

# UML (Unified Modeling Language)

## 02-01 需求和用例

Software Institute of  
Nanjing University

# [ 愿景(Vision) ]

- 愿景是一个具体的目标，是一个心向往之的将来的生动画面，它既是可以被描述的，又是具有挑战性的。
  - 如果仅允许设计一份文档、模型或工具来支撑项目，我会选择愿景文档
- Philippe kruchten*

# [ 愿景(Vision) ]

- 为什么要开发这个系统
  - 谁出钱“买”这个系统？怎样才能觉得“值”

# [ 愿景(Vision) ]

- 无纸化办公→加快公文流转速度
- 把档案电子化→提高查询速度和保存时限
- 能够让人上网买书→减少买书的时间和金钱
- 能够让人网上订机票→减少买机票的时间和金钱

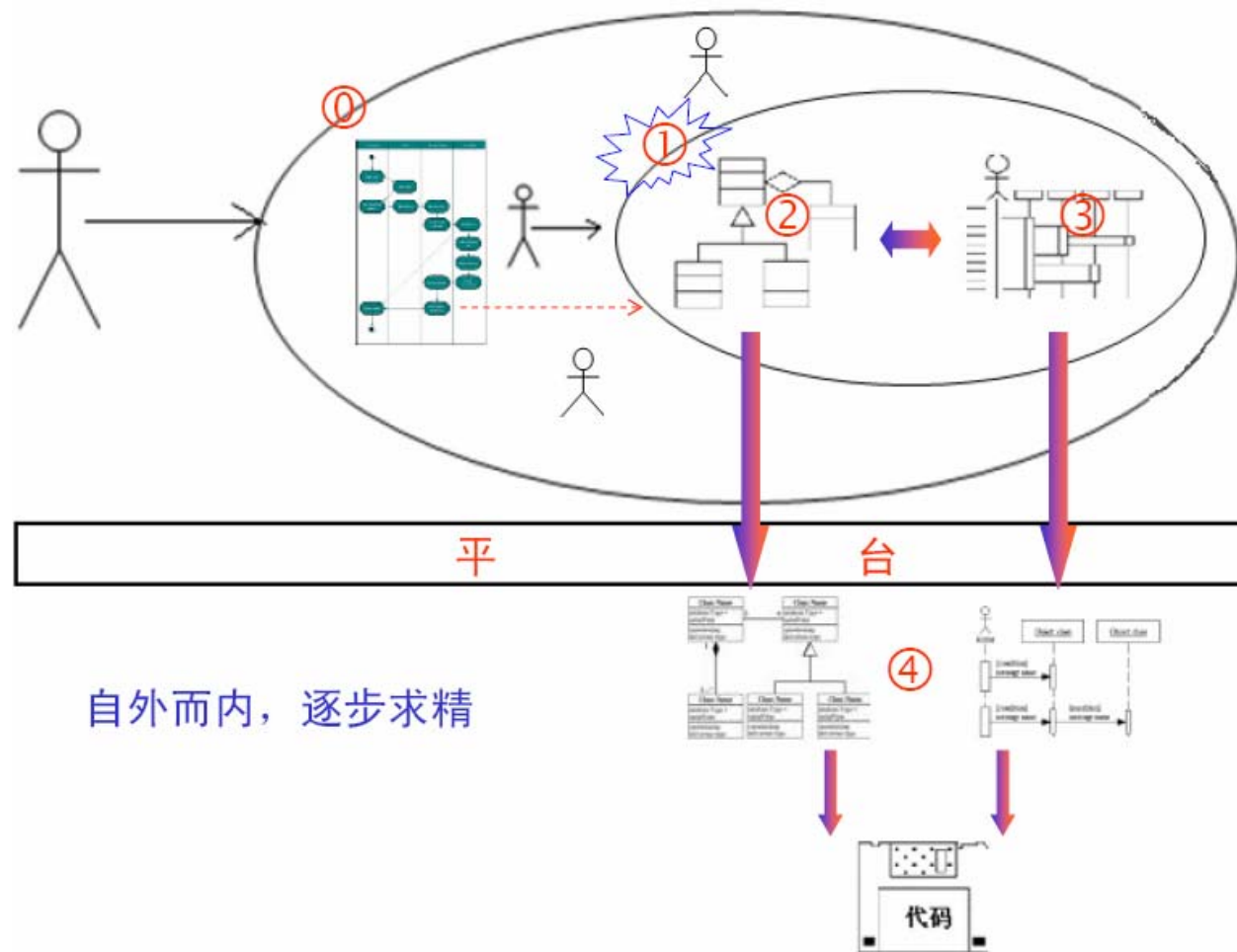
虚假的愿景到处都是

# [ 愿景(Vision) ]

- 开发这个系统的目的是为了改善我们的工作
- 开发这个系统的目的是为了工作效率
- 开发这个系统的目的是为了帮助我们进行招标工作

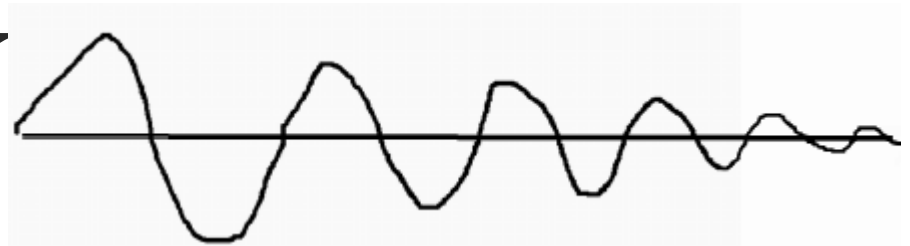
愿景必须能够度量

# 开发过程



# 需求——石头问题

- 我要一块石头...
- 差不多，但我要小一点的...
- 很好，不过我要蓝色的...
- 啊，没有那么小...
- 咳，还是原来那个好了



难捕获、易变

# 需求问题——对策

- 难捕获→从用户的视角看问题
- 易变→合理的结构

→ 用例



# 用例：基于用户目标的需求组织形式

## 立足开发者的视角

- 系统要求用户输入合法的密码
- 系统能够接受用户录入取款金额
- 系统能够从帐户中扣除取款金额
- 系统允许选择“打印或不打印收据”
- 系统能够显示交易结束信息

## 立足用户视角

- 用户插入ATM卡
- 系统要求输入密码
- 用户输入密码
- 系统验证密码正确
- 系统提示用户输入取款金额
- 用户输入取款金额并确认
- 系统验证取款金额合法
- 系统从帐户中扣除取款金额
- 系统询问用户是否打印收据
- 用户要求不打印收据
- 系统显示“交易结束”，退卡

# [ 用例：有层次的需求组织形式 ]

低精度，稳定

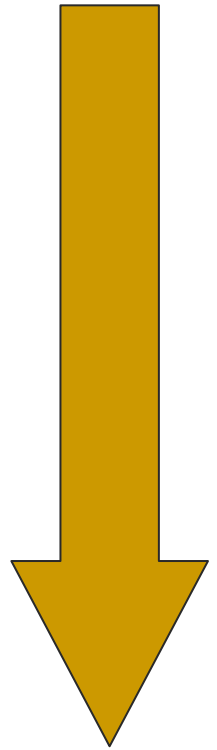
## ■ 用例（取款）

### ○ 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50的倍数）

高精度，不稳定



# 用例：使用用例探索需求



# 以用例为核心组织需求

## ■ 需求有三类：

- 功能需求
- 非功能需求
- 设计约束

“可以使用系统下订单”

“一次购物满1000元即获VIP卡”

“支持300个用户并发使用”

“使用java开发”



# 用例文档就是需求文档

- 用例名，前置条件，（涉众利益），主事件流，次要事件流，错误流，后置条件。
- 如果需要需求文档的话，增加一些字典列表、业务规则、非功能性需求、设计约束、待解决的问题
- *<http://alistair.cockburn.us/usecases/usecases.html>*

