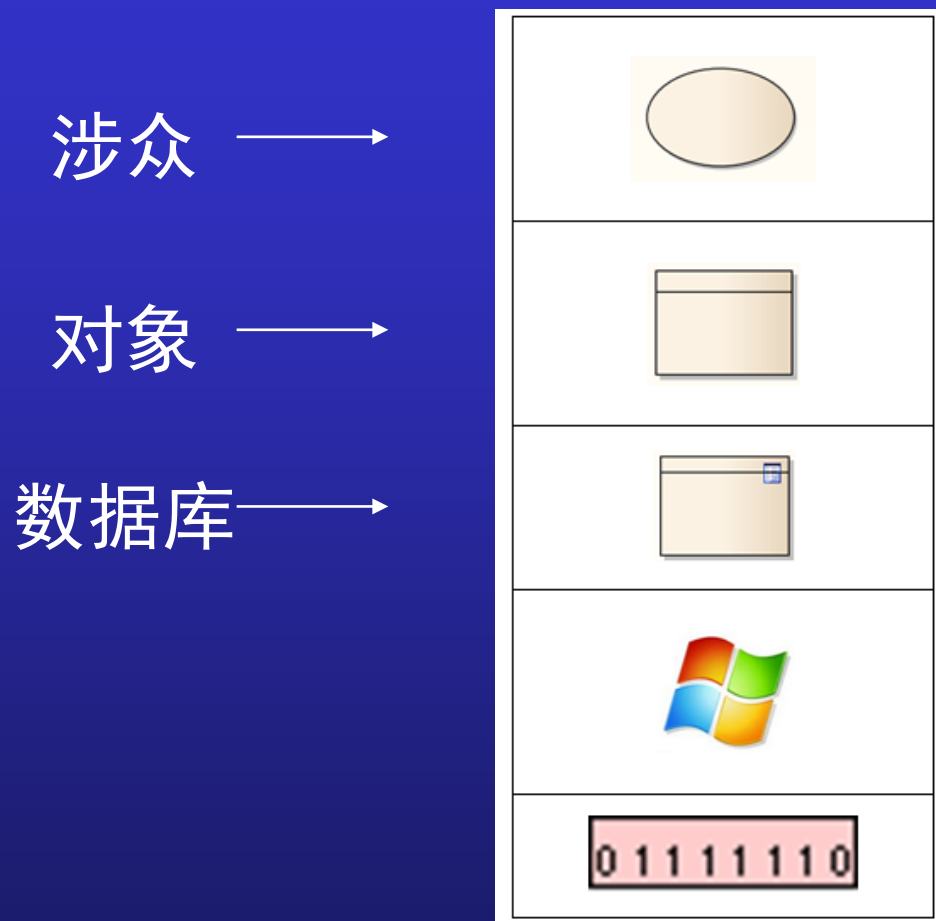
用有色眼镜看
所有业务最终都会成为CRUD

CRUD：新增、读取、修改、删除

四轮马车



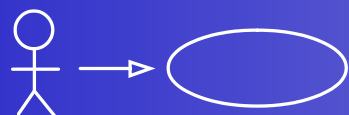
系统用例



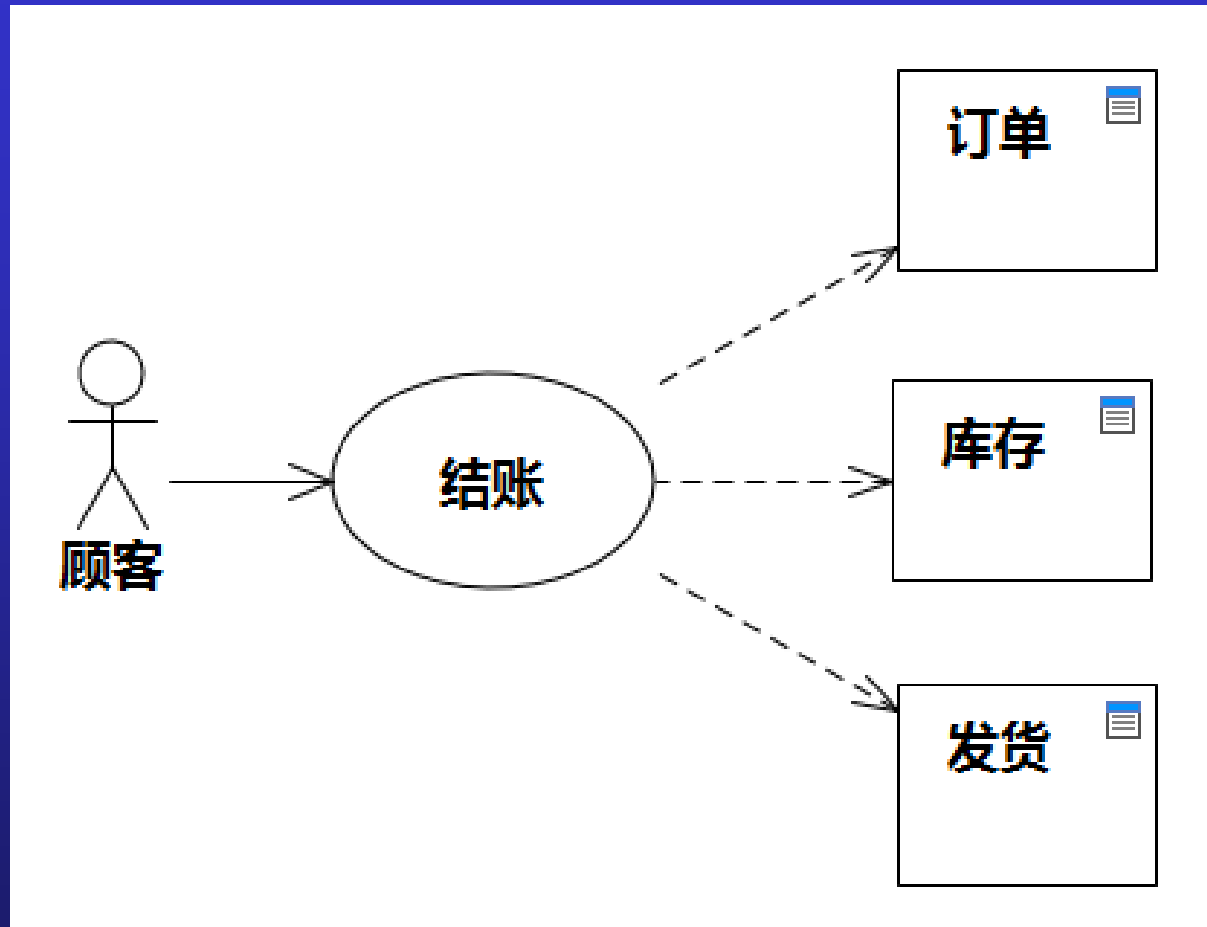
对“好卖”无帮助的
“需求”工作无意义

← 我就是程序，程序就是我

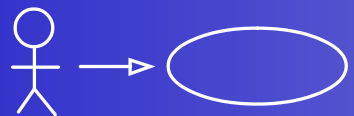
需求只能从涉众视角获得——卖



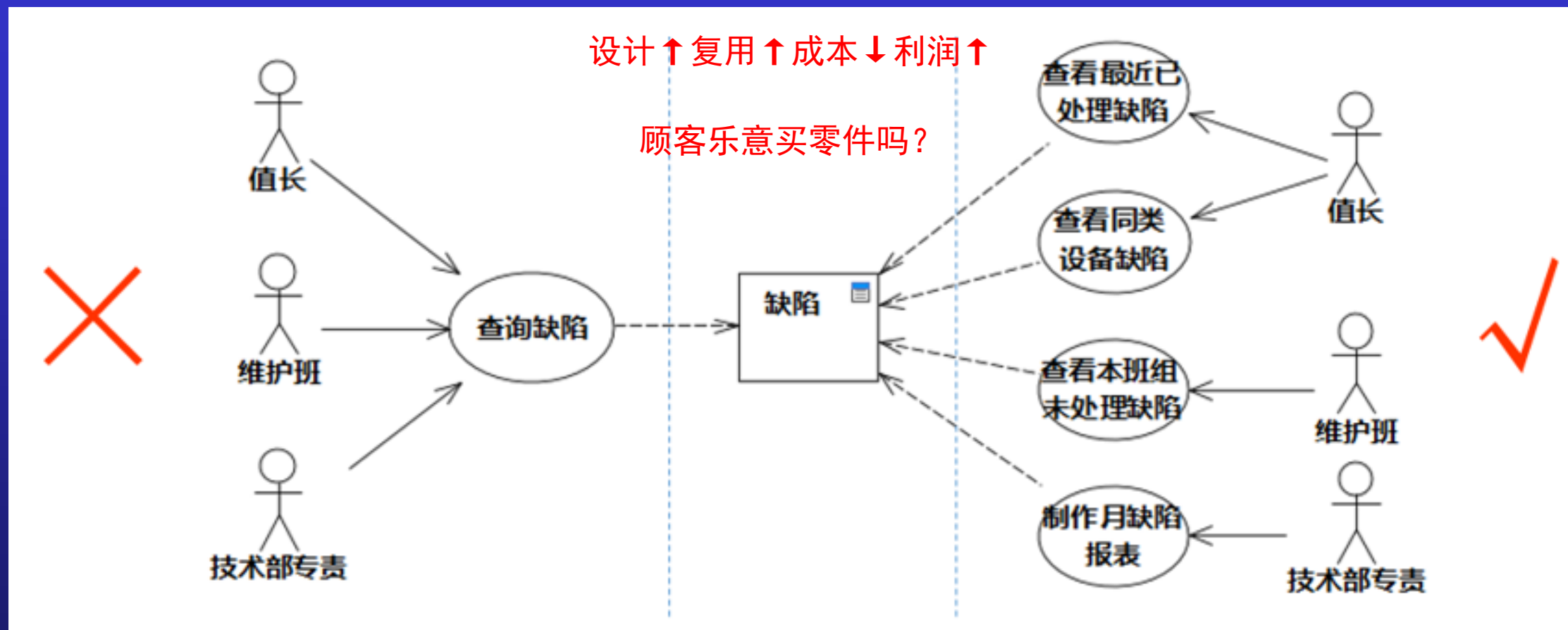
系统用例



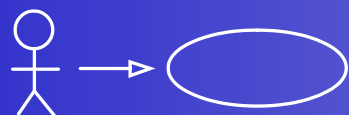
一个用例可能读写许多数据



系统用例



多个用例会读写同一数据

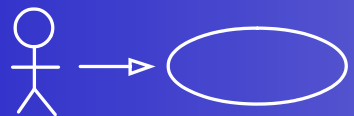


系统用例



闭着眼睛把饺子
当馄饨吃下去？

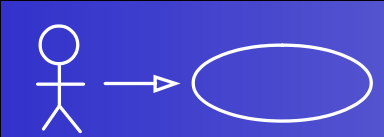
需求是不“复用”的，“复用”的是设计



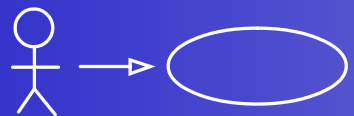
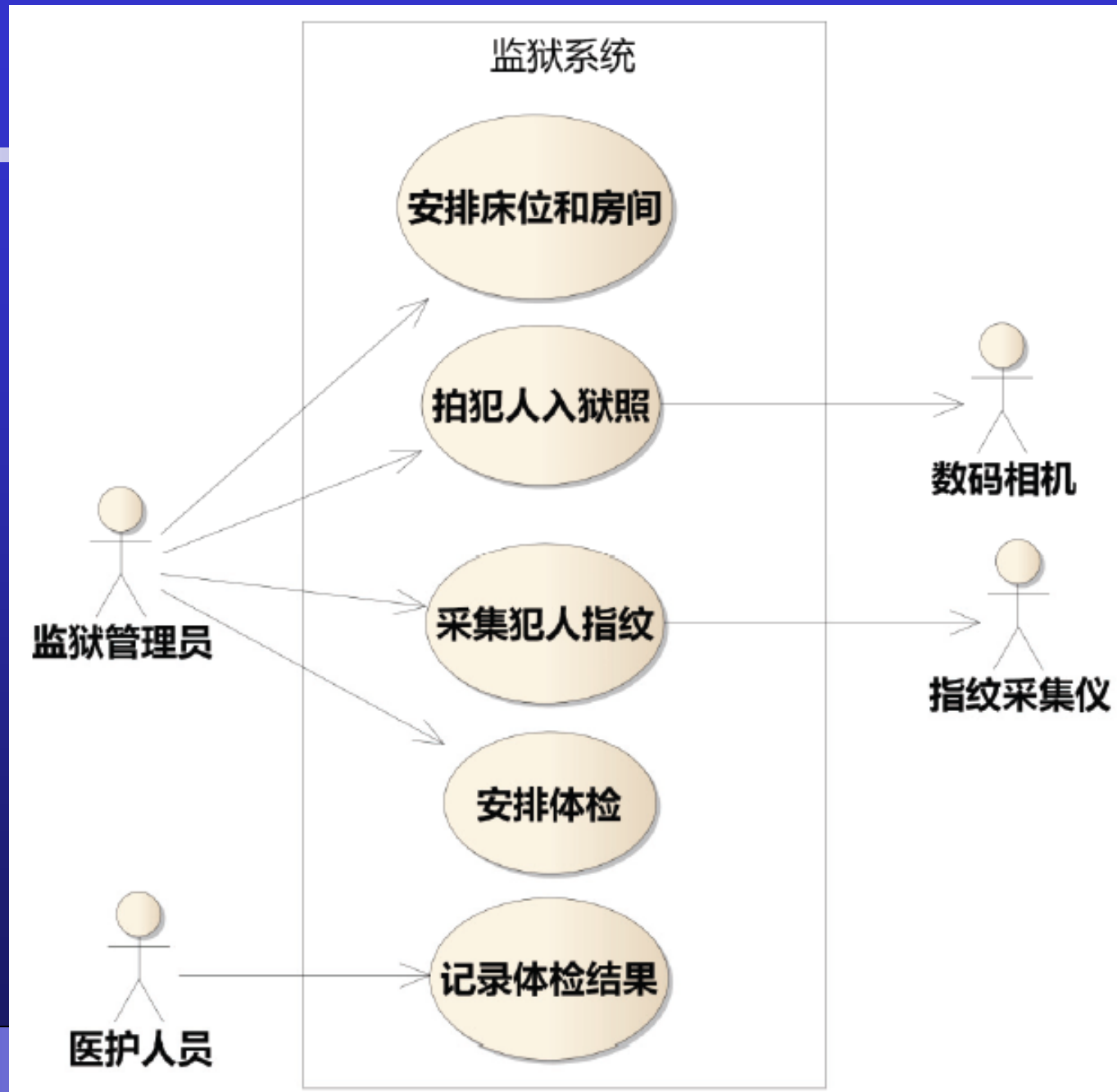
系统用例