# [ 应该怎么做？——步骤 ]

- 识别Actor;
- 识别用例;
- 书写用例文档
- 通过关系整理用例



# 识别Actor

## ■ Actor—关键词（边界）

- 在系统之外，透过系统边界与系统进行有意义交互的任何事物



引入执行者帮助确定系统边界

# [ 识别Actor ]

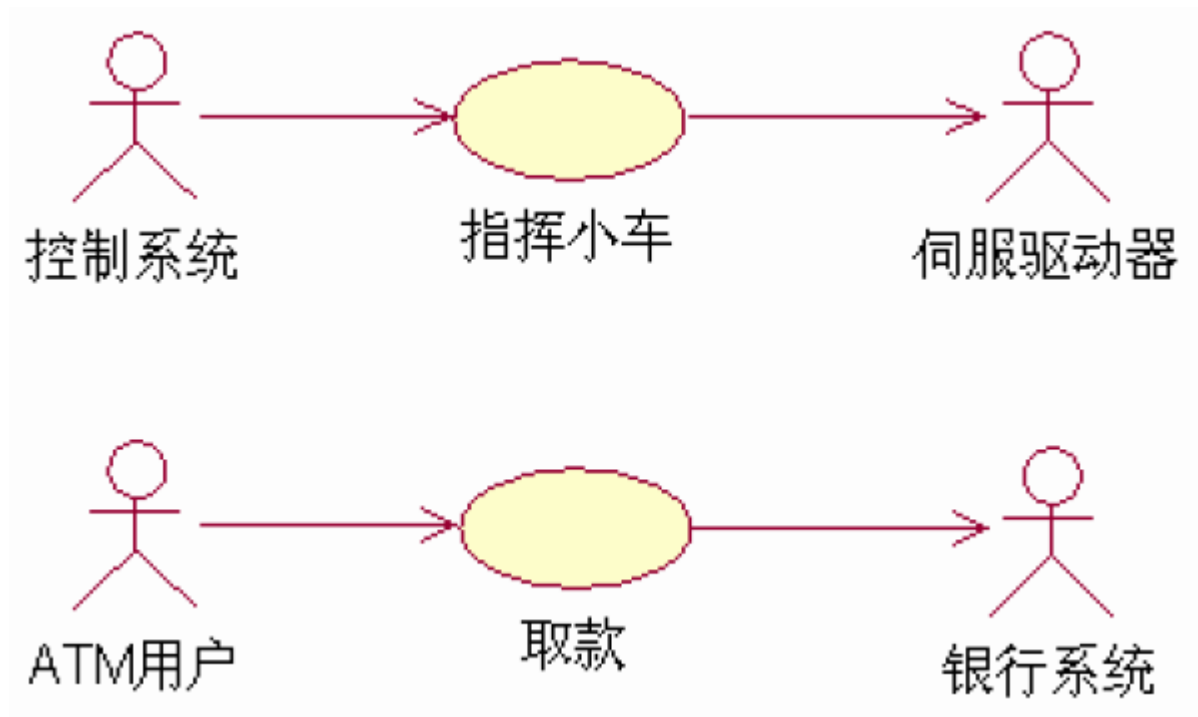
## ■ Actor要点

- 系统外——必须和它交互
- 系统边界——责任边界，非物理边界
- 系统边界——直接与系统交互
- 有意义交互——属于目标系统的责任
- 任何事物——人、外部系统、外部因素



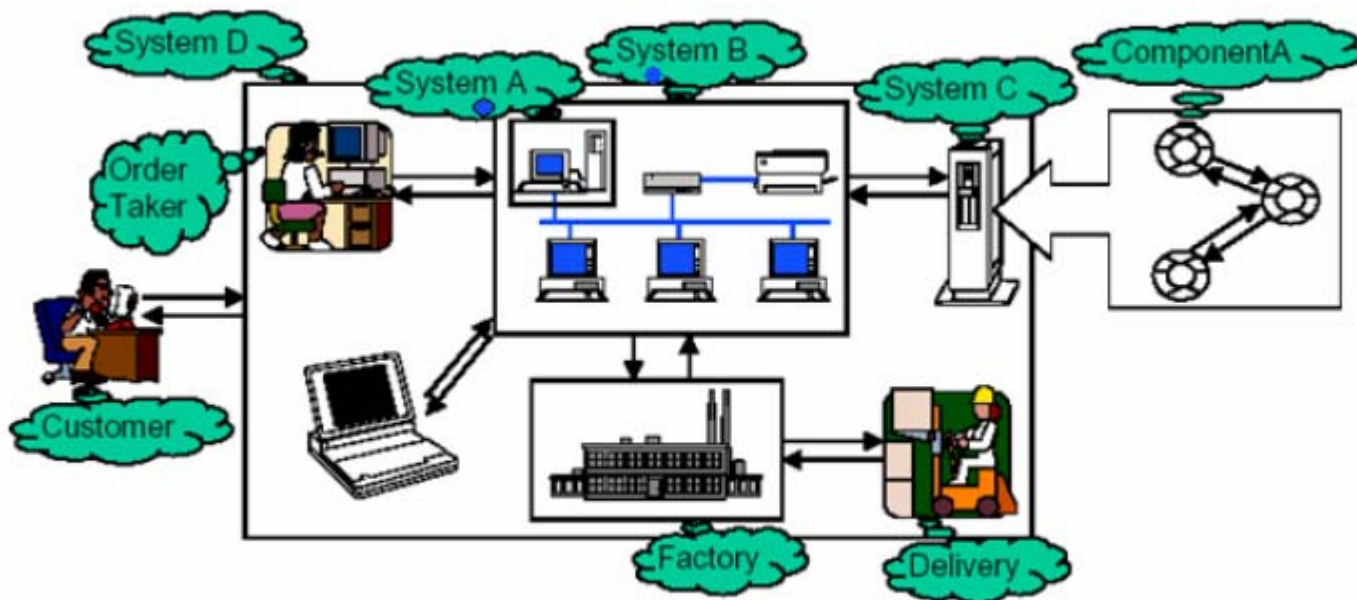
# 识别Actor

- 系统外：必须和它交互



# 识别Actor

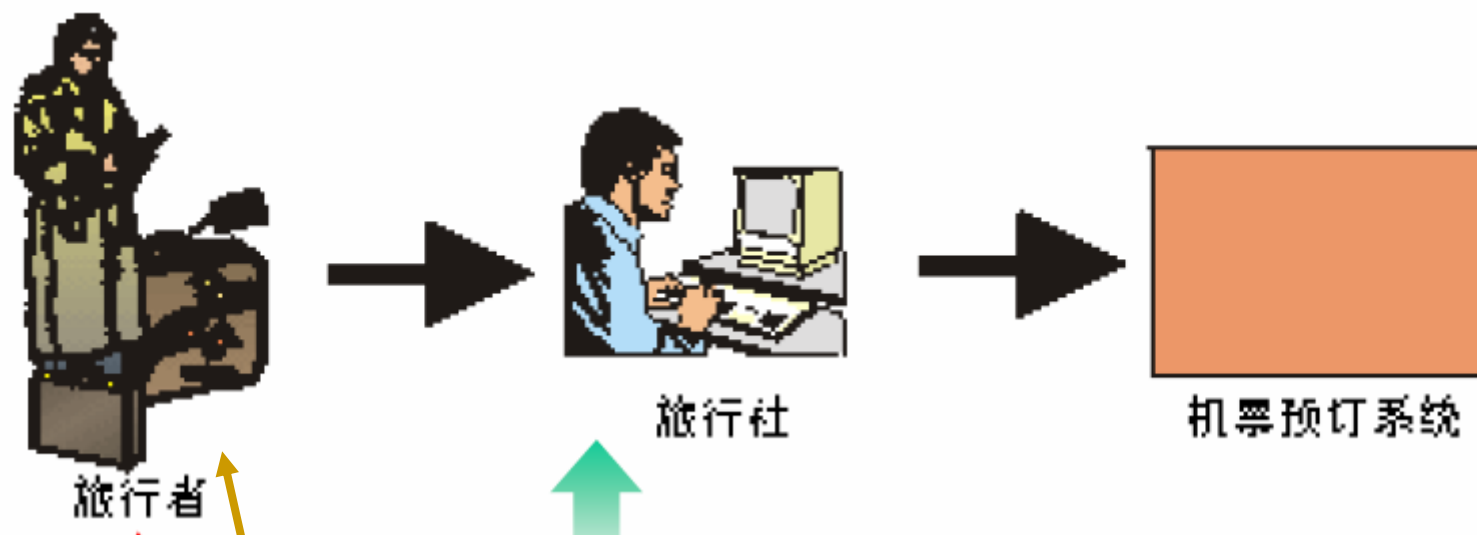
- 责任的边界，不是物理的边界



即使最终都是操作同一数据库同一张表，也可以有不同的系统

# 识别Actor

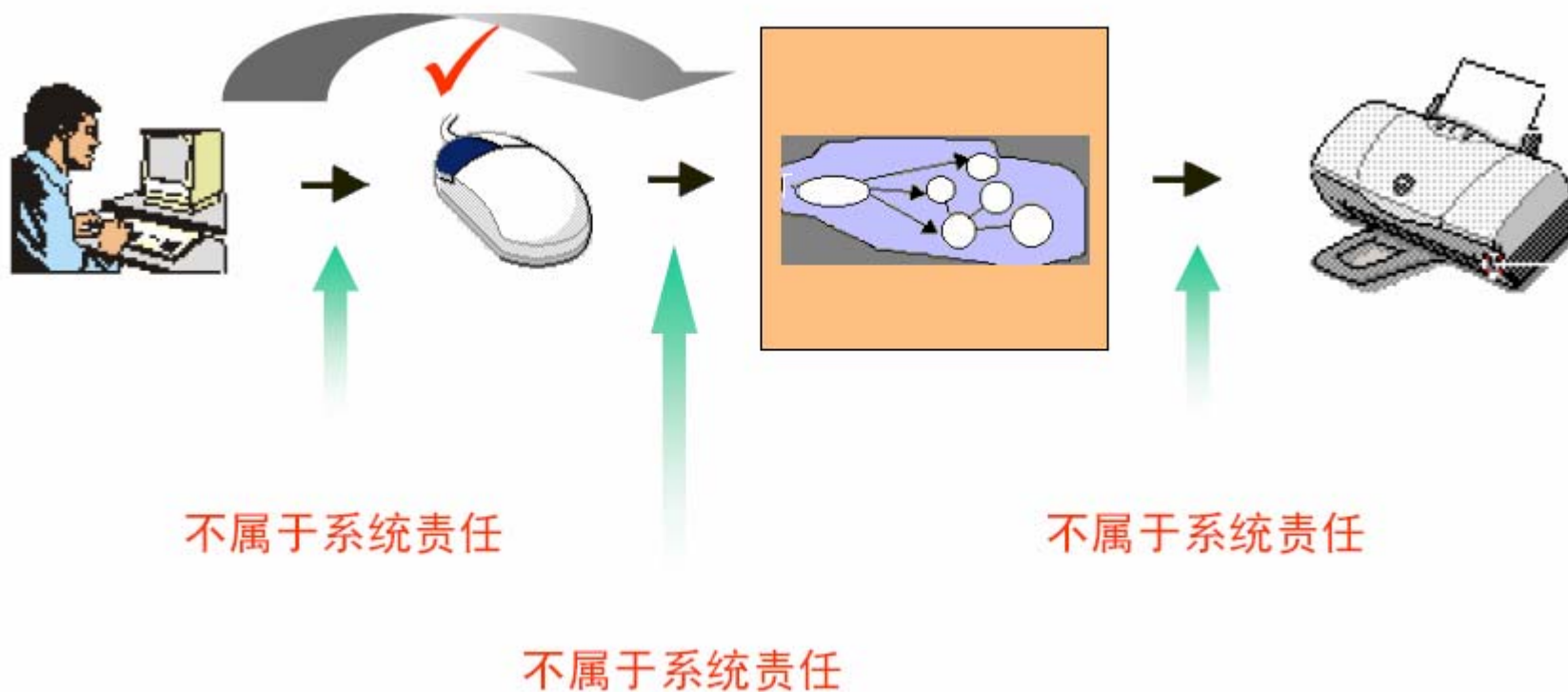
## ■ 直接与系统交互



那么，他算什么？

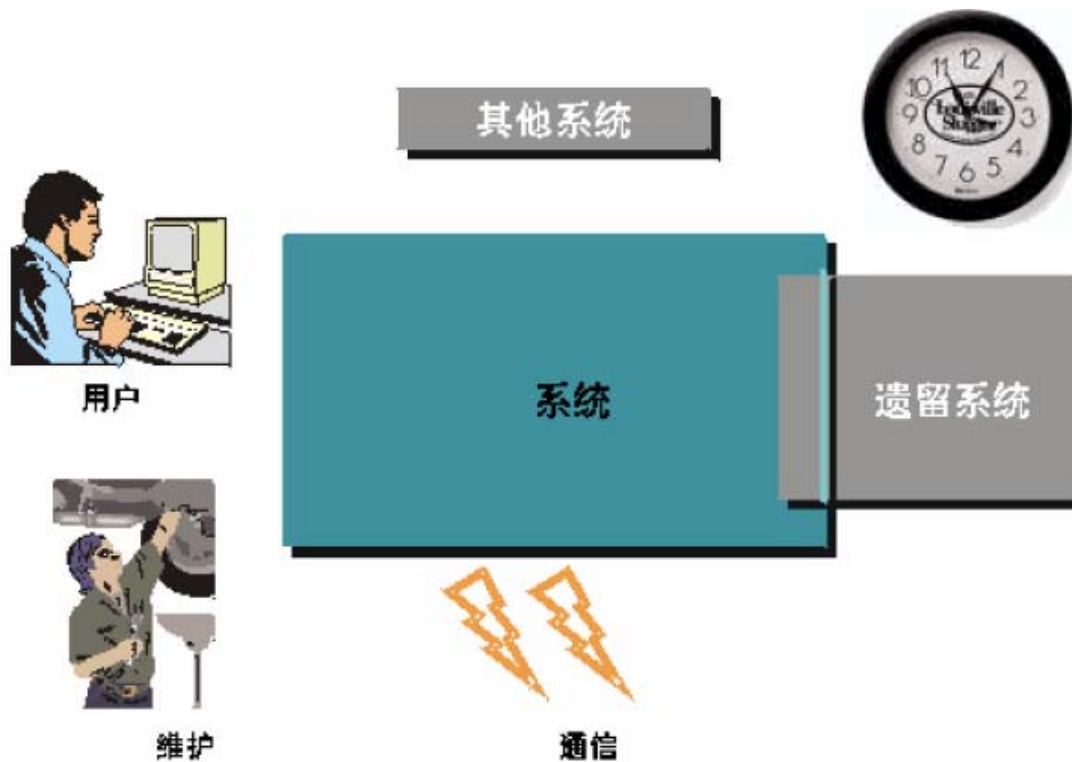
# 识别Actor

## ■ 有意义的交互



# 识别Actor

## ■ 任何事物



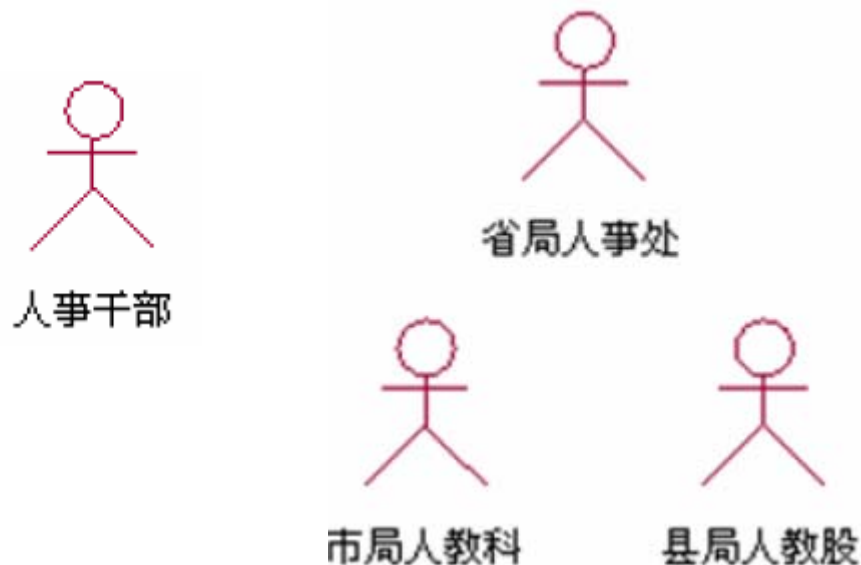
# 识别Actor

## ■ 识别Actor的思路

- 谁在使用系统的主要功能
- 谁改变系统的数据
- 谁从系统获取信息
- 谁需要系统的支持以完成日常工作任务
- 谁负责维护、管理并保持系统正常运行
- 系统需要应付（处理）哪些硬件设备
- 系统需要和哪些外部系统交互
- 谁（或什么）对系统运行产生的结果感兴趣
- 有没有自动发生的事情

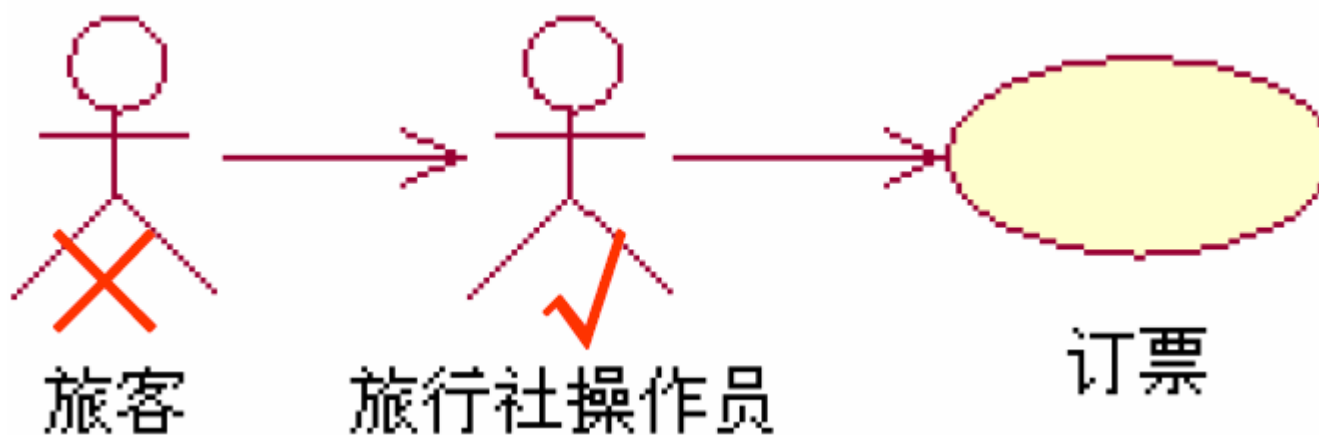
# [ 识别Actor ]