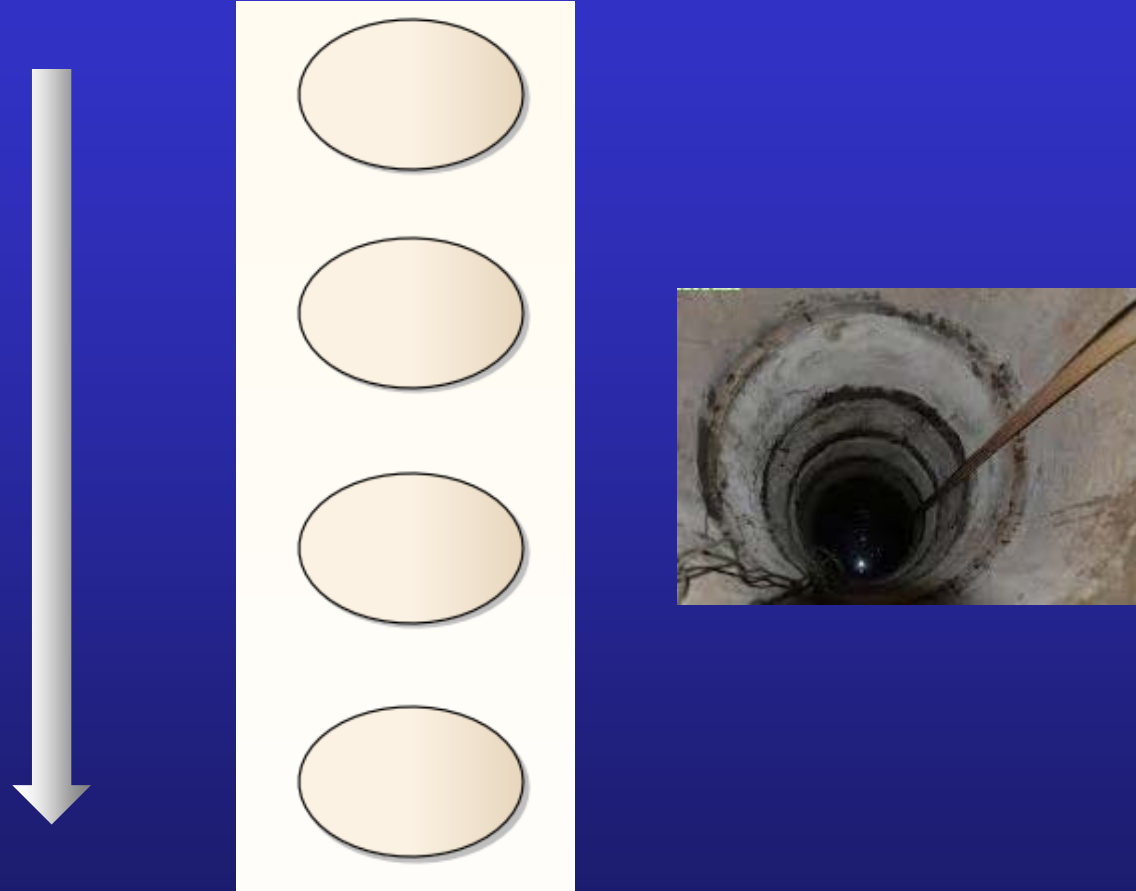
有了业务建模，就不用头脑风暴了！
从外部指向所研究系统的就是该系统的用例



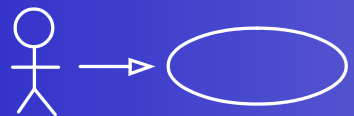
系统用例



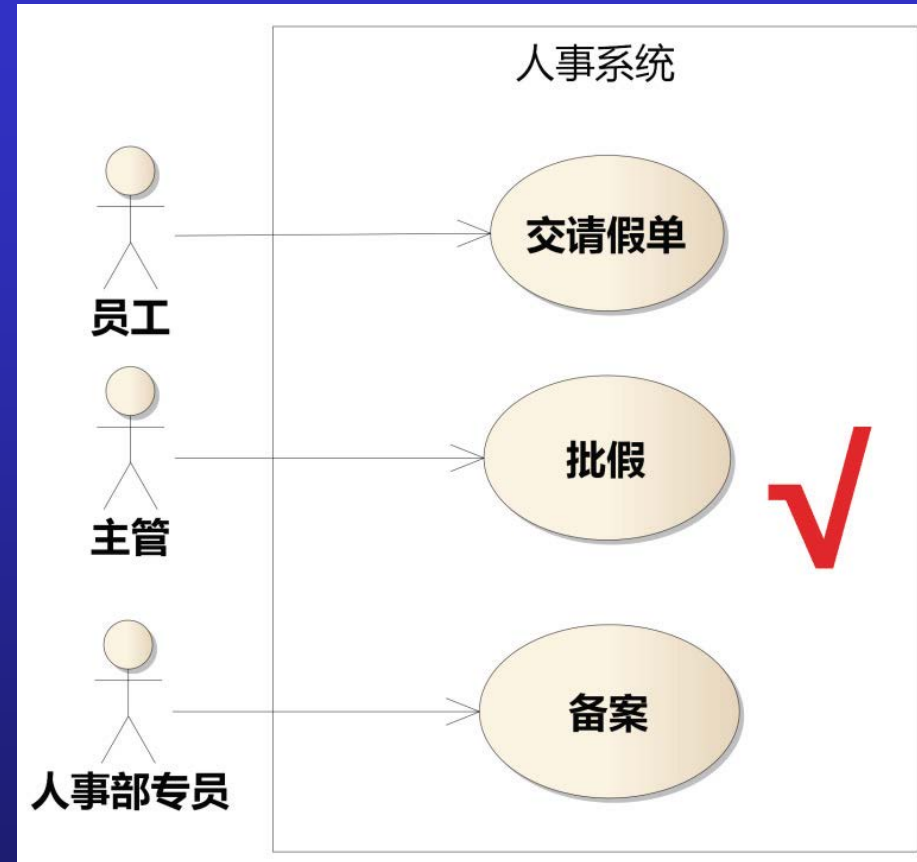
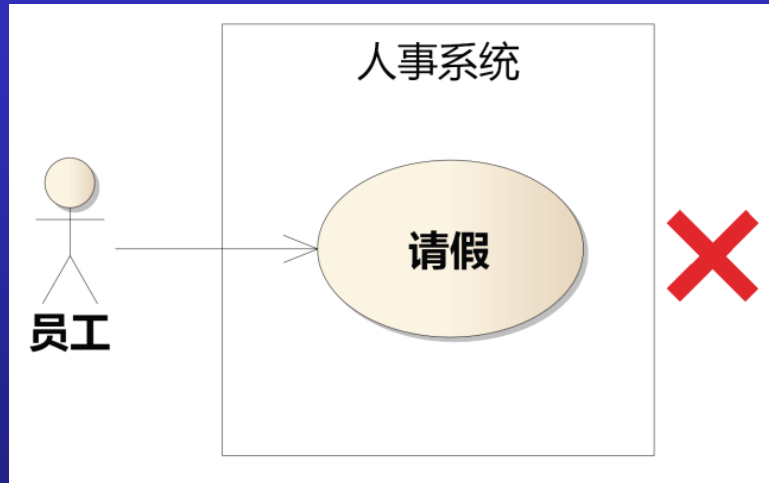
系统用例