- 都对，不丢用例就行（慢慢清理）



# 识别Actor

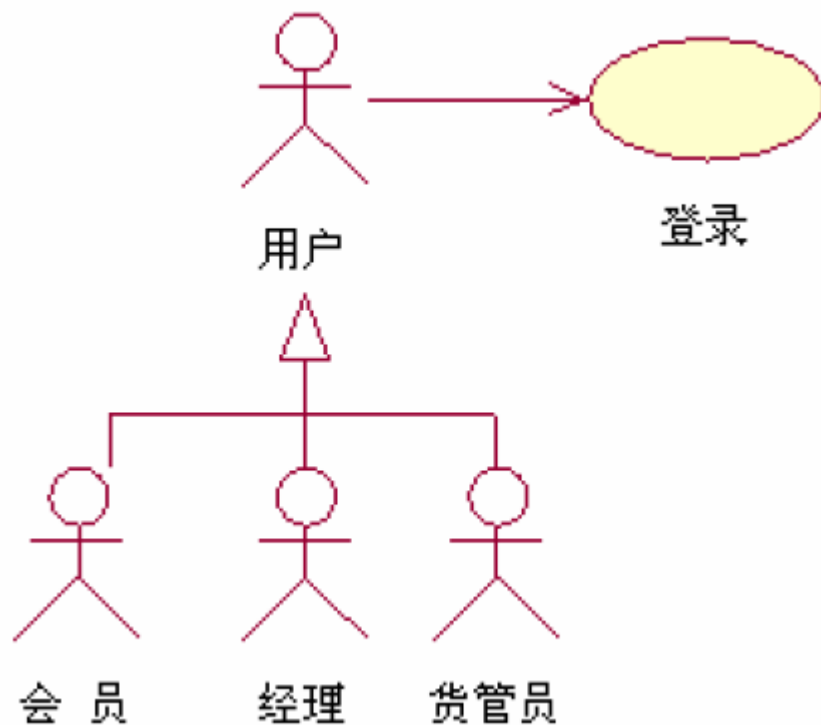
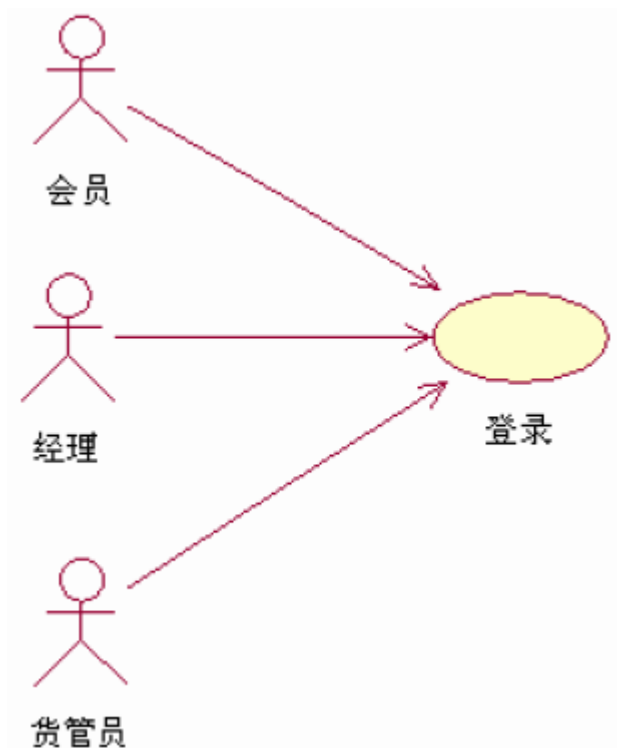
- 关键在边界，不在数量





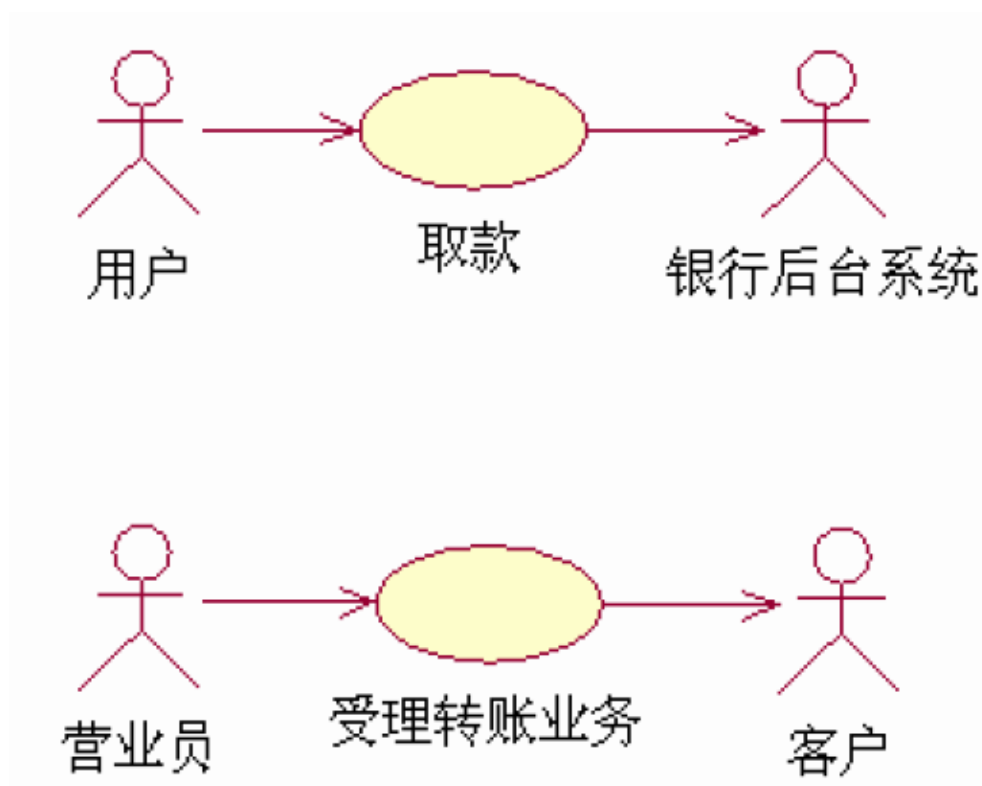
# 识别Actor

- 责任类似或重叠——泛化出一个抽象的执行者



# 识别Actor

- 主Actor完成用例时可能需要辅助Actor



# 应该怎么做？——步骤

- 识别Actor;
- 识别用例;
- 书写用例文档
- 通过关系整理用例



# 识别用例

## ■ 用例的基本定义（关键词—价值）

- 用例实例是在系统执行中的一系列动作，这些动作将产生特定执行者可见的价值结果。一个用例定义了一组用例实例。

目标

步骤

路径

- 通俗的说

- 执行者通过系统达到某个目标

# [ 识别用例 ]

## ■ 用例要点

- 价值结果→有意义的目标
- 系统执行→价值结果由系统生成
- 执行者可见→业务语言，用户观点
- 一组用例实例→用例的粒度

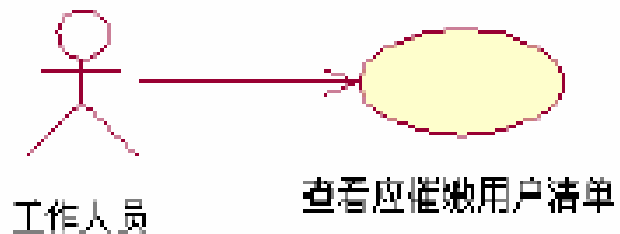
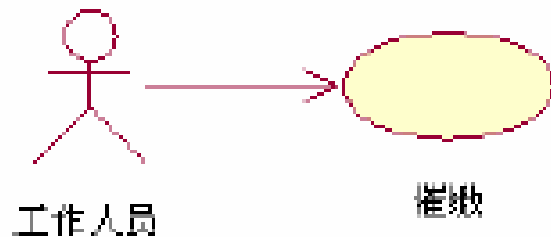
# 识别用例