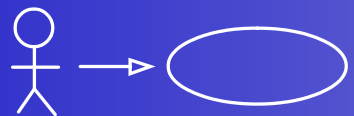
根据愿景对用例排序



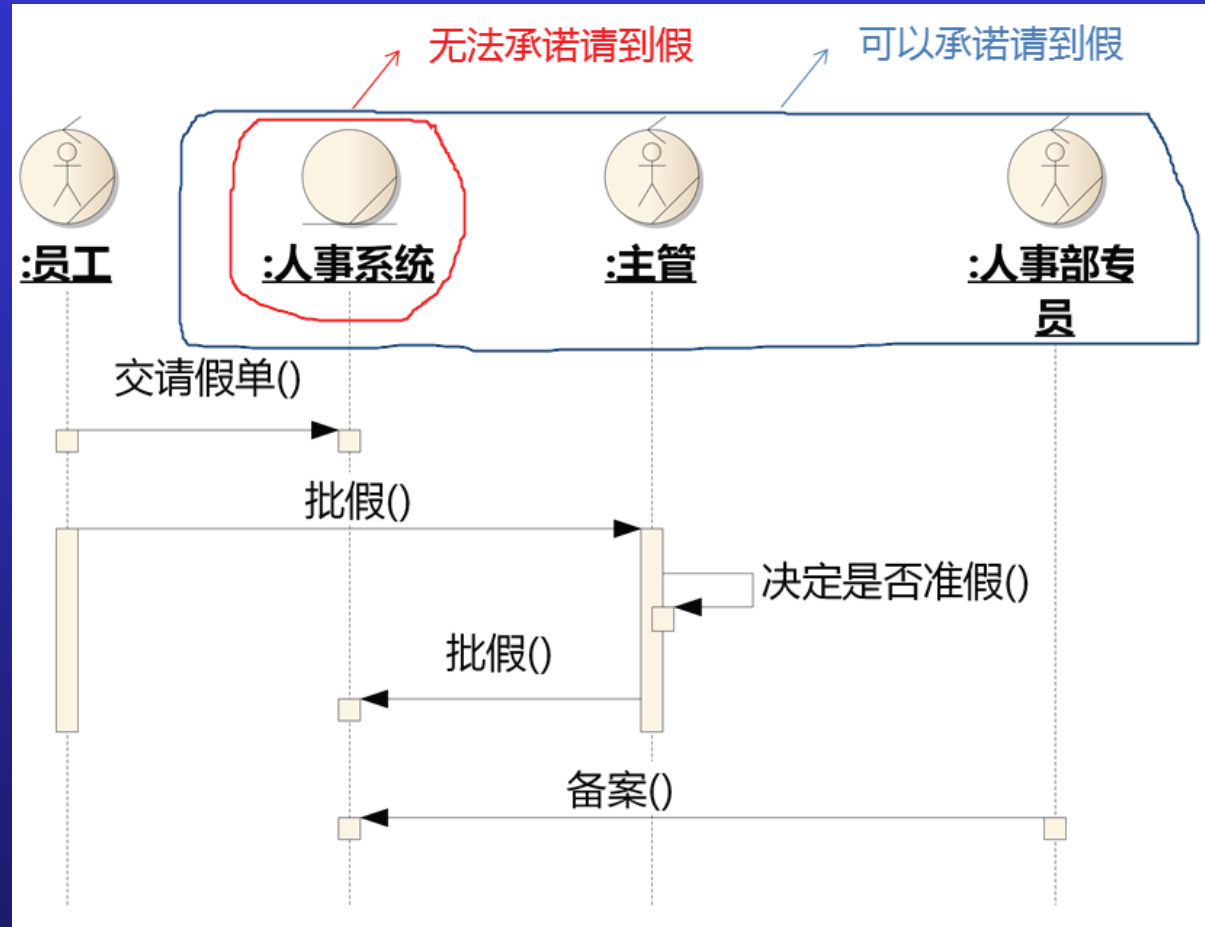
系统用例



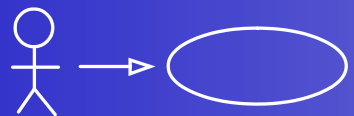
用例多大——期望、契约、承诺



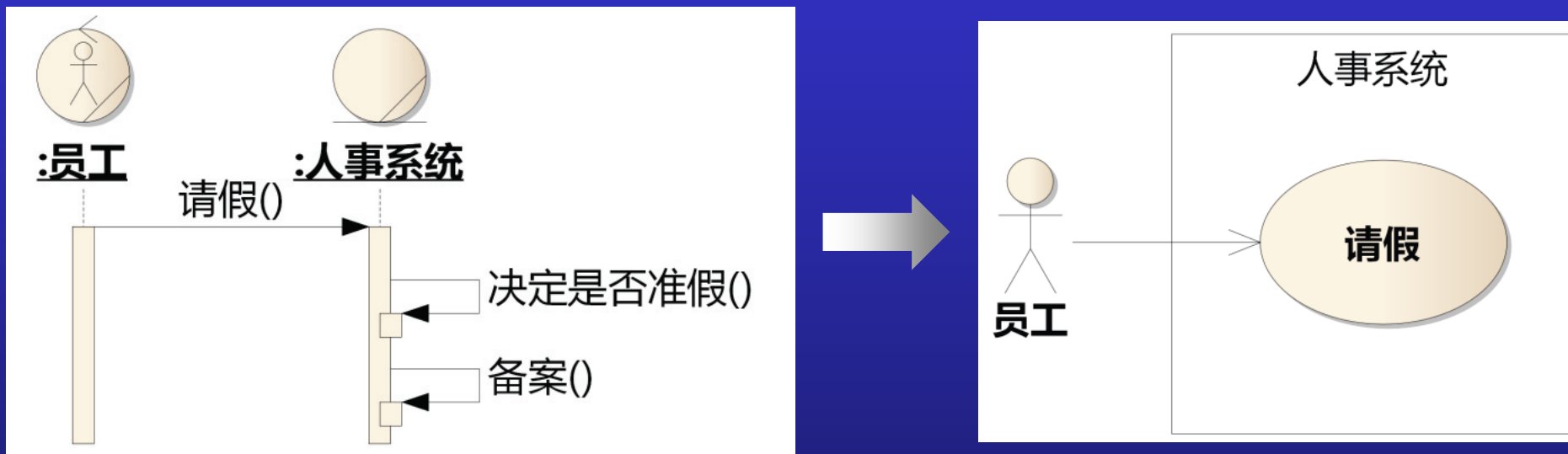
系统用例



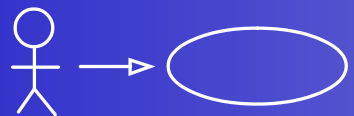
业务序列图帮助理清责任



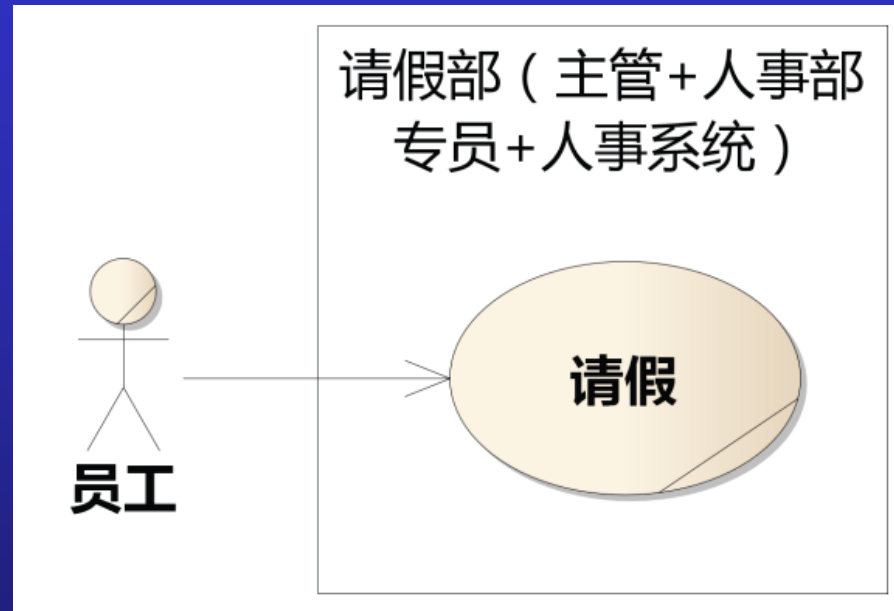
系统用例



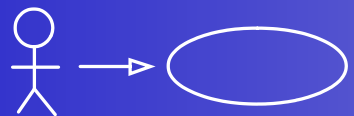
智能批假——请假变成了人事系统的用例



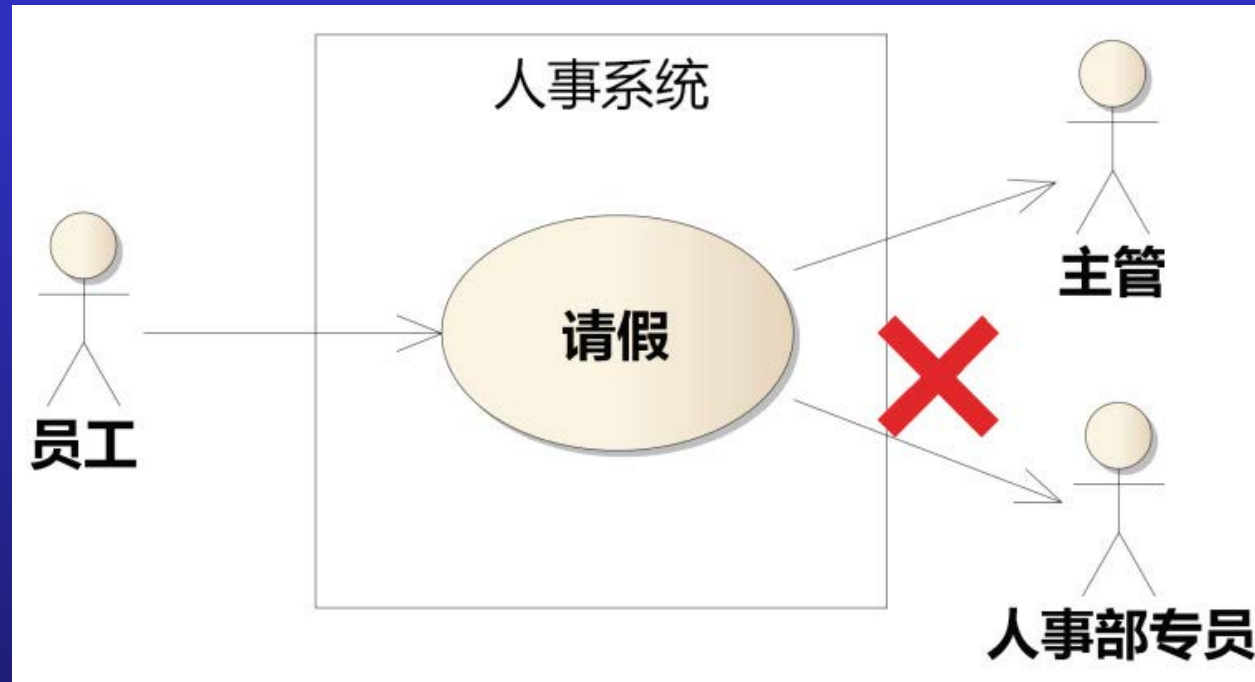
系统用例



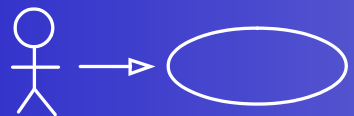
研究对象改变，用例也随之而变



系统用例



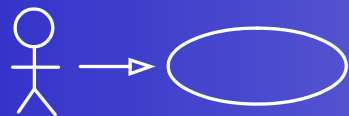
员工心中没有期望请到假才离开



系统用例

动词（+宾语）
↑ ↑
状语 定语

命名：动宾结构



建模 workflow

*业务建模

愿景

业务用例图

现状业务序列图

改进业务序列图

*需求

系统用例图

系统用例规约

*分析

分析类图

分析序列图

分析状态机图

*设计

建立数据层

精化业务层

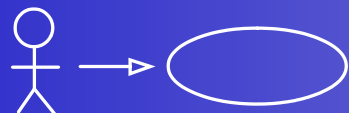
精化表示层

需求

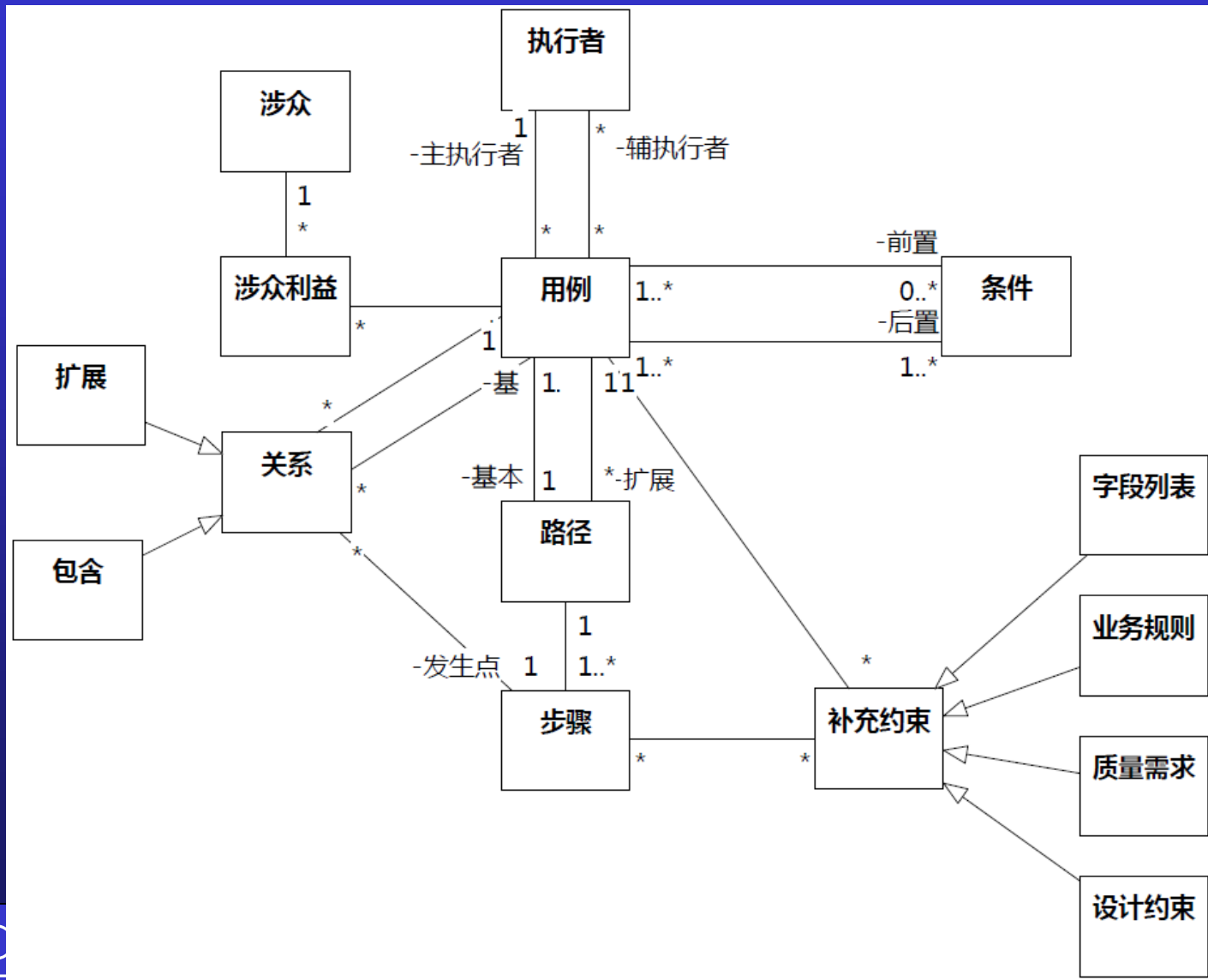
提升销售

设计

降低成本



用例规约



| 需求类别 | 用例规约中的对应 |
|------|--------------------------------|
| 功能需求 | 用例 路径 步骤 字段列表 业务规则 |
| 质量需求 | 质量需求 |
| 设计约束 | 设计约束 |

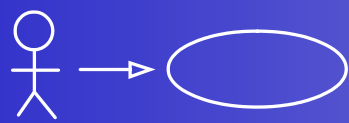
用例规约

- 用例编号：用例名
- 执行者
- 前置条件
- 后置条件
- 涉众利益
- 基本路径
 - 1..... ××××
 - 2.....××××
 - 3..... ××××
- 扩展
 - 2a. ××××：
 - 2a1.... ××××
- 字段列表
- 业务规则
- 质量需求
- 设计约束

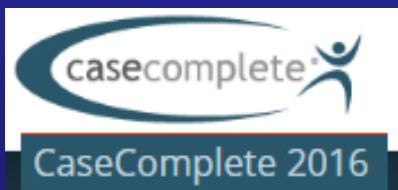
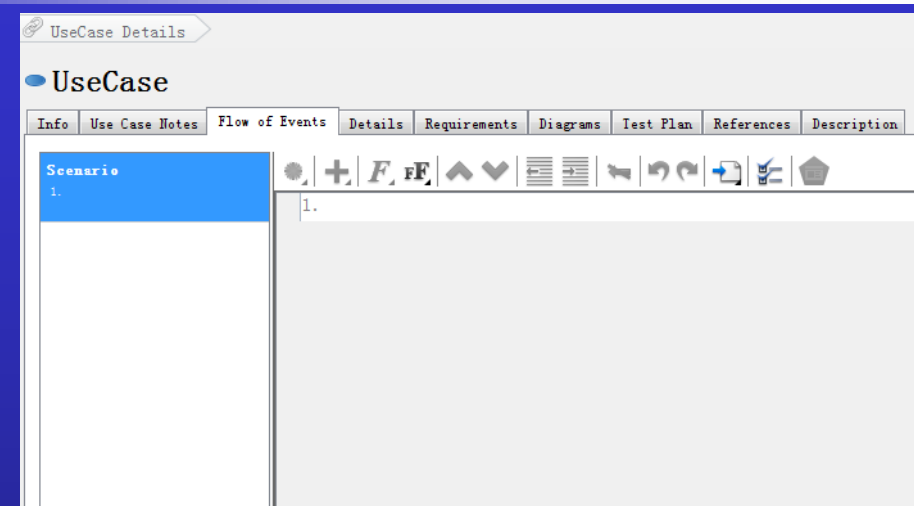
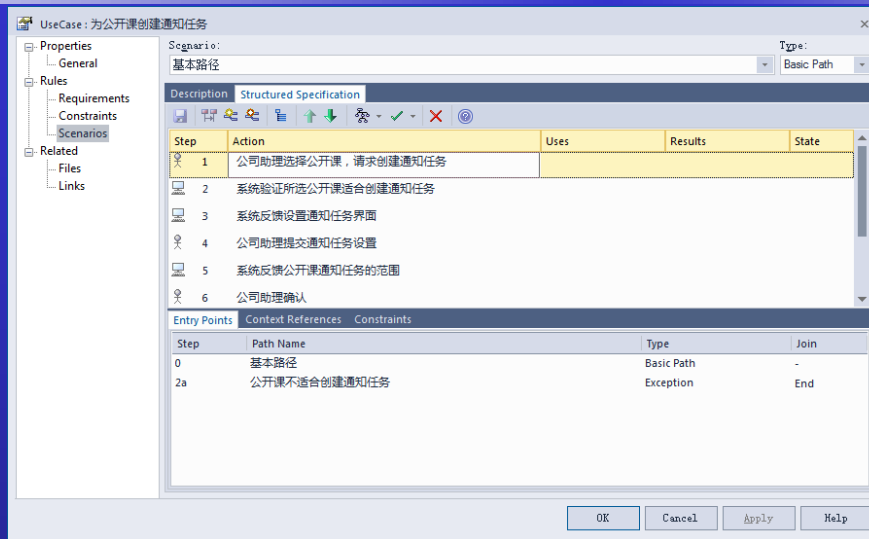
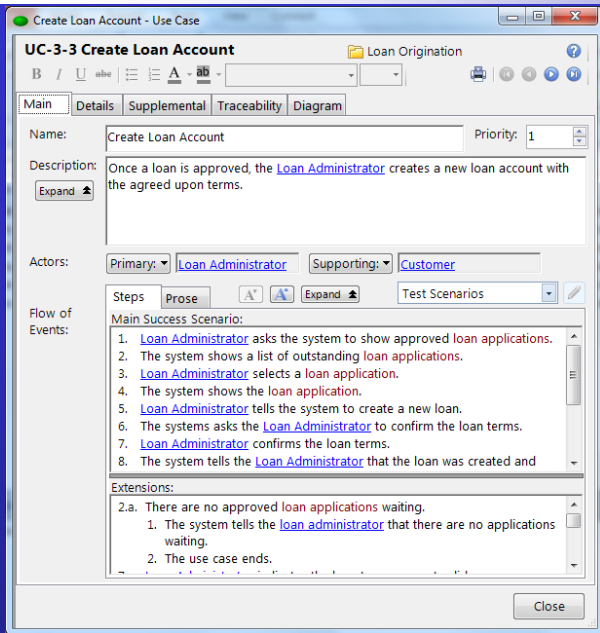
- 用例编号：用例名
- 执行者
- 前置条件
- 后置条件
- 涉众利益

| 步骤 | 基本路径 | 扩展路径 | 补充约束 |
|----|------|---------|------|
| 1 | ×××× | a. ×××× | 字段列表 |
| | | a1. | 业务规则 |
| | | ×××× | 质量需求 |
| | | | 设计约束 |
| 2 | ×××× | | |

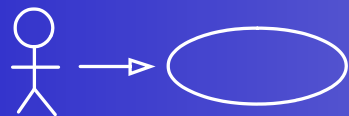
线性式和表格式



用例规约



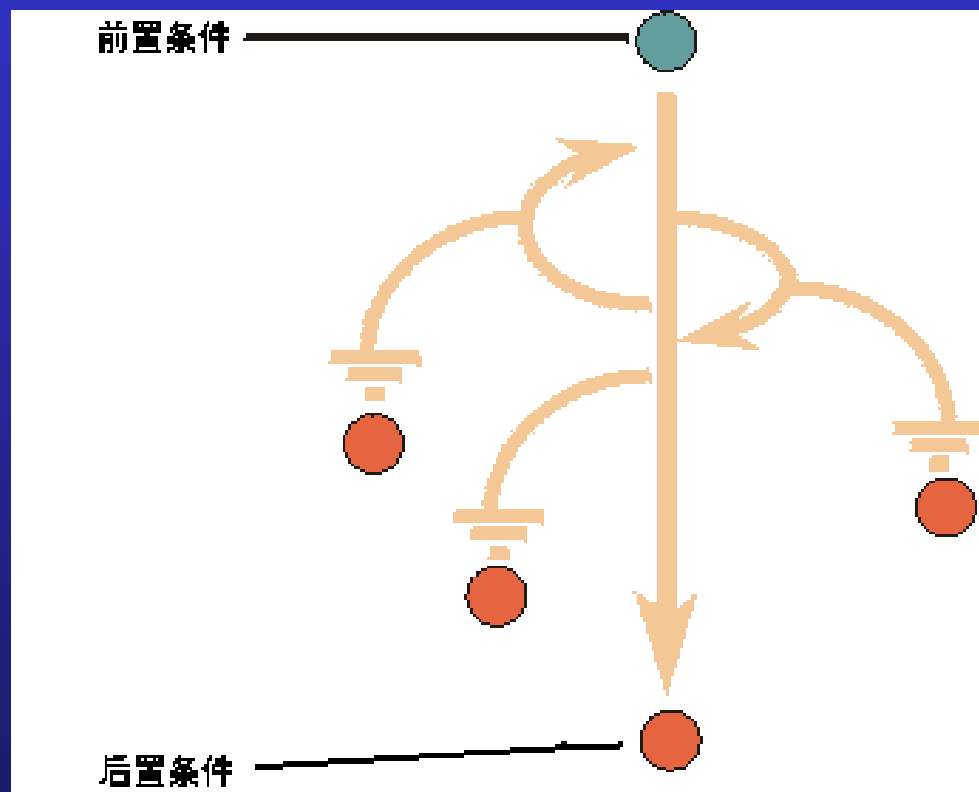
工具



<http://www.umlchina.com>

用例规约

——前置、后置条件



用例开始前，系统需要满足的约束

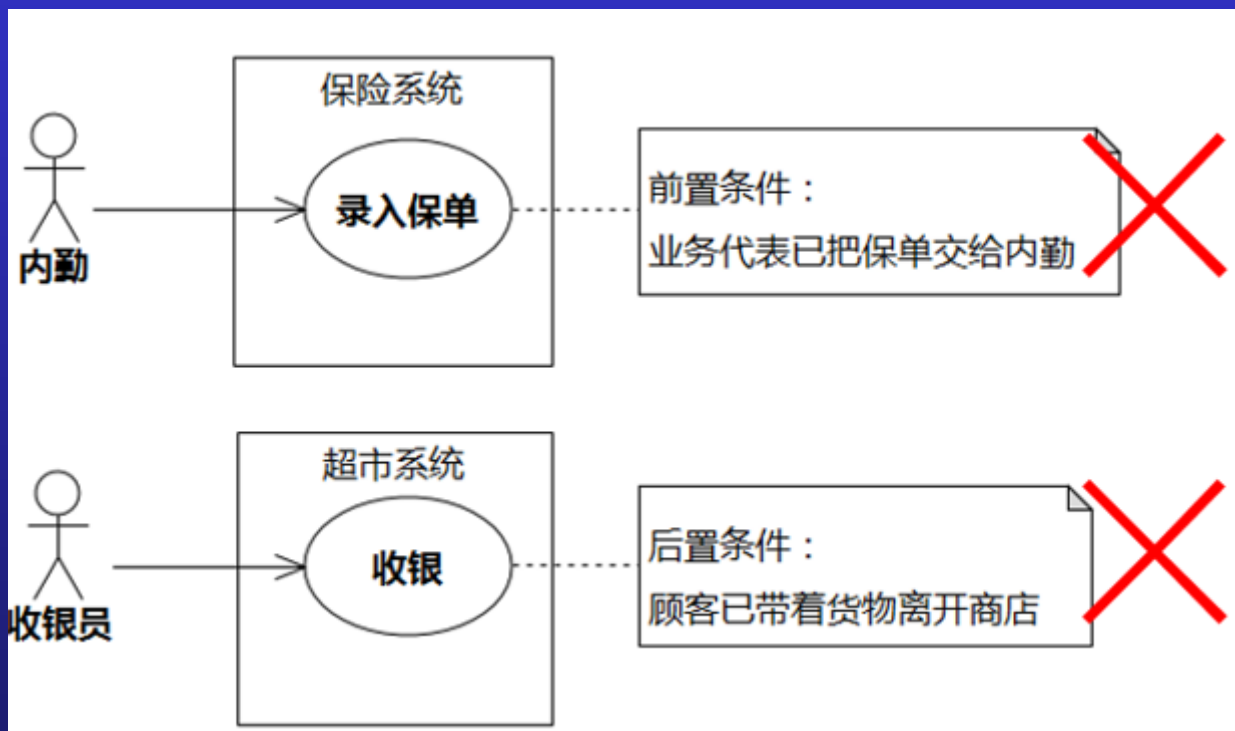
形式：必须是系统能检测的
内容：不满足会伤害涉众的利益

用例结束后，系统需要满足的约束

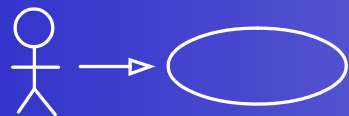


用例规约

——前置、后置条件

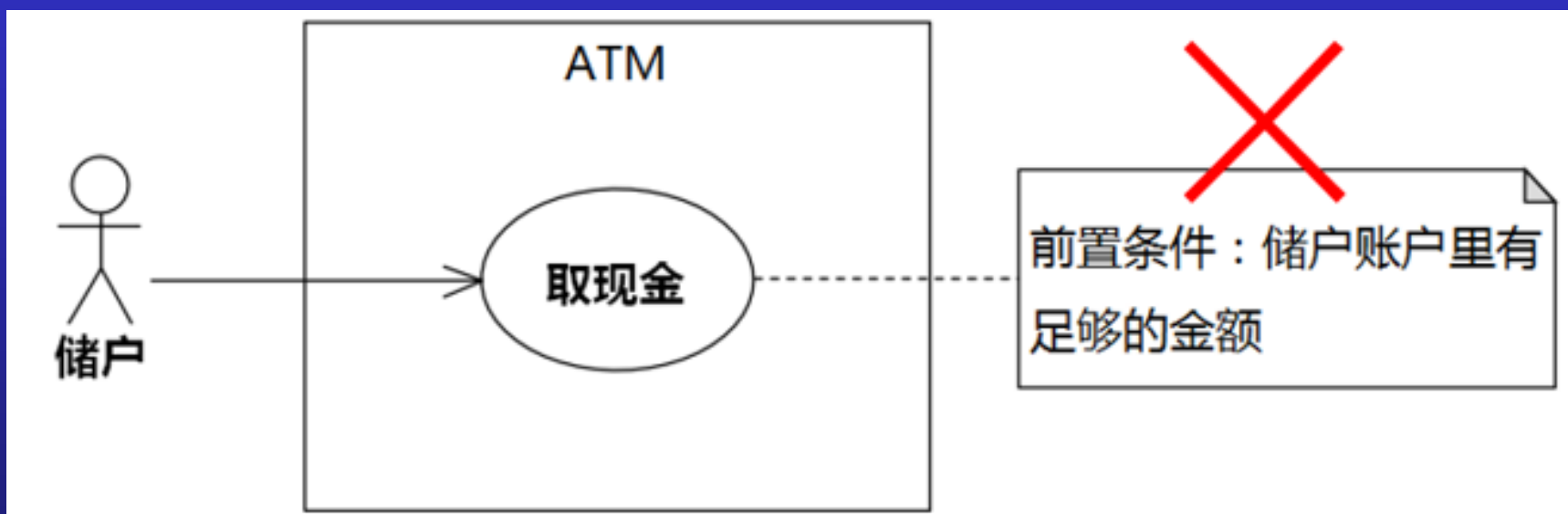


必须是系统能检测到的



用例规约

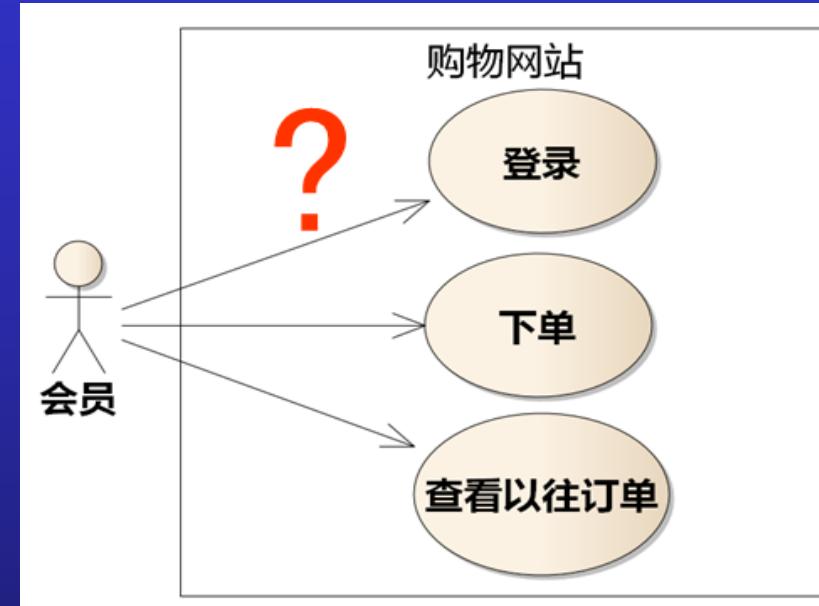
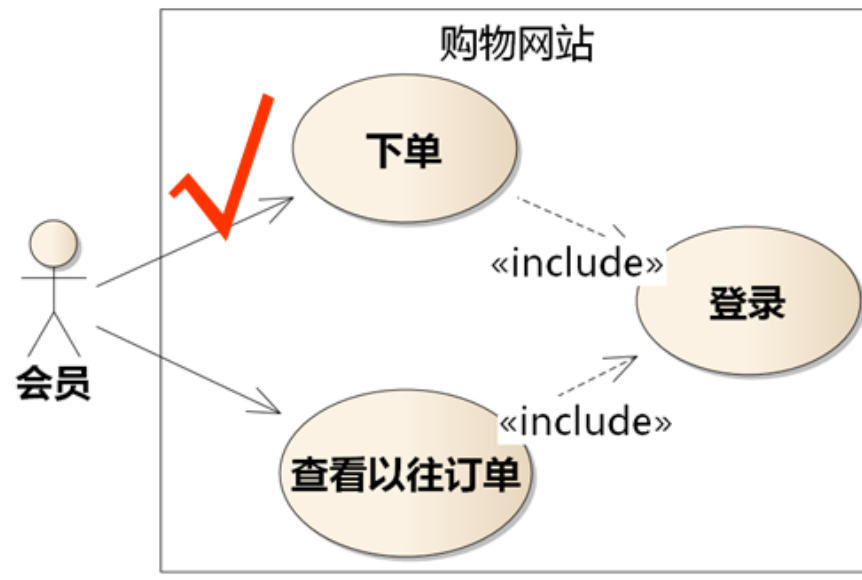
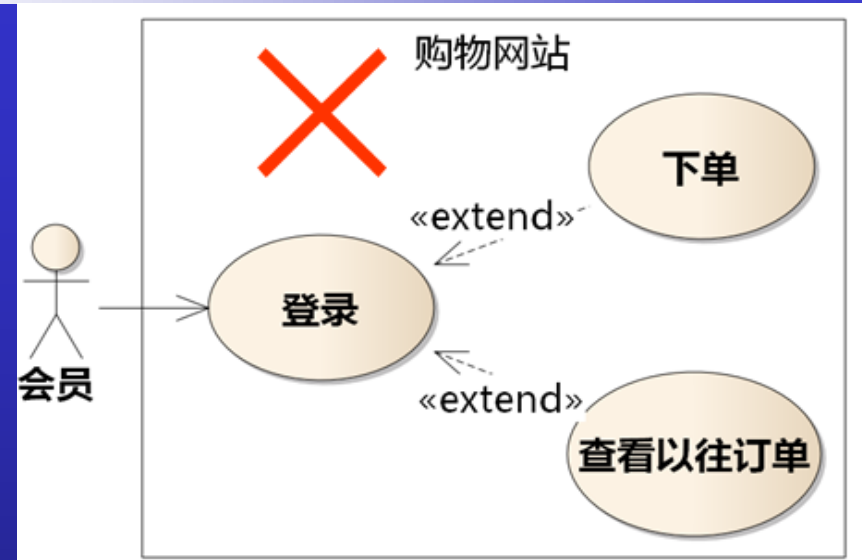
——前置、后置条件



前置条件必须是用例开始前系统能检测到的

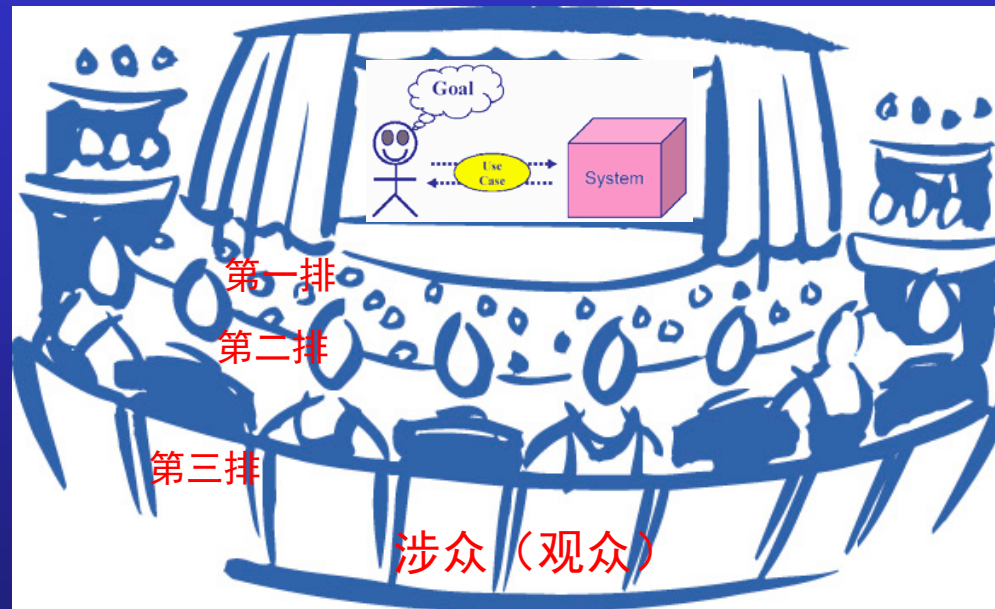


用例规约

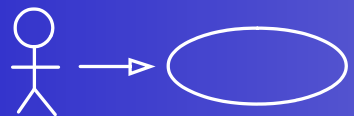


用例规约

——涉众利益



演员和观众



用例规约

——涉众利益

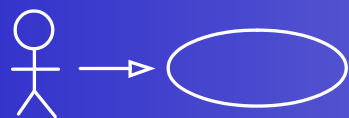


同样是“取钱”：

为什么家里的抽屉不用密码，取款机要用？

为什么取了钱以后要写“系统扣除账户金额”？

同样的目标，不同的过程...



用例规约

——涉众利益

涉众利益

储户—希望操作方便；希望24小时服务；担心权益受损。

银行负责人—希望安全；希望节约运营成本。

基本路径

1. 储户提交账户信息
2. 系统验证账户信息合法
3. 系统提示输入密码
4. 储户输入密码
5. 系统验证密码合法、正确
6. 系统提示输入取款金额
7. 储户输入取款金额

8. 系统验证取款金额合法

9. 系统记录取款信息，更新账户信息

.....

业务规则

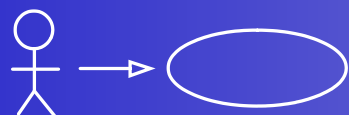
5. 密码为6位数字

8. 取款金额应为100元的倍数；取款金额应少于账户余额；单次取款金额不超过3000元；当日取款金额不超过20000元

设计约束

1. 通过磁条卡或芯片卡提交账户信息

用例平衡涉众之间的利益



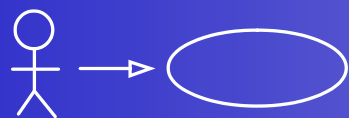
用例规约

——涉众利益

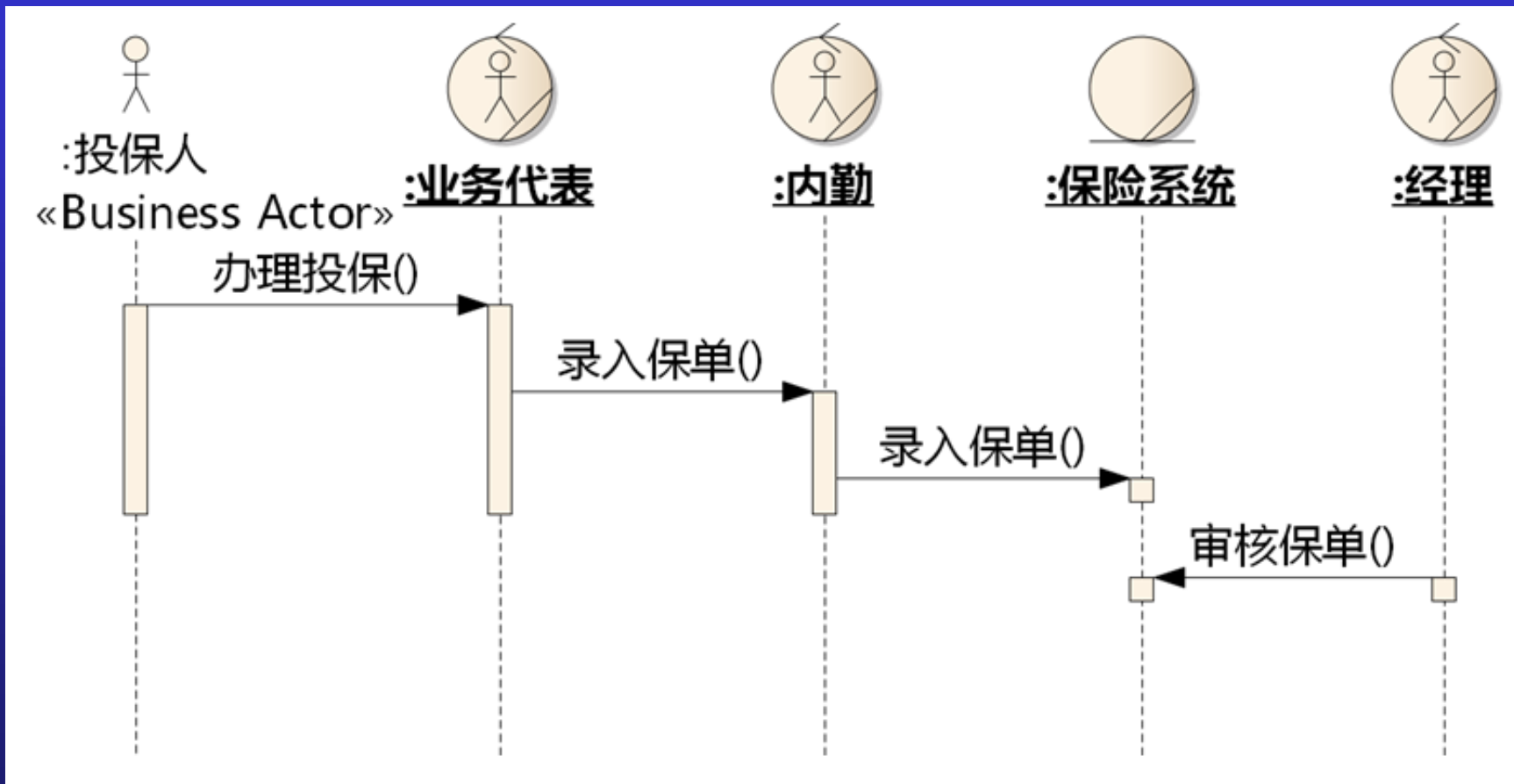


醉酒法+互害法

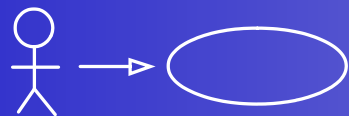
- 人类执行者
- 上游
- 下游
- 信息的主人
- ...