- 有意义的目标（可度量的价值）



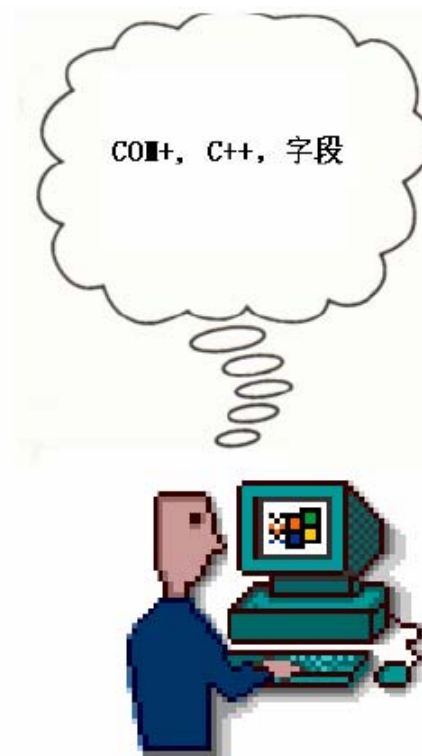
# 识别用例

## ■ 价值结果由系统生成



# 识别用例

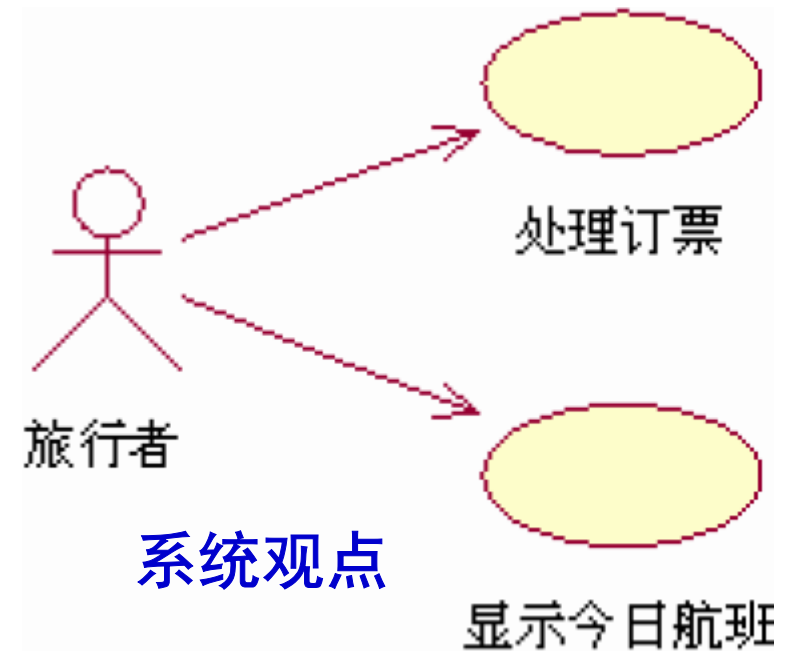
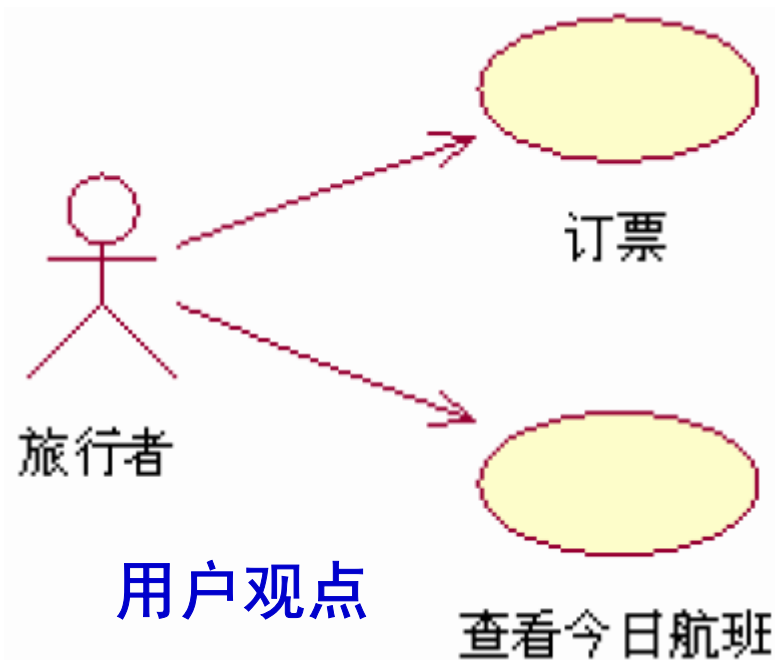
## ■ 业务语言非技术语言





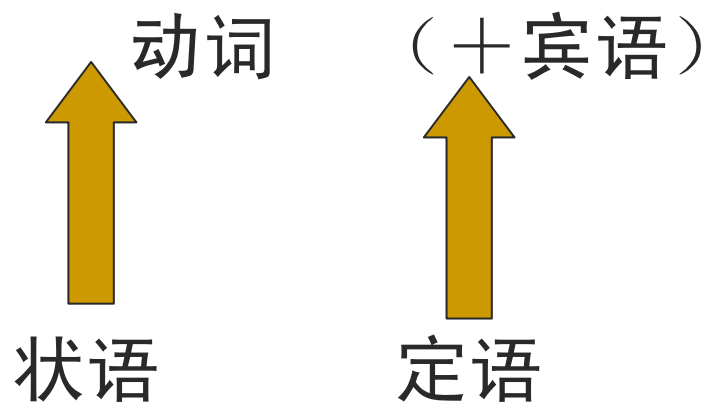
# 识别用例

## ■ 用户观点而非系统观点



# [ 识别用例 ]

- 用例命名：执行者视角



# [ 识别用例 ]

- 用例命名：慎用弱动词弱名次
- 弱动词：进行、使用、复制、加载、重复...
- 弱名词：数据、报表、表格、表单、系统...

会掩盖真正业务

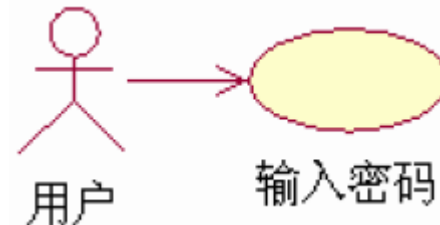
# [ 识别用例 ]

- 用例的“粒度”
- 粒度原则：用例要有路径，路径要有步骤。而这一切都是“可观测”的。

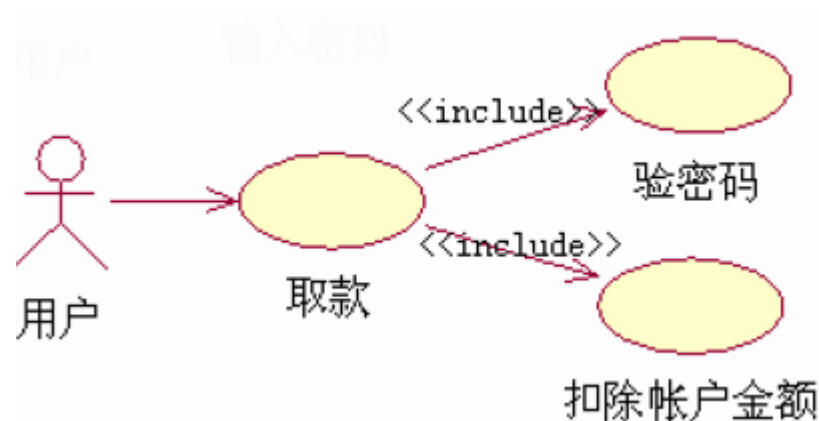
# 识别用例

## ■ 用例的“粒度”

- 最常见的错误——把步骤当作用例



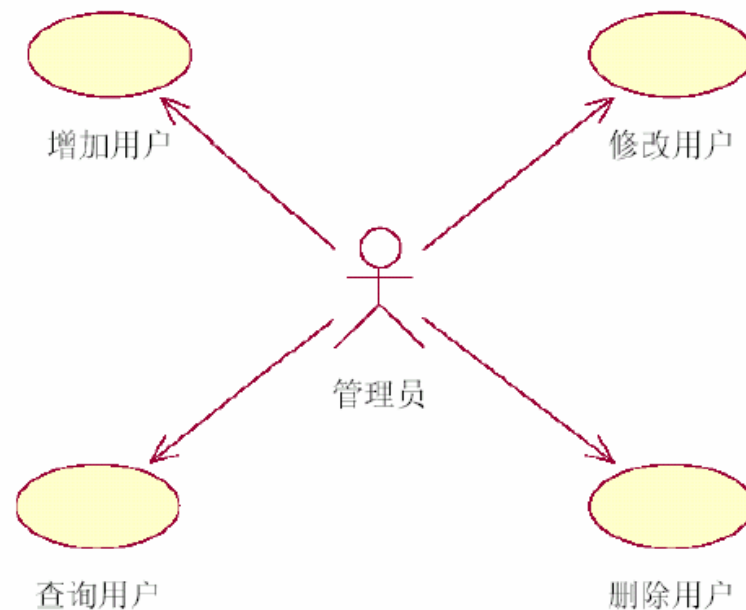
- 把系统活动当用例



# 识别用例

## ■ 用例的“粒度”：四轮马车

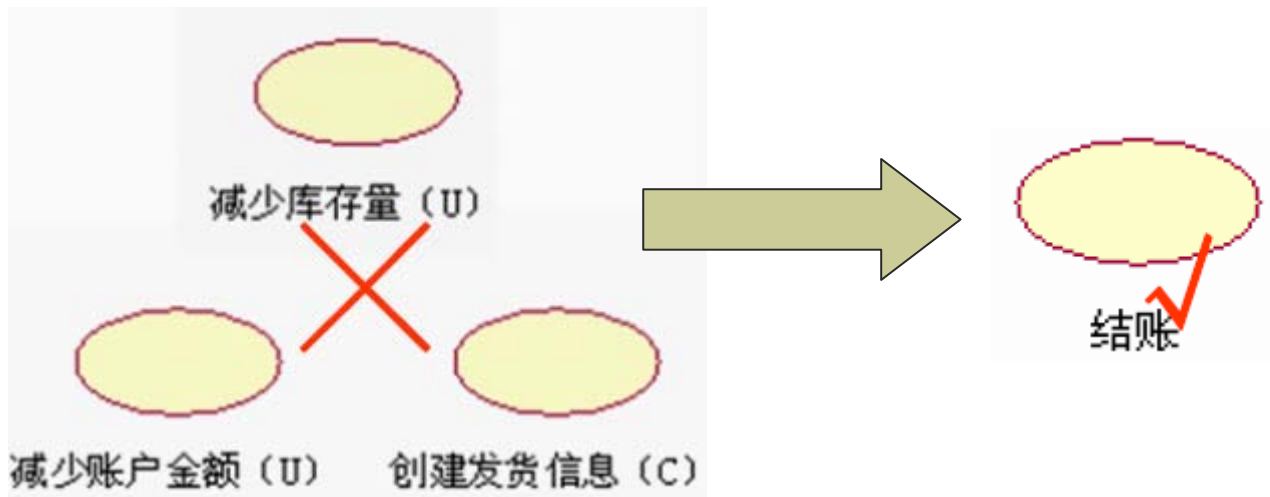
- 警惕CRUD泛滥
- 用有色眼镜看，所有业务最终都会变成CRUD
- 多问：为什么要CRUD？  
光CRUD能为actor提供价值嘛？