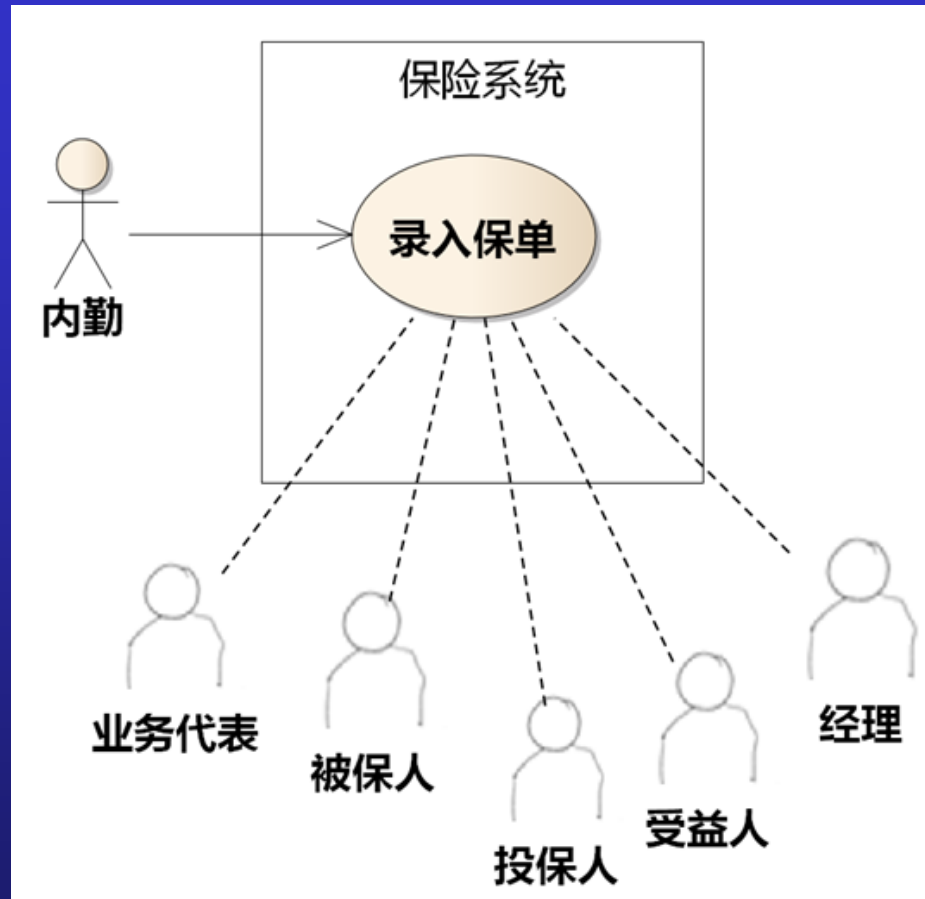
用例规约



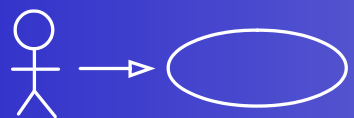
业务建模对识别涉众的作用



用例规约



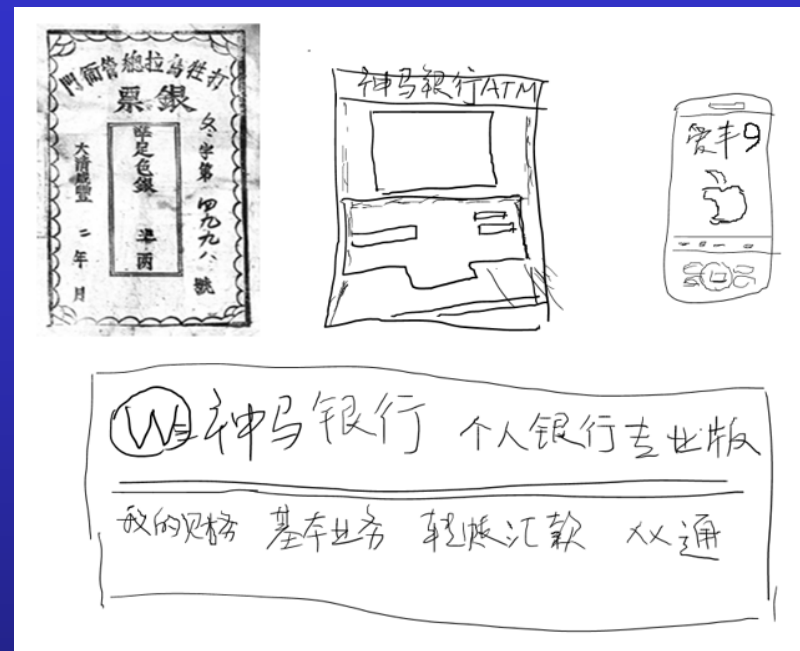
业务建模对识别涉众的作用



用例规约

储户——希望方便；担心权益受损

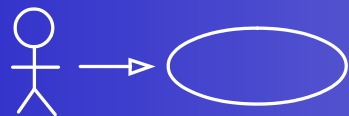
银行负责人——希望安全；希望节约运营成本



不变

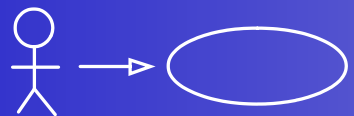
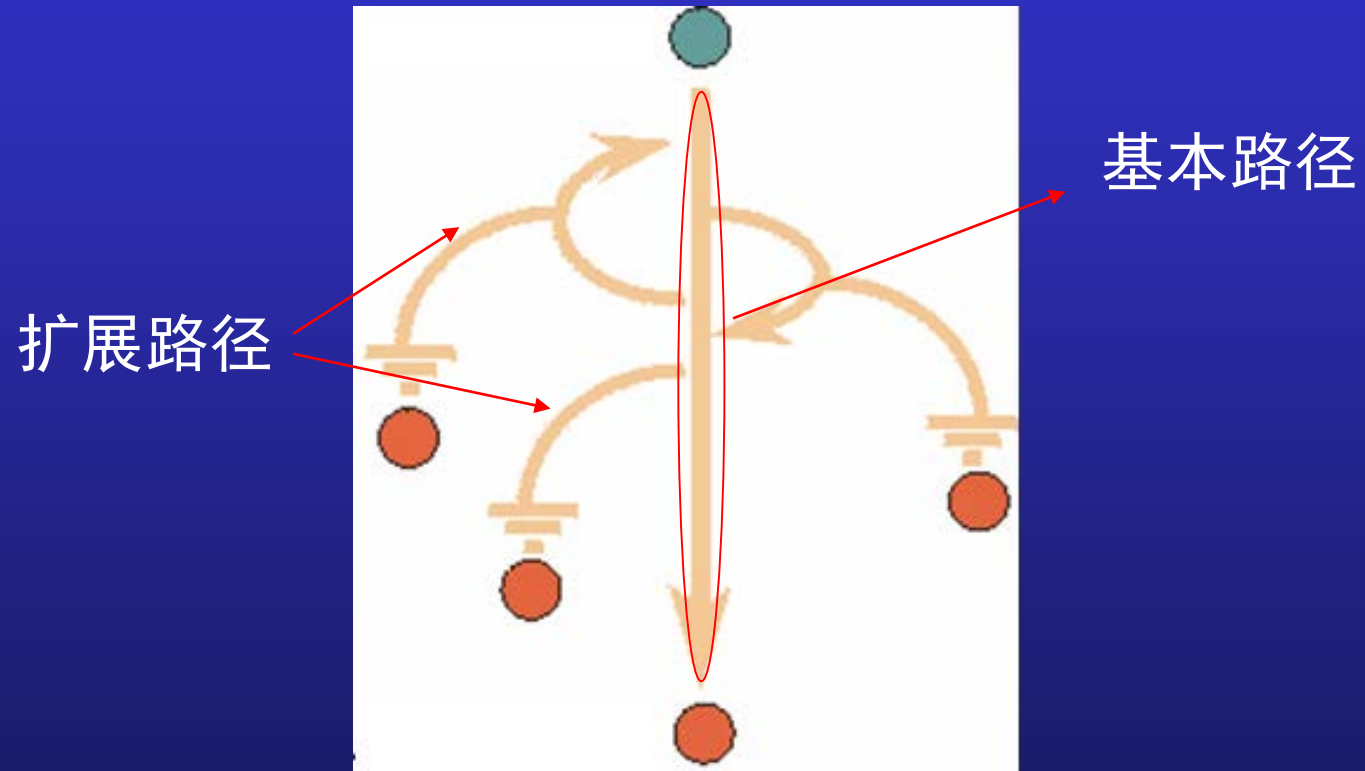
易变

涉众利益是可以积累的财富



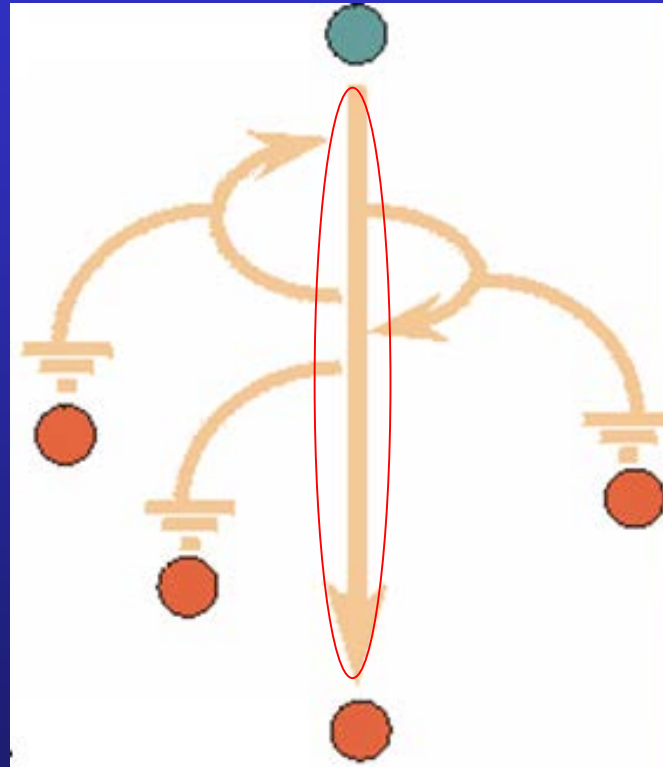
用例规约

——路径步骤



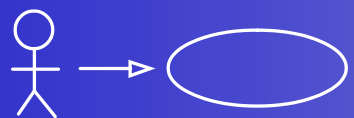
用例规约

——路径步骤：基本路径



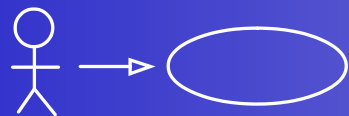
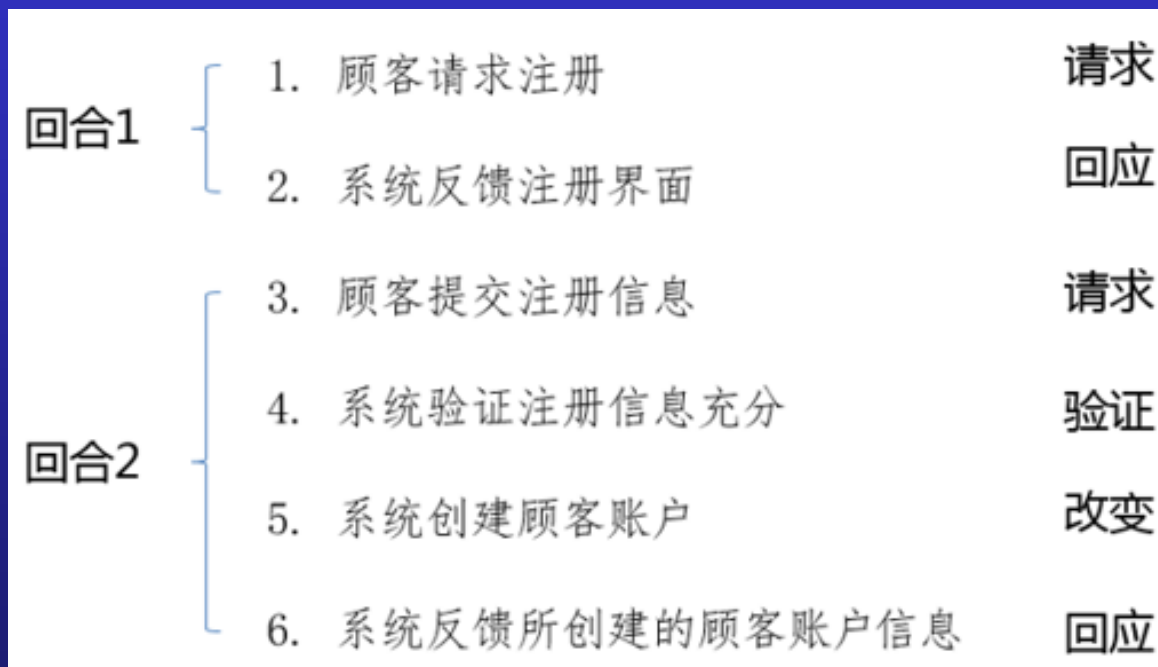
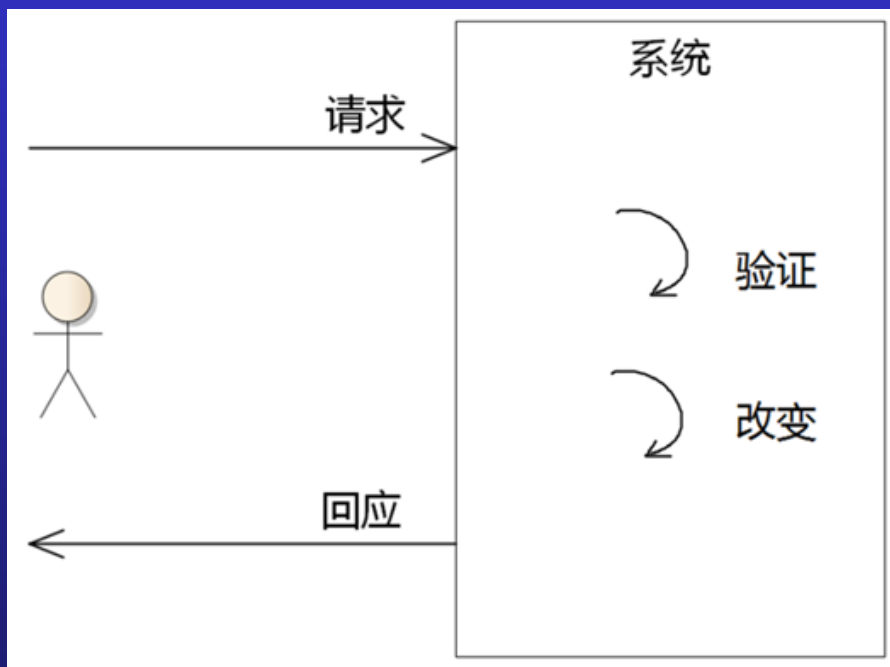
不要忘记初衷！

凸现用例核心价值的路径



用例规约

——路径步骤



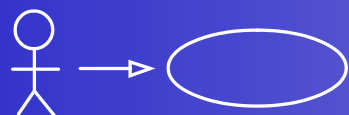
用例规约

——路径步骤

- 伊布从瓦伦西亚处得到传球，舒梅切尔扑救伊布的射门…
✗
- 瓦伦西亚传球，伊布射门，舒梅切尔扑救…
✓



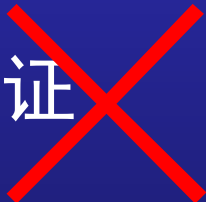



使用主动语句——理清责任

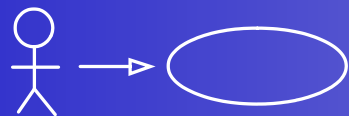


用例规约

——路径步骤


- 系统从会员处获取用户名和密码 
- 会员提交用户名和密码 
- 用户名和密码被验证 
- 系统验证用户名和密码 


主动语句



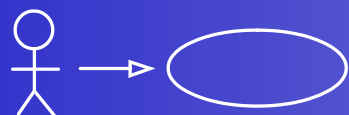
用例规约

——路径步骤

- 会员保存订单？存在自己肚子里？
- 会员提交订单信息
- 系统保存订单

- 会员查询商品？会员在自己肚子里面查？
- 会员提交查询条件
- 系统查询商品
- 系统反馈查询结果

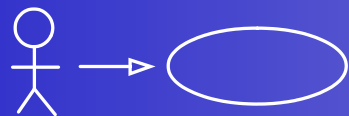
主动语句



用例规约

——路径步骤

| 错误示例 | 说明 |
|---------------------|-----------------------------|
| 会员 打开 系统 | 用手撕开？ |
| 会员 进入 系统 | 像黑客帝国一样进去？ |
| 系统 自动 计算订单总价 | 系统当然是自动的 |
| 会员 手动 输入订单信息 | 会员当然是手动的 |
| 会员提交订单信息 给系统 | 冗余 |
| 系统 获得 订单信息 | 从哪获得？ |
| 系统 获取 商品单价 | 其实想的是“读取数据库里商品表的单价字段”，系统的设计 |



用例规约

——路径步骤

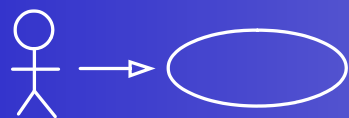
- 执行者 × × × × ×
- 系统 × × × × ×

➤ 代理、客户端、×子系统

如果害怕丢东西
丢的可能是质量需求



主语只能是执行者或系统

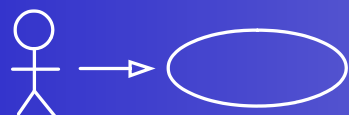


用例规约

——路径步骤

| 系统 | 涉众 | 描述 | 合格吗 |
|--------|------|-------------------------------|-----|
| 零件采购系统 | 采购员 | 系统建立连接，打开连接，执行SQL语句，从“零件”表查询… | 不合格 |
| 零件采购系统 | 采购员 | 系统按照查询条件搜索零件 | 合格 |
| 数据访问框架 | 开发人员 | 系统建立连接，打开连接，执行SQL语句 | 合格 |

使用核心域词汇（说人话）

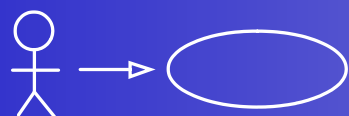


用例规约

——路径步骤

| 错误示例 | 更正 |
|--|----------------------------|
| 系统显示订单信息 | 系统反馈订单信息 |
| 系统反馈查询到的商品列表 | 系统反馈查询到的商品 |
| 会员拖动商品到购物篮，勾选优惠券，点击结算按钮 | 会员输入商品和优惠券信息，请求结算 |
| 顾客填写用户名 系统验证用户名未被使用 顾客填写密码 系统验证密码合法 | 顾客提交用户名和密码 系统验证用户名和密码合法 |

不要涉及界面组件和交互



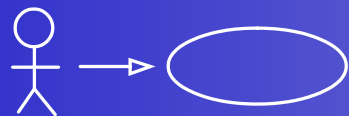
用例规约

- 用例（取款）
 - 路径（正常取款）
 - 步骤（系统验证取款金额合法）
 - 补充约束（取款金额必须为100元的倍数）

低精度，稳定

高精度，不稳定

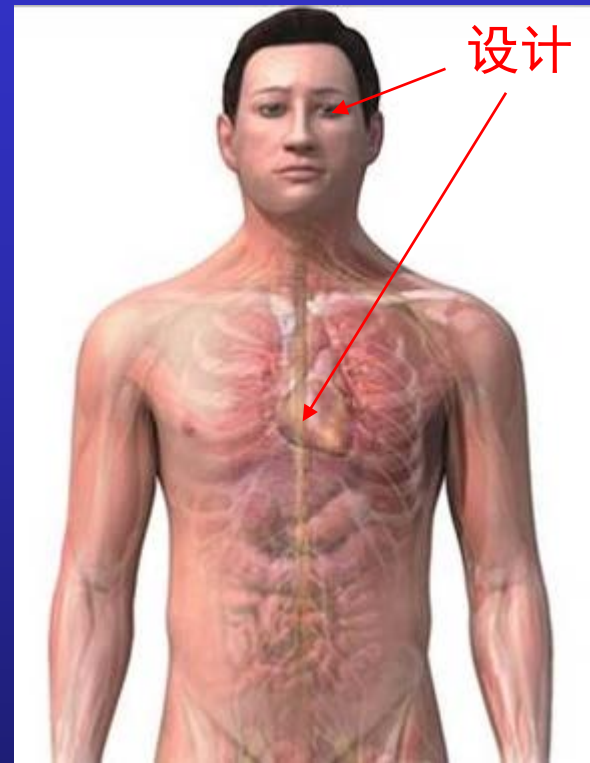
不同级别的需求



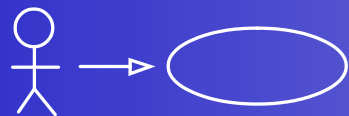
用例规约

——路径步骤

- 看得见摸得着？
- 涉众是否在意！
- 许多需求人员唯一的招数：界面原型



不要涉及界面组件

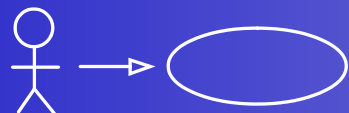


用例规约

——路径步骤

- 商品的价格由供求关系决定
- 这样行吗 vs 不这样行吗
- 洋洋得意的“无二义需求”
- 医生无力诊断病情，却把药描述得头头是道

判断需求的标准：不这样行吗



用例规约

——路径步骤

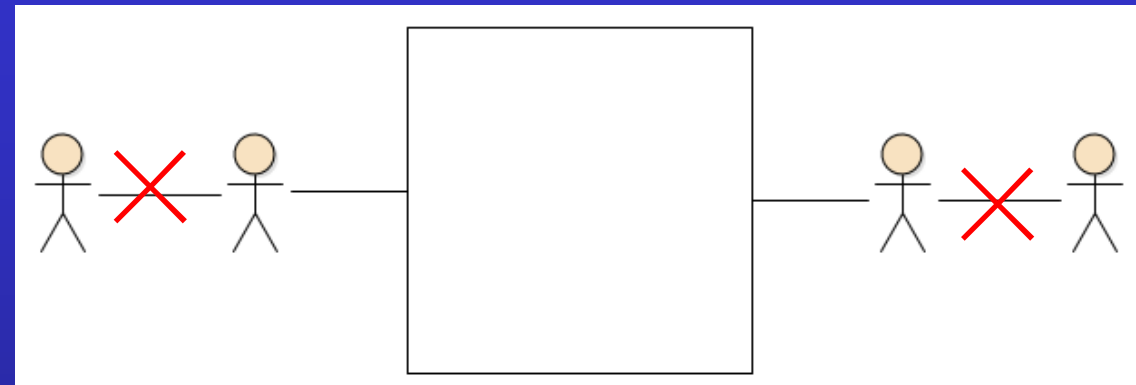
➤ ~~科员根据纸质房屋所有权登记表~~
录入房屋登记信息

➤ ~~经理审查支票~~

➤ 经理确认支票通过审查

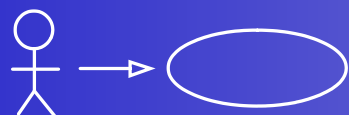
➤ 系统请求AFIS系统比对指纹

➤ ~~AFIS系统比对指纹~~



模糊了系统的契约
抹杀进一步改进的可能

不要写系统不能负责的事情

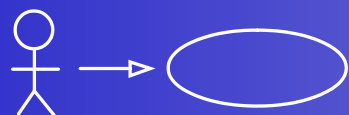


用例规约

系统请求 × × 做某事
(不用写 × × 系统做某事)