**CRUD: 新增、读取、修改、删除**

# 识别用例

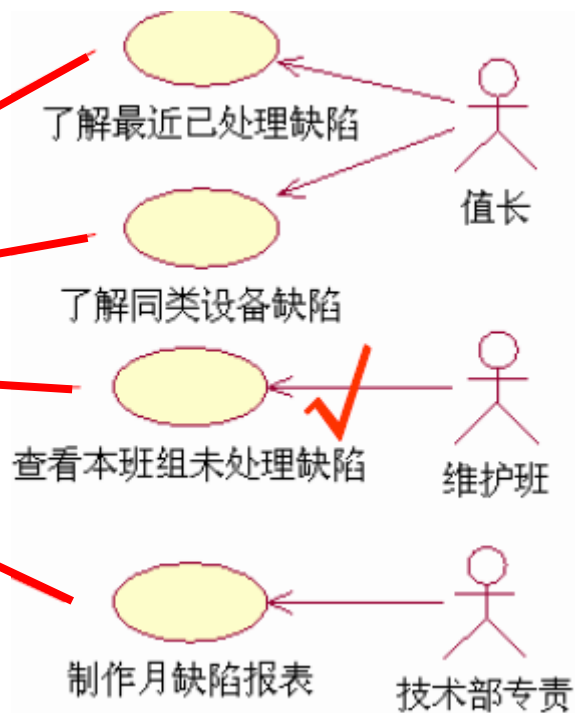
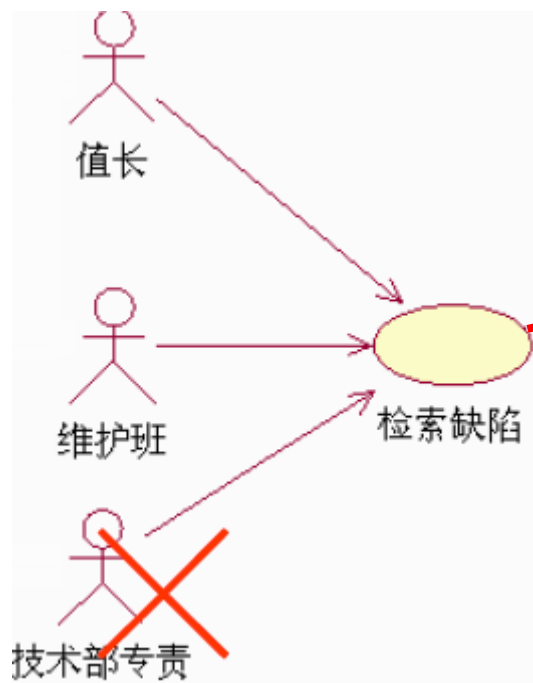
## ■ 用例的“粒度”：四轮马车



一个用例背后可能隐藏着许多数据操作

# 识别用例

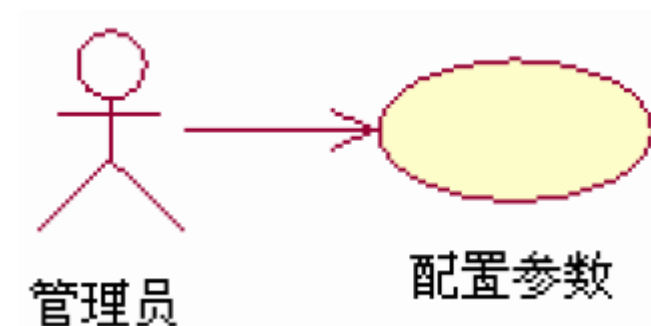
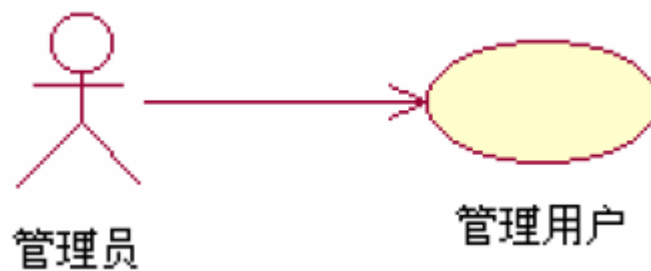
## ■ 用例的“粒度”：四轮马车





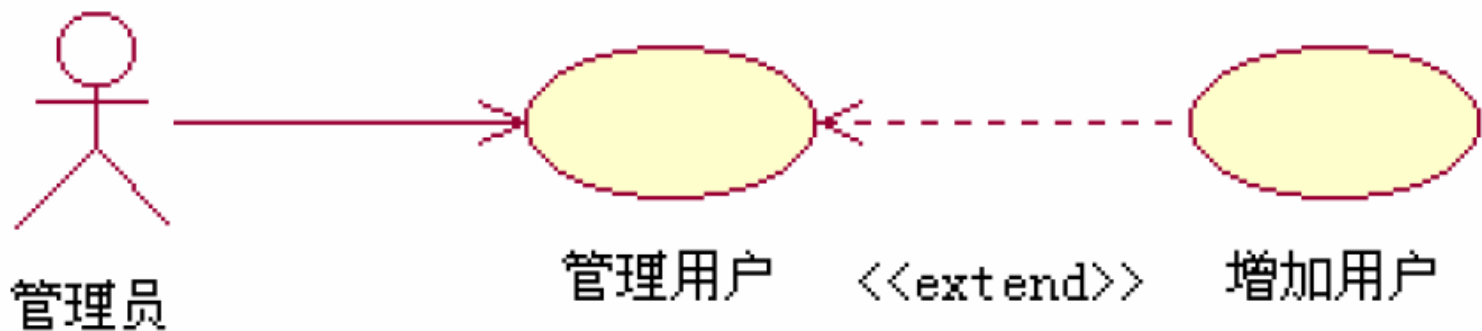
# 识别用例

- 用例的“粒度”：如果确实是纯CRUD
  - 如果CRUD不涉及复杂的交互，一个用例“管理XX”即可
  - 不管是CRUD都是为了完成管理的目标
  - 甚至很多种基本数据的管理都可以用一个用例表示



# 识别用例

- 用例的“粒度”：灵活处理CRUD



也可以把包含复杂交互的路径独立出去形成用例

# 识别用例

## ■ 用例的“粒度”：用例的实质



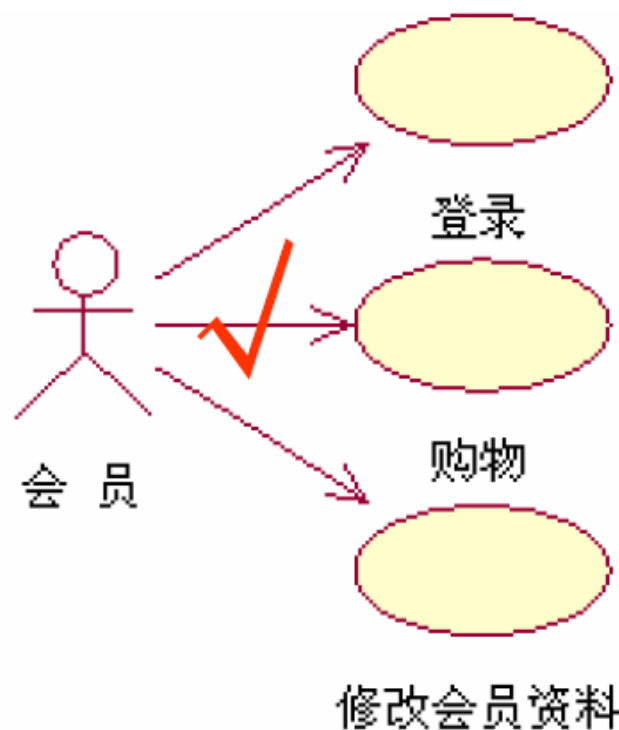
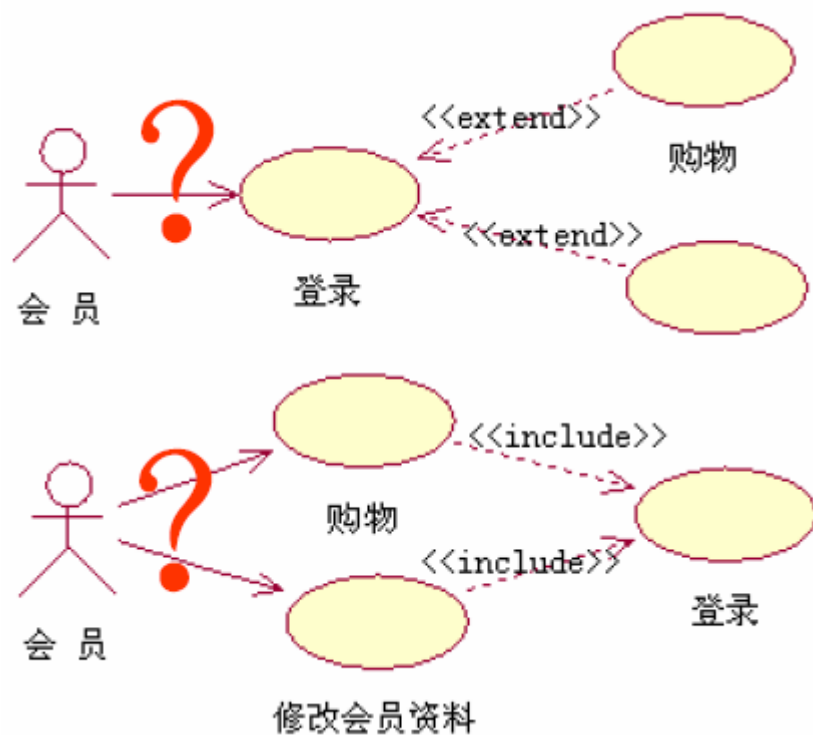
用例表征系统使用复杂度，与系统内部复杂度无关

# [ 识别用例 ]

- 执行者使用这个系统达到什么目标？
- 语法测试：【执行者】使用系统来【用例】

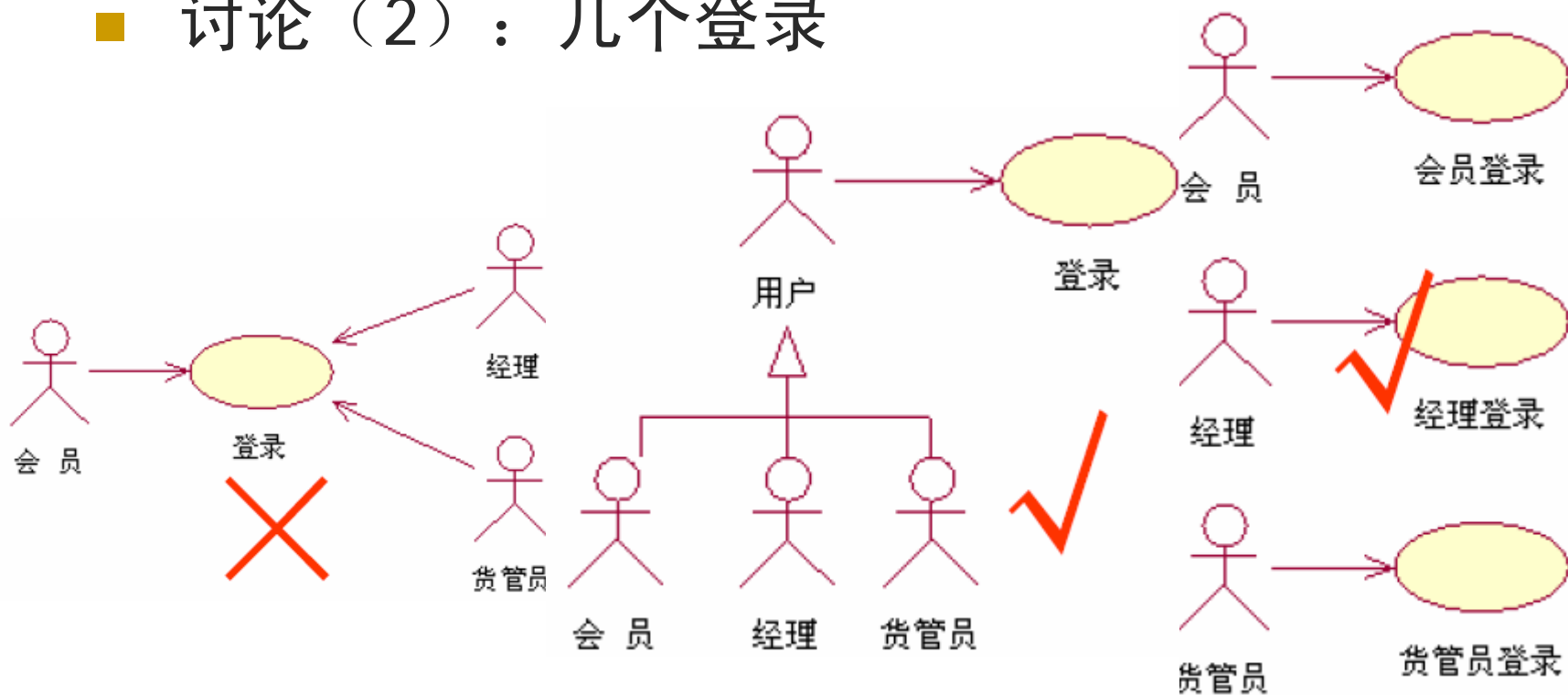
# 识别用例

## ■ 讨论（1）：登录怎么处理



# 识别用例

## ■ 讨论（2）：几个登录



# 识别用例

- 为什么总是急于加入细节



虚假的安全感

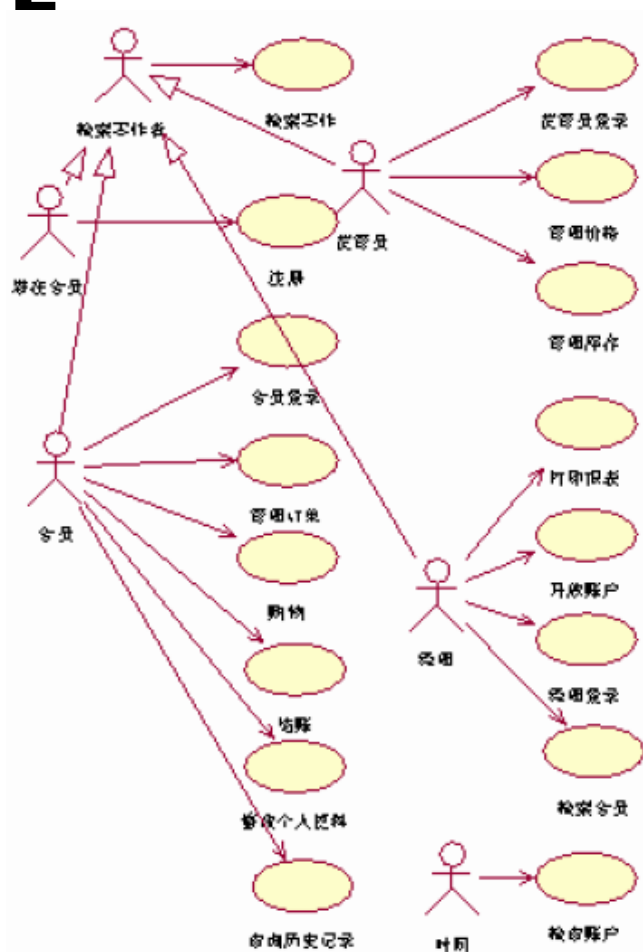
# 应该怎么做？——步骤

- 识别Actor;
- 识别用例;
- 书写用例文档
- 通过关系整理用例





# 用例文档：更进一步的精度

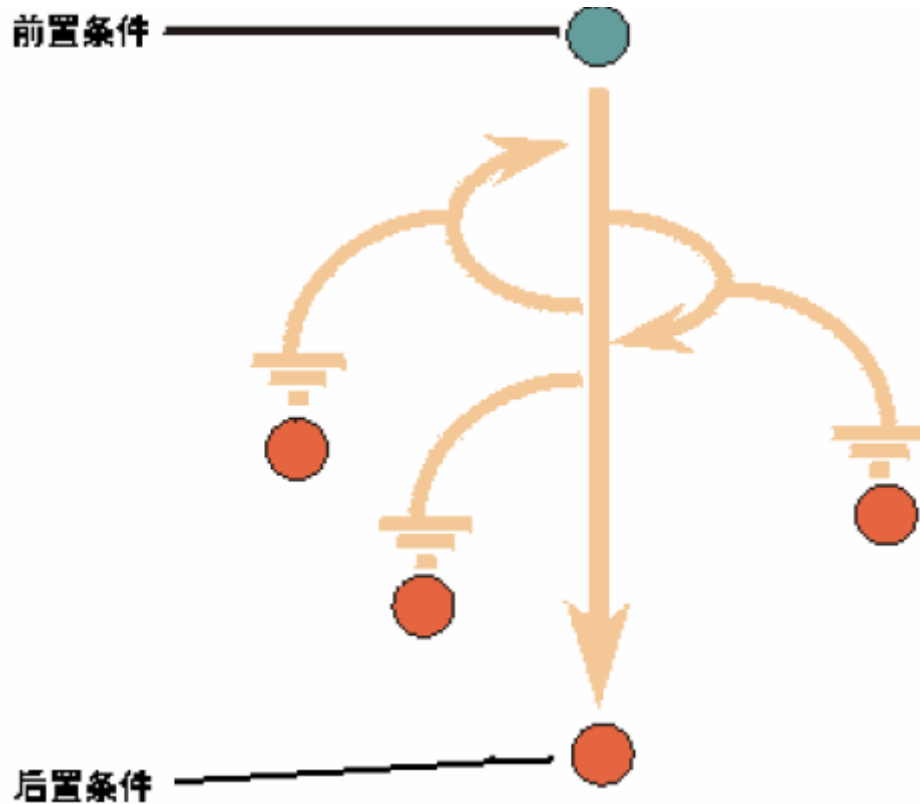


- 用例图可以作为用例文档的总图
- 进一步的精度：有层次的文档
- 文档中每一句话都有其价值

# 书写用例文档—用例的内容

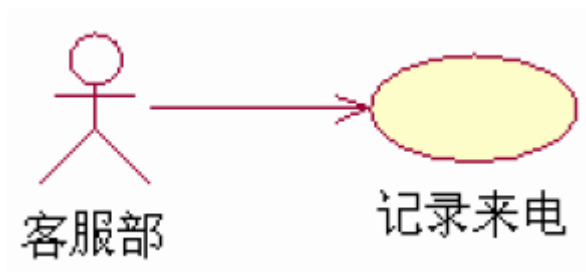
- 用例编号：用例名
- 执行者
- 前置条件
- 后置条件
- 涉众利益
- 主事件流（基本路径）
  - 1.....××××
  - 2.....××××
  - 3.....××××
- 次要时间流（扩展路径）
  - 2a××××：
    - 2a1.....××××
- 错误流（系统错误异常流）
- 字段列表
- 业务规则
- 非功能需求
- 设计约束
- 待解决问题

# 书写用例文档—前置、后置条件



- 开始用例前必需的系统及环境状态
- 注意：必须是系统能检测的
- 用例成功结束后系统应该具备的状态

# 书写用例文档—前置、后置条件



前置条件

客户打来电话

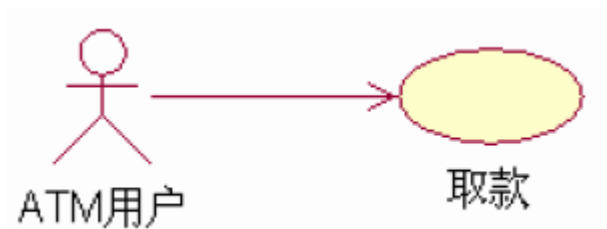


前置条件

客服部人员已经登录



前置条件必须是系统能检测到的



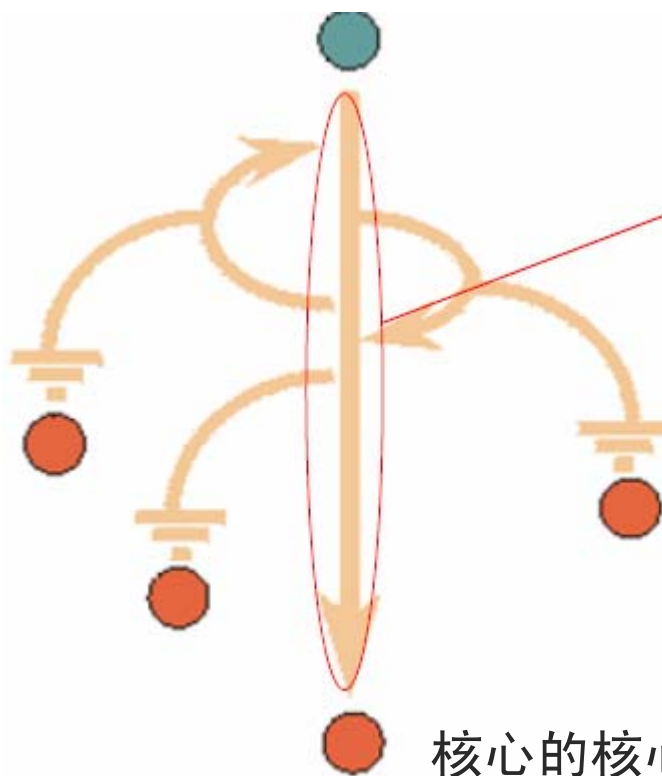
前置条件

ATM用户的账户里有足够的金额

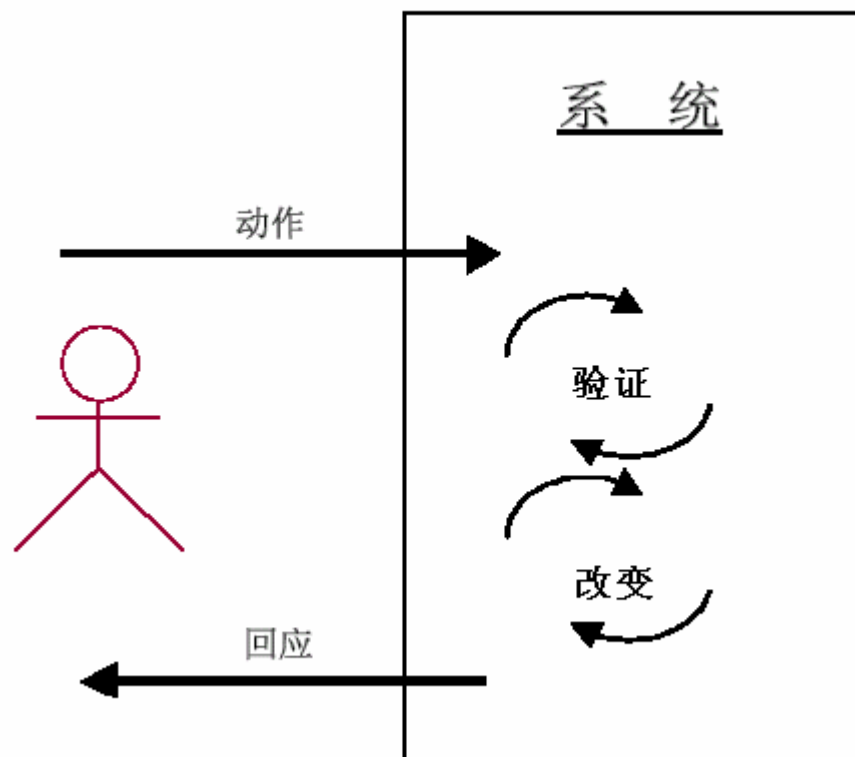


前置条件必须是系统在用例开始能检测到的

# 书写用例文档—主事件流



# 书写用例文档—用例交互四步曲



步骤里面写什么？写需求呗

# 书写用例文档—事件流步骤的描述

- 只书写“可观测”的（说人话）
- 使用主动语句
- 句子必须以Actor或系统为主语
- 每一句都要向目标迈进
- 分支和循环
- 不要涉及界面细节

# 书写用例文档—事件流步骤的描述（1）

- 系统通过ADO建立数据库连接，~~传送SQL查询语句，从“零件”表查询...~~
- 系统按照查询条件搜索零件

只书写“可观测”的







## 书写用例文档—事件流步骤的描述（2）

- 欧文从贝克汉姆处得到传球，守门员...
- 贝克汉姆传球给欧文，欧文射门，守门员扑救...

- 主动语句—球在谁那里？



## 书写用例文档—事件流步骤的描述（2）

- 系统从会员处获得用户名和密码 