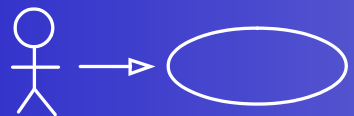
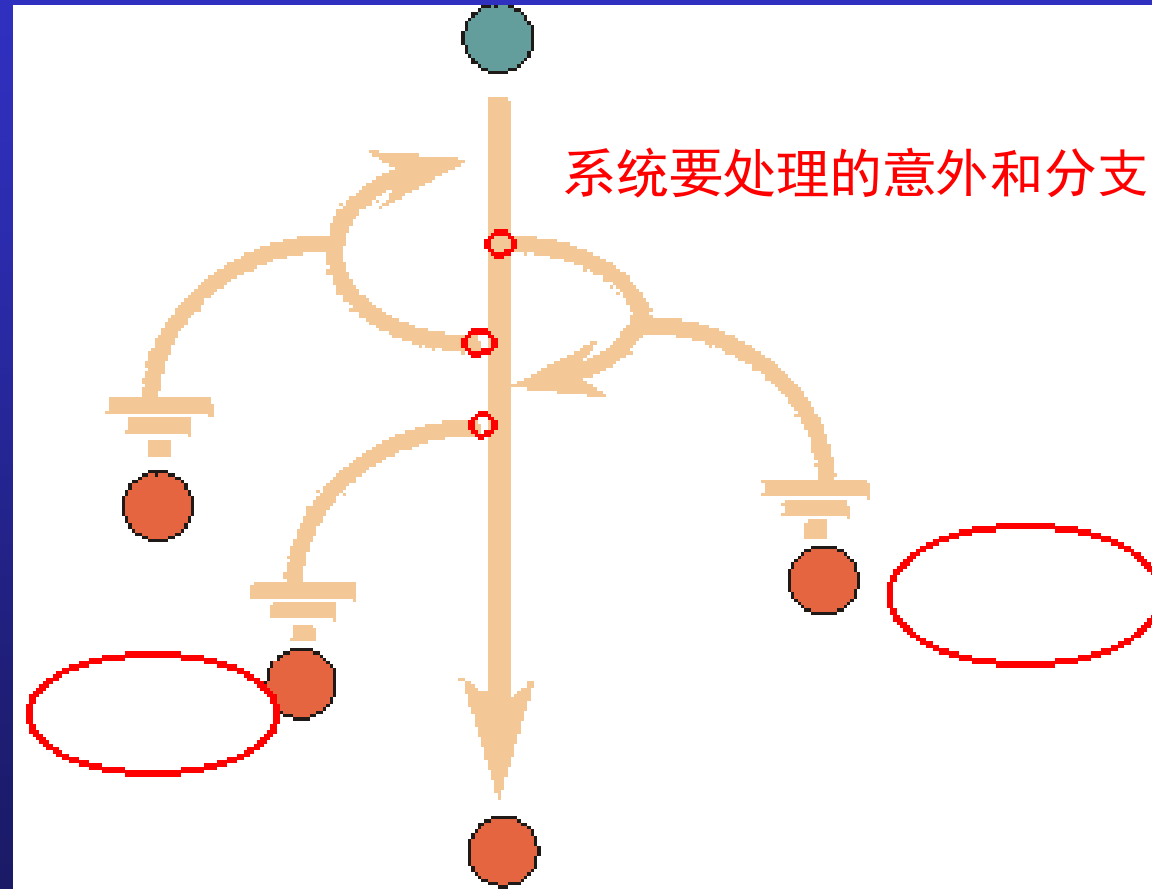
- 3、→ 痕检员 编辑提交 指纹特征信息、案件描述信息。
- 4、→ 痕检员提交查找任务。
- 5、→ 系统将 案件信息发送给请求 AFIS 系统 进行查找比对。

辅助执行者



用例规约

——路径步骤：扩展



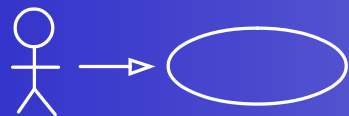
用例规约

——路径步骤：扩展

- 执行者的选择（慎用！）
- 系统验证
- 步骤失败
- ○ ○ ○ ○ ○ ○

必须是系统能
感知和要处理的

识别扩展点思路



用例规约

4. 用户选择如下操作：

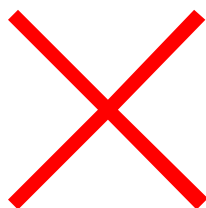
不让利

单条商品折扣

单条商品折让

削价

5. 系统计算单条商品价格。



3. 评议专家可以选择以下动作：

给出“5分”的评分

给出“4分”的评分

给出“3分”的评分

给出“2分”的评分

给出“1分”的评分

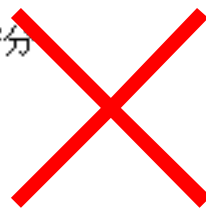
给出“0分”的评分

扩展：

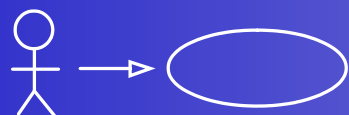
3a. 评议专家给 5 分：

3a1. 评议专家给出“5分”的评分

3a2. 系统保存该学员本题分数为 5



扩展！ = 选项，看交互行为变化



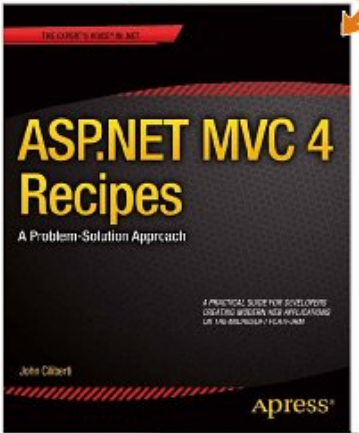
用例规约

amazon [Try Prime](#) [Your Amazon.com](#) [Today's Deals](#) [Gift Cards](#) [Sell](#) [Help](#)

Shop by Department [Search](#) [Kindle Store](#)

[Buy a Kindle](#) [Kindle eBooks](#) [Advanced Search](#) [Daily Deals](#) [Free Reading Apps](#) [Kindle Singles](#) [Newsstand](#) [Access](#)

Start reading *ASP.NET MVC 4 Recipes* on your Kindle **in under a minute**



Click to **LOOK INSIDE!**

ASP.NET MVC 4 Recipes
A Problem-Solution Approach
John Ciliberti
Apress®

kindle edition

Click to open expanded view

[Share your own customer images](#)

ASP.NET MVC 4 Recipes: A Problem-Solution Approach, 2nd Edition
[John Ciliberti](#) (Author)
★★★★☆ (10 customer reviews)

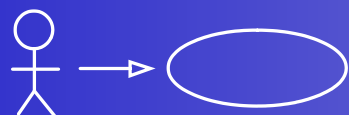
Digital List Price: ~~\$39.99~~ [What's this?](#)
Print List Price: ~~\$49.99~~
Kindle Price: **\$22.99**
You Save: **\$27.00 (54%)**

- Length: 627 pages (Contains Real Page Numbers)
- Don't have a Kindle? [Get your Kindle here.](#)

| Formats | Amazon Price | New from | Used from |
|----------------|----------------|----------|-----------|
| Kindle Edition | \$22.99 | -- | -- |
| Paperback | \$37.91 | \$32.79 | \$33.44 |

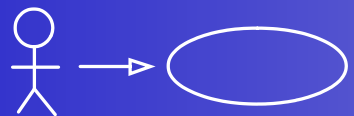
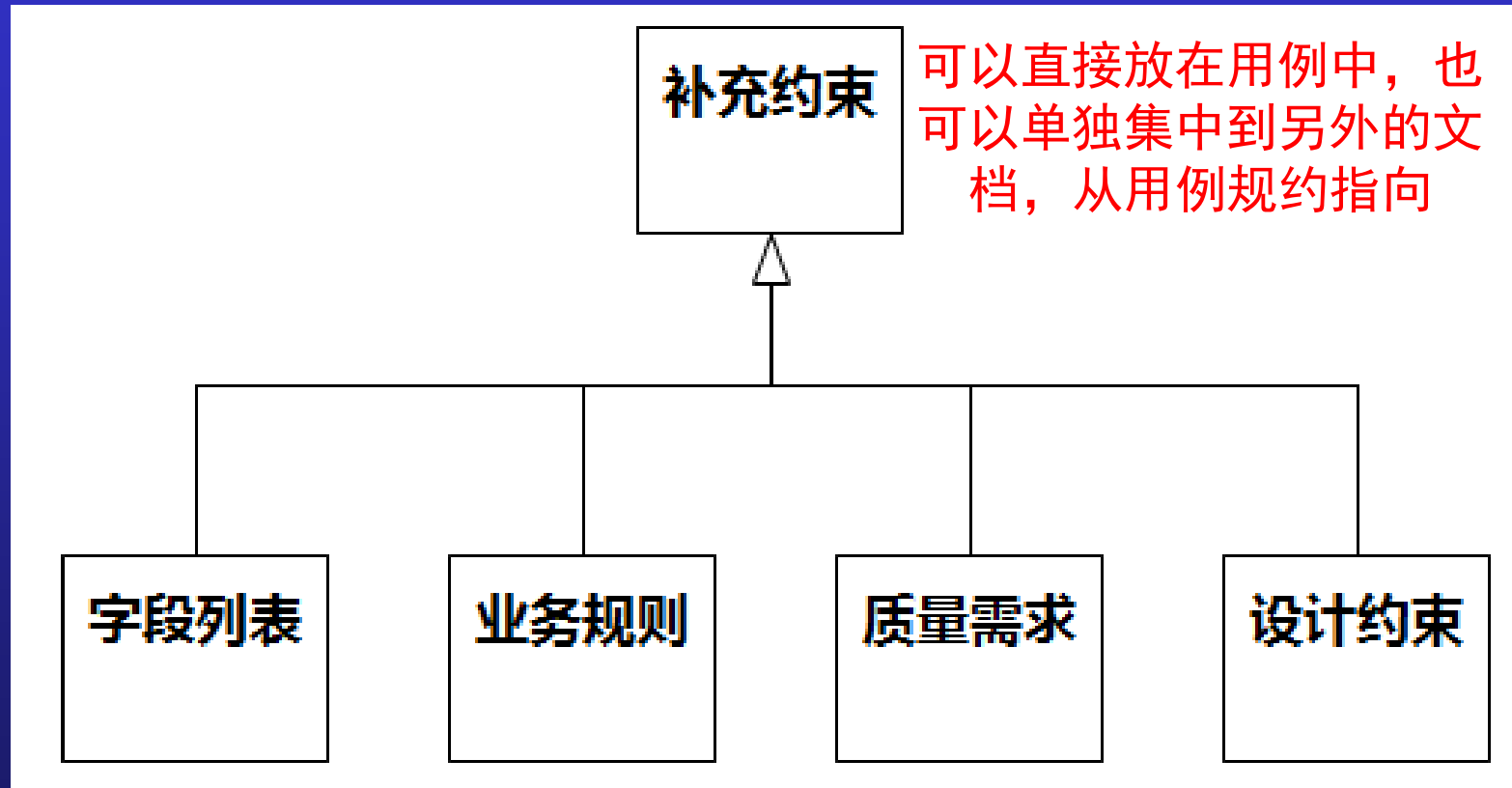
[digital](#) [Shop the New Digital Design Bookstore](#)

扩展！ = 链接



用例规约

——补充约束

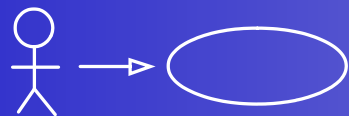


用例规约

——补充约束：字段列表

- + → 数据序列
- [] → 可选项
- { } * → 多个
- { | | | } → 可能取值
- A=B → 把B的结构赋给A

可以用自然语言，也可以用表达式



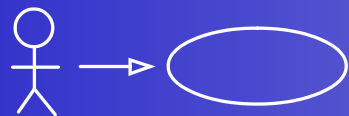
用例规约

——补充约束：字段列表

- 注册信息=公司名+联系人+电话+{联系地址}*
- 联系地址=州+城市+街道+邮编
- 保存信息=注册信息+注册时间
- 客房状态={空闲|已预定|占用|维修中}

多细？
涉众共识！

用表达式表示

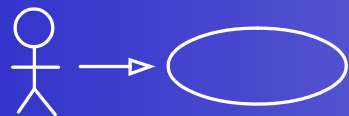


用例规约

——补充约束：字段列表

- 不同于数据模型——只是一部分
 - 可以用E/R图或业务对象图作为辅助说明，但不宜直接作为需求
- 不等于数据字典——容易过早把时间花在细节上
 - 一开始好像做了很多事情，其实却回避了困难的业务问题

注意



用例规约

——补充约束：业务规则

➤ 推理

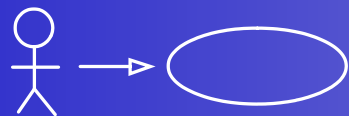
- 如果过了计划中的交付日期，货物还没有送到，即为“未按时送货”

➤ 约束

- 合同的总金额不能超出买方的信用额度

➤

各种业务规则

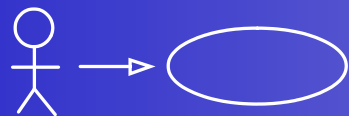


用例规约

——补充约束：业务规则

- 文字说明
- 决策表
- 行业上适用的任何方式
-

业务规则的各种表示方法



用例规约

——补充约束：业务规则

| | | | | | | |
|----|--------------|---|---|---|---|---|
| 条件 | 1. 双人房用作单人房 | Y | N | Y | | |
| | 2. 家庭额外客房 | N | Y | Y | | |
| | 3. 立即入住 | | | | Y | Y |
| | 4. 18:00 之前 | | | | Y | Y |
| | 5. 预订率低于 50% | | | | Y | Y |
| | 6. 晴天 | | | | N | Y |
| 行为 | a. 25% | √ | | | | |
| | b. 10% | | √ | | | |
| | c. 立即入住 25% | | | | | √ |
| | d. 立即入住 0% | | | | √ | |

决策表



用例规约

——补充约束：业务规则

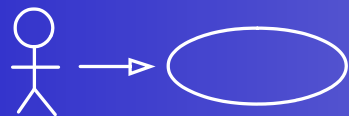
➤ 系统将语音输入翻译为文字

➤ 采用××识别算法

➤ 背景噪音强度为××的情况下，识别率应在××以上

警惕误把设计当成规则

业务规则！ = 实现算法

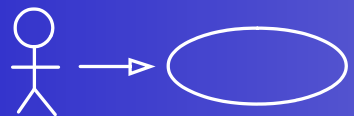


用例规约

——补充约束：质量需求

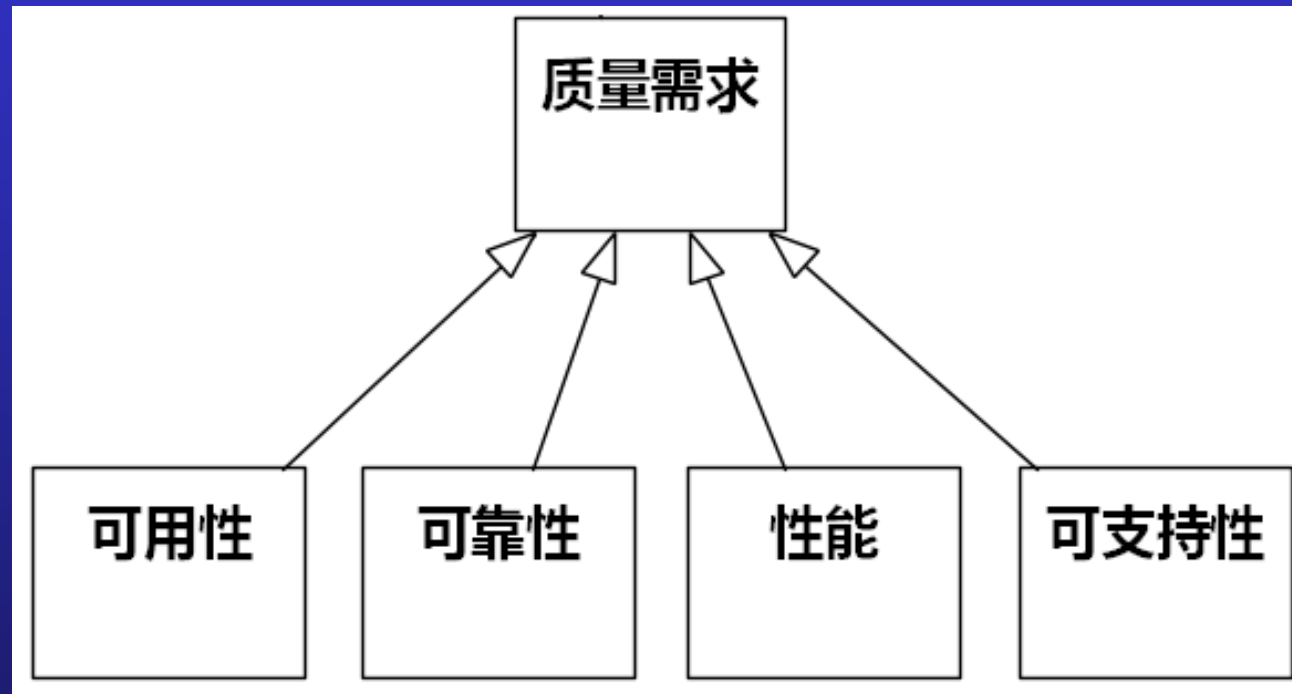


激烈竞争的决胜点

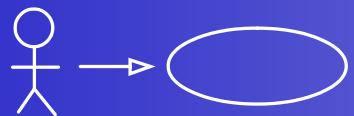


用例规约

——补充约束：质量需求



URPS

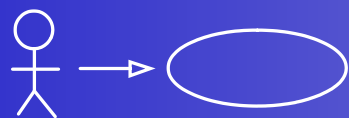


用例规约

——补充约束：质量需求

- 声音：分贝
- 光照强度：流明
- 漂亮：三围、脸型
- 方便：操作次数？操作时间？单手操作？

关键词：度量——万事皆可度量

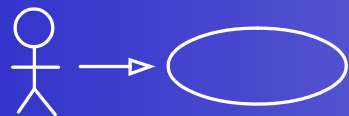


用例规约

——补充约束：质量需求：可用性

- 系统没有按程序员的意图工作→程序错误
- 系统无法执行一项任务→功能需求遗漏
- 系统能按照程序员意图工作，并且支持任务→但用户仍然不知道如何使用系统执行任务或者不喜欢使用系统执行任务→可用性问题

“可用性”！=用例



用例规约

——补充约束：质量需求：可用性

➤ 系统应易于使用



➤ 人事专员第一次使用时30分钟内能学会添加新员工（任务时间）

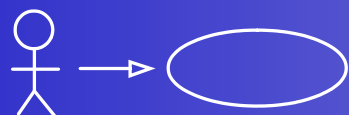
➤ 前台5次击键能完成客人入住服务，不需要使用鼠标（操作次数）



➤ 系统界面应如××附件所示的屏幕图像



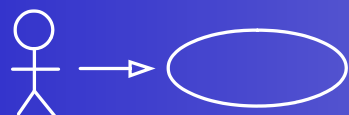
可用性需求的表达



用例规约

——补充约束：质量需求：可用性

- 90%的顾客应该能在6秒种内，通过少于四次动作，定位任意一件商品
→需求工程师的责任
- 什么样的交互界面适合满足这样的需求
→交互设计师的责任
- 如何用软件组件实现这样的交互界面
→程序员的责任

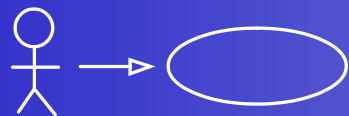


用例规约

——补充约束：质量需求：可靠性

- ❖ 系统应能防范磁盘故障（安全）
 - ❖ （对照）系统应采用冗余磁盘阵列
- ❖ 系统应保证收到的数据和发送的数据一致（完整）
- ❖ MTBF (Mean Time Between Failures)
- ❖ MTTR (Mean Time To Repair) (稳定)

各种可靠性需求

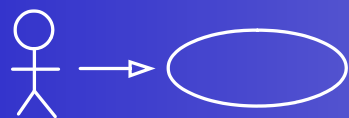


用例规约

——补充约束：质量需求：性能

- 系统应在0.5秒之内拍摄超速车的照片（速度）
- 系统应允许1000个用户同时使用（容量）
- 在标准工作负荷下，系统的CPU占用率应少于50%（能力）

各种性能需求

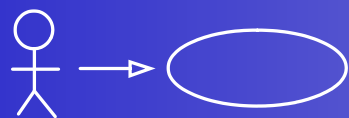


用例规约

——补充约束：质量需求：可支持性

- 95%的紧急错误应能在30工作小时内修复
- 在修复故障时，未修复的相关缺陷平均数应小于0.5
- 在两年内，以每功能点××的价格升级系统
- 升级新版本时，应保存所有系统设置和个人设置

各种可支持性需求



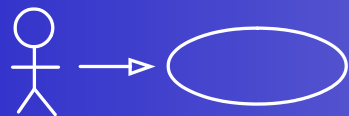
用例规约

系统应在3秒内向顾客反馈应收款项

Why?

系统在服务器端计算应收款项，然后把结果传回客户端？

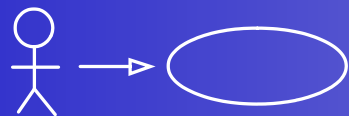
混入需求的设计——往往隐含质量需求



用例规约

——补充约束：设计约束

- 界面样式
- 报表
- 平台
- 语言
- 外系统接口……



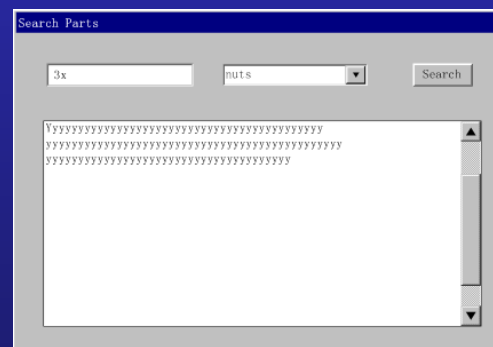
用例规约

——补充约束：设计约束

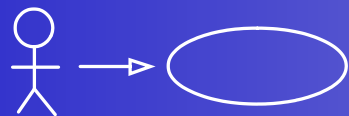
➤ 5. 系统显示订单明细

必须来自涉众！

➤ 5. 应采用附件××所示的屏幕格式.....



界面样式



用例规约

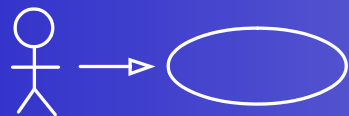
——补充约束：设计约束

➤ 5. 系统打印工资单

必须来自涉众!

➤ 5. 工资单的格式如附件××所示.....

报表



用例规约

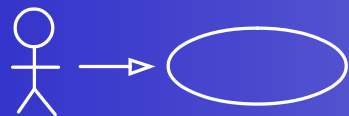
——补充约束：设计约束

由于×公司的IT人员有Oracle专长，而且目前已有的其他应用系统也使用Oracle数据库，所以系统应使用Oracle数据库。



必须来自涉众!

平台：一般针对整个系统



用例规约

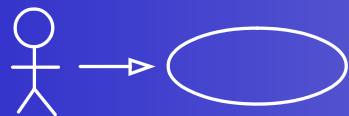
——补充约束：设计约束

➤ 5. 系统读取探测器读数

➤ 5. 通过RS-232接口读取

必须来自涉众！

技术接口



用例规约

——补充约束：设计约束

- 系统应采用三层架构方式搭建

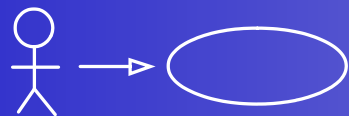


什么是涉众
能够理解和验证的？

- 系统应允许分布在各地的多个用户并发使用



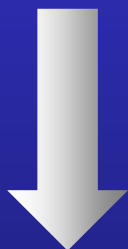
小心伪装成“设计约束”的“设计”



用例规约

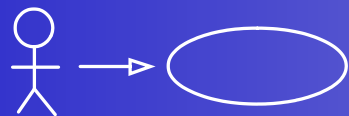
——补充约束：设计约束

- 录单界面应分为3个页面，每个页面填写完毕，?点击“下一步”，出现下一页面。



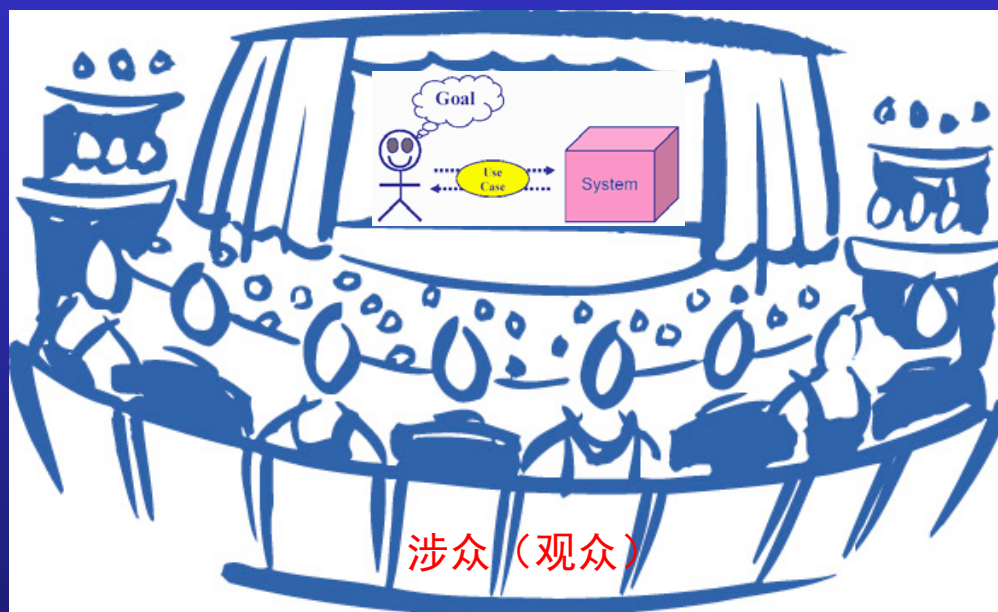
- 系统应在××秒内显示录单界面（质量需求）

小心伪装成“设计约束”的“设计”

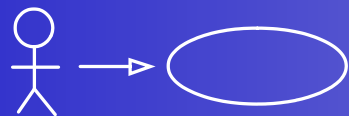


用例规约

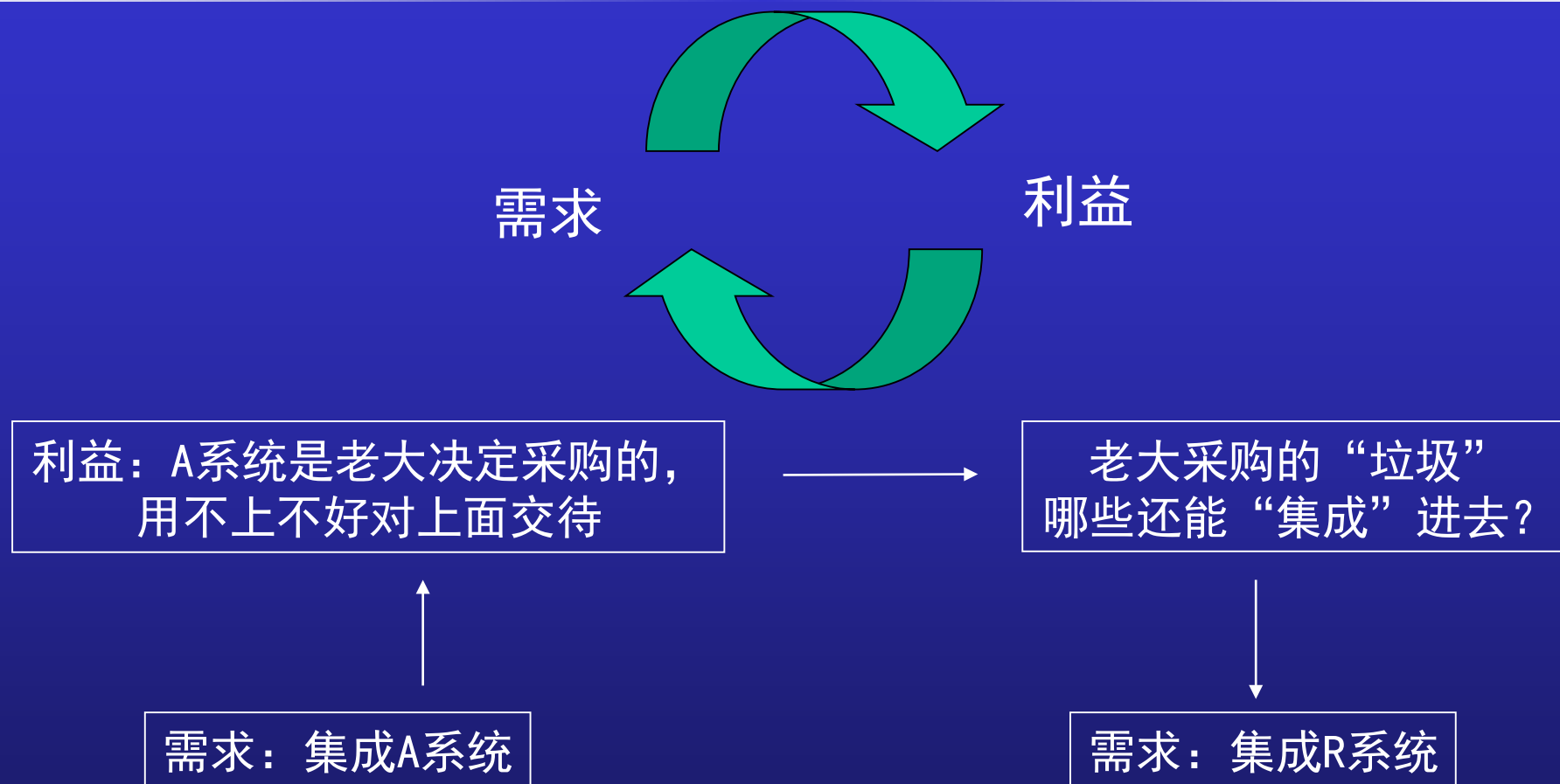
——最后，再看一遍！



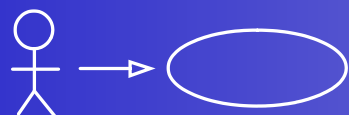
写对了吗？别忘了和涉众利益对照！



用例规约



涉众利益和需求



用例规约

➤ 涉众利益

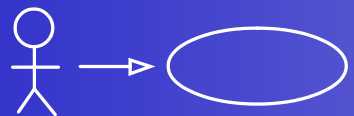
CEO——希望佣金能灵活按照公司佣金政策调整准确计算；
经纪——担心少算；担心拖延；

➤ 设计约束

使用C#和.NET开发



涉众利益和需求

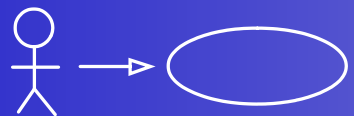


用例规约

- 每个用例都要写规约吗？
- 用例规约一定要规范完整吗？
- 心里有观众胜过整齐的剧本

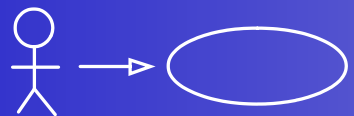


不浪费原则

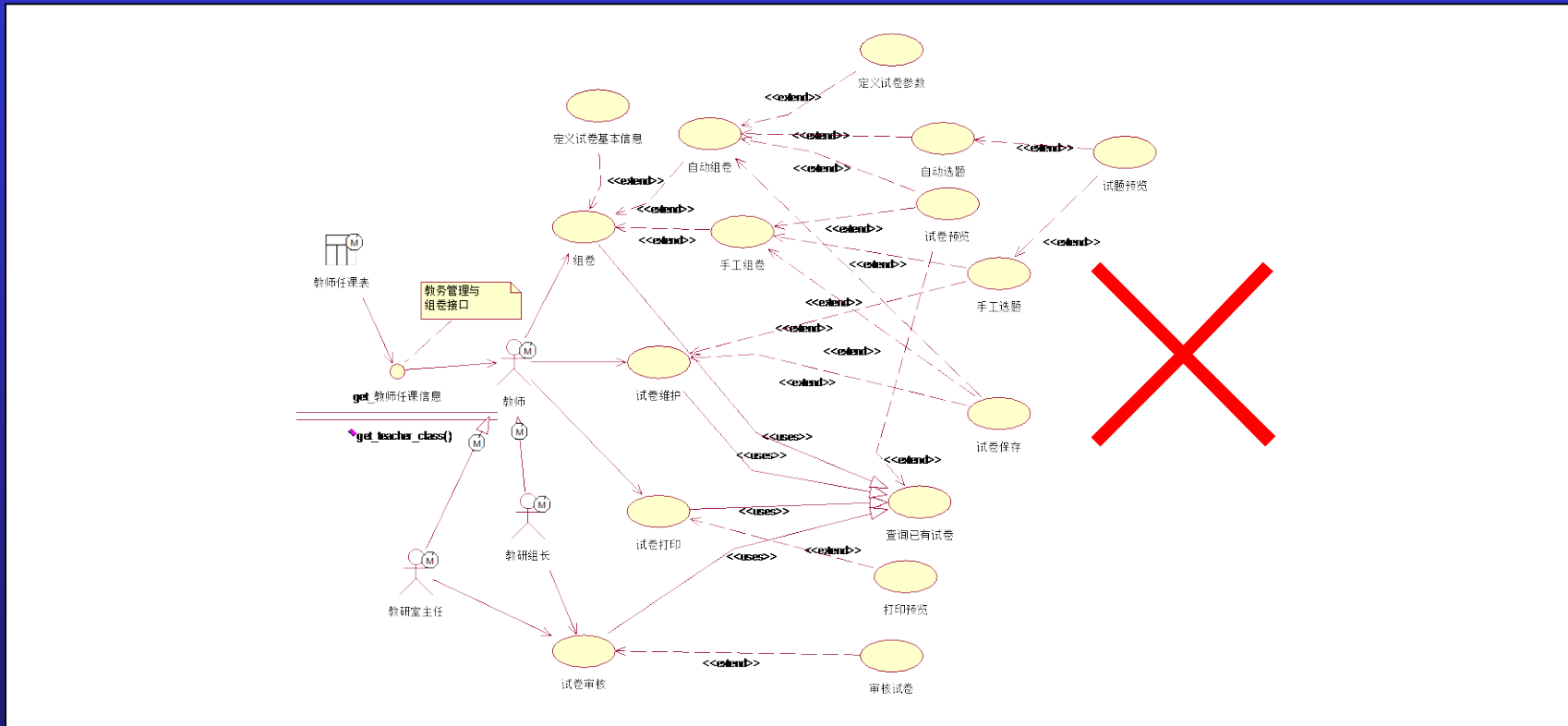


用例规约

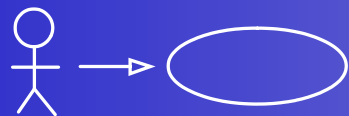
——讨论与练习、项目实作



用例关系



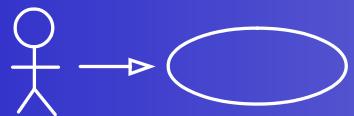
不要滥用用例关系!!!



用例关系



某些书也有误导



用例关系

➤ 扩展：分离扩展路径



➤ 包含：提取公共步骤集合



➤ 泛化：扩展的变体



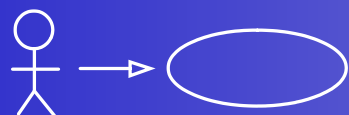
关系用于整理用例规约



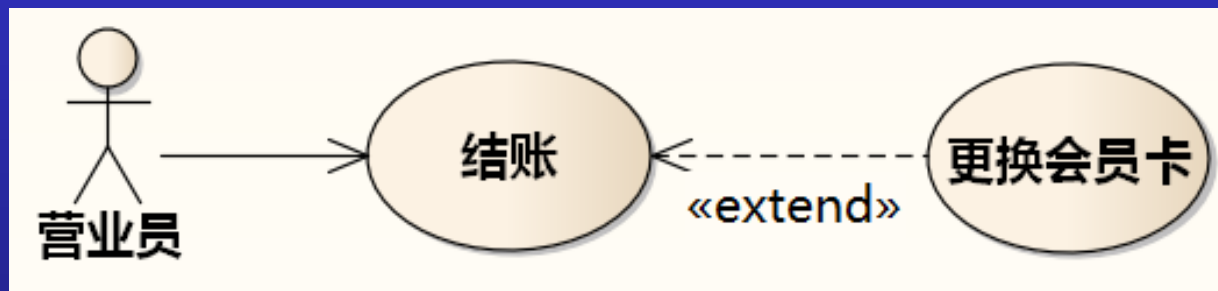
用例关系

- 扩展路径步骤多
- 扩展路径内部还有扩展点——扩展之扩展
- 扩展路径未定或容易变化——分离以“冻结”基用例

扩展的使用场合



用例关系



7 营业员提供会员卡信息，请求结账

8 系统验证会员卡未过期

9 系统计算价钱

10 系统……

扩展

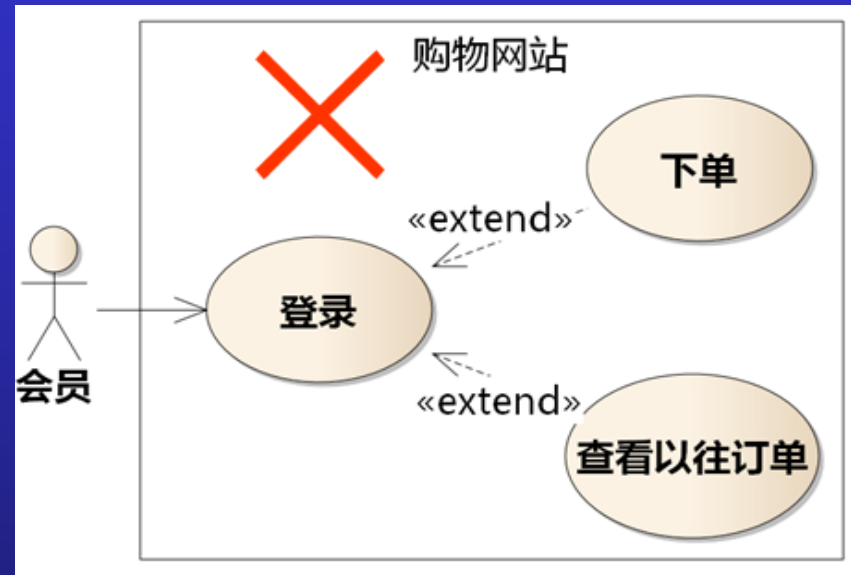
8a 会员卡已过期:

8a1 营业员更换会员卡

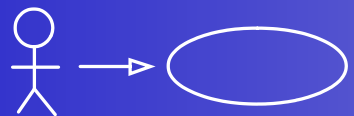
扩展举例



用例关系



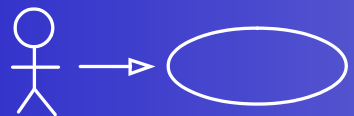
扩展的误用



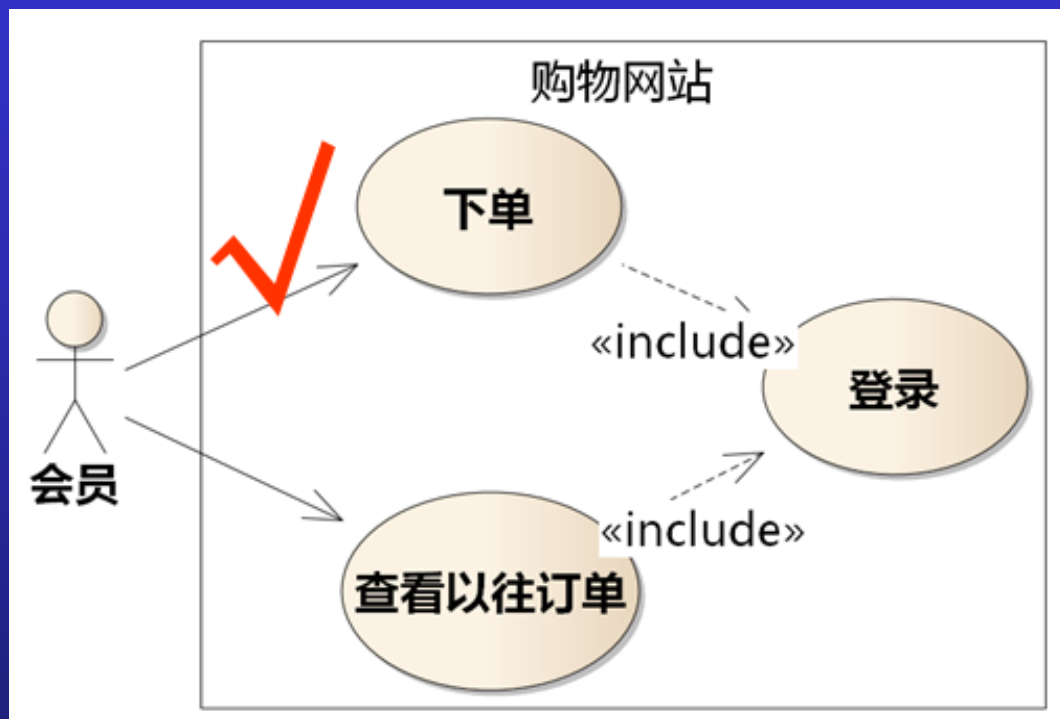
用例关系

- 步骤集合在多个用例重复出现，且单独形成价值

包含的使用场合



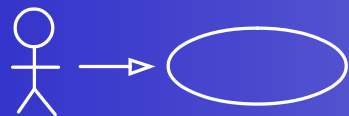
用例关系



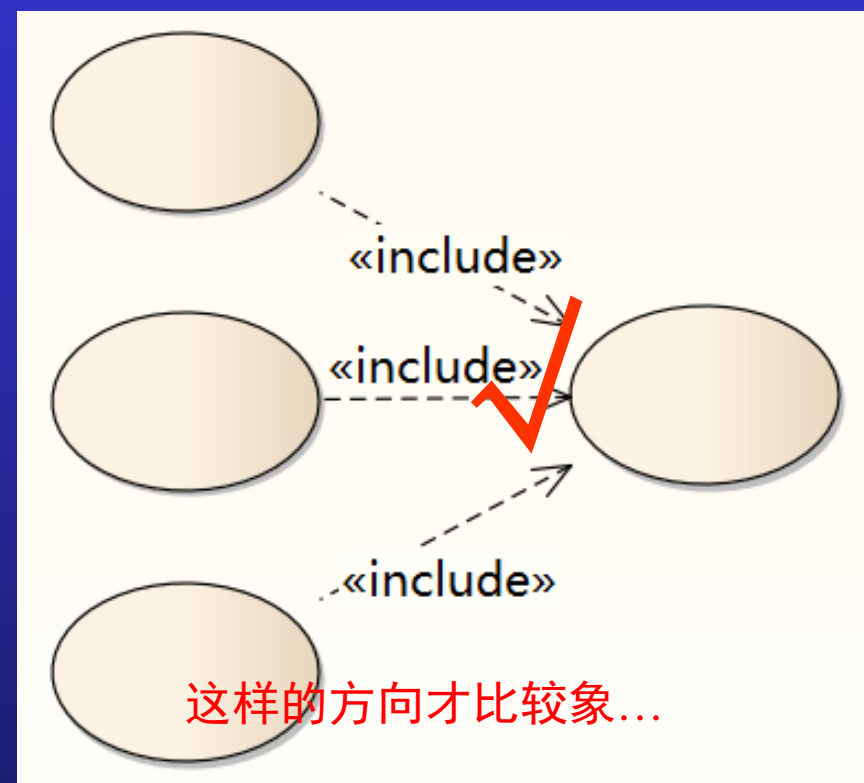
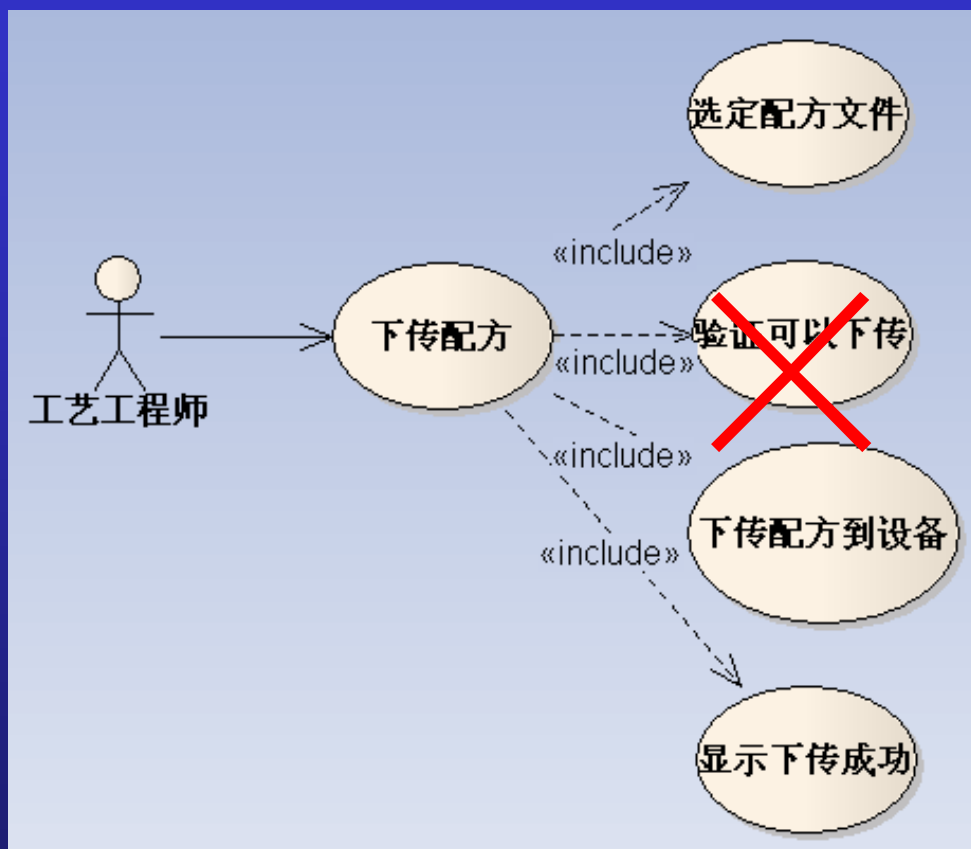
- 1 会员登录
- 2 会员提交订单信息
- 3 系统验证订单信息合法
- 4 系统……

- 1 会员登录
- 2 会员提供查询条件，请求查看以往订单
- 3 系统反馈以往订单

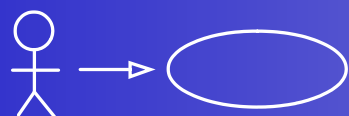
包含举例



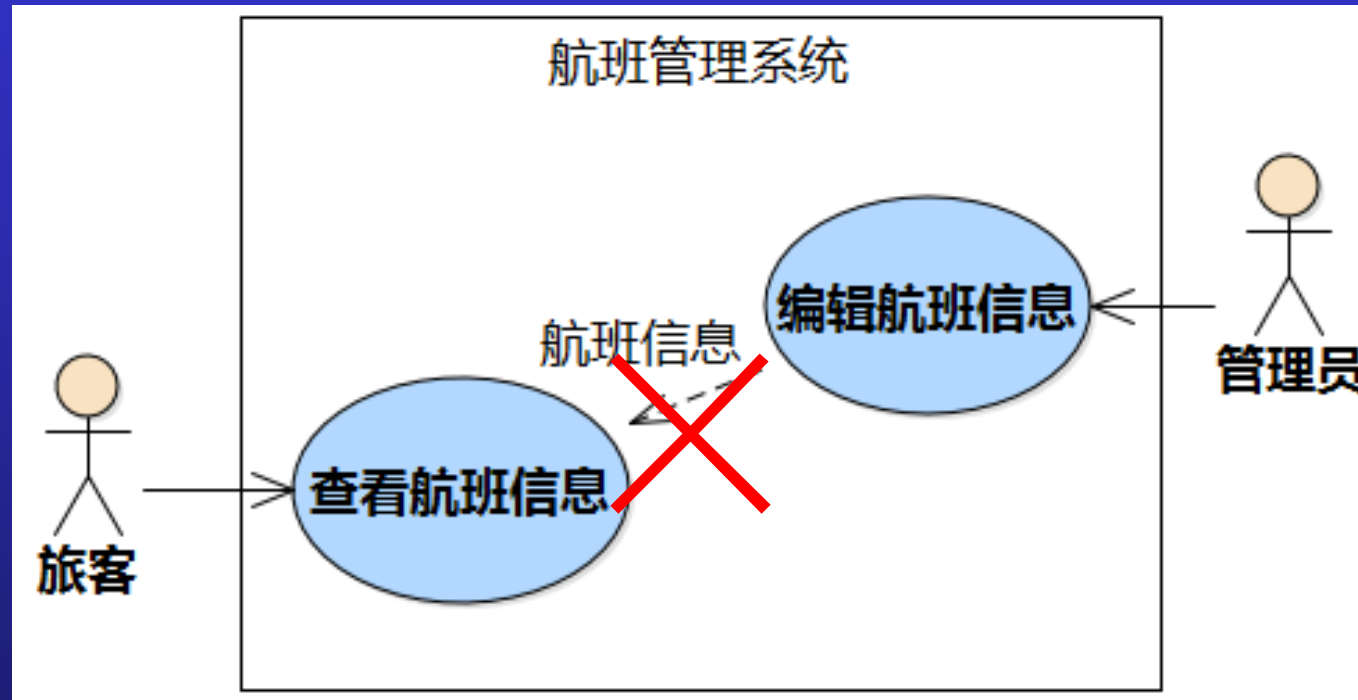
用例关系



包含的误用



用例关系



没有别的关系：不能通讯