- 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 

## 书写用例文档—事件流步骤的描述（3）

- Actor × × × ×
- 系统 × × × ×
- 系统 × × × ×
- Actor × × × ×

句子必须以Actor或系统为主语

## 书写用例文档—事件流步骤的描述（4）

- Actor填写姓名
- Actor填写电话
- Actor填写联系地址
- Actor提交
- × × × ×



每一句都要朝目标迈进

# 书写用例文档—事件流步骤的描述（5）

- 分支：放到次要事件流（扩展路径）
- 循环：直接描述

## 书写用例文档—事件流步骤的描述（6）

- 会员从下拉框中选择类别
- 会员在相应文本框中输入查询条件
- 会员点击“确定”按钮
- ...

不要涉及界面细节

# 书写用例文档—补充约束

- 字段列表
  - 业务规则
  - 非功能需求
  - 设计约束
- 
- 可以直接放在用例中，也可以单独集中到另外的文档从用例文档指向

# 书写用例文档—字段列表

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



# 书写用例文档—字段列表

- 注册信息 = 公司名 + 联系人 + 电话 + {联系地址}\*  
\* 表示可选
- 联系地址 = 省 + 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}

用表达式表示

# 书写用例文档—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
  - 一开始就好像做了很多事情，其实却回避了困难的业务问题

# 书写用例文档——业务规则

- 事实

- 设备是资产的一种

- 推理或者解释

- 如果过了计划中的交付日期，货物还没有到，即为“未按时送货”

- 约束

- 合同的总金额不能超过卖方的信用额度

- .....

# 书写用例文档——业务规则

## ■ 区别规则和解决方案

- 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

# 书写用例文档—业务规则

## ■ 业务规则的各种表示方法

- 文字说明
- 决策表
- OCL
- .....

# 书写用例文档—业务规则

## 决策表

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%				√	

# 书写用例文档—非功能需求

- 一开始，功能需求决胜
- 类似产品多了，非功能需求决胜

非功能!=不重要

# 书写用例文档—非功能需求

- 可用性
- 可靠性
- 性能
- 可支持性






# 书写用例文档—非功能需求：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→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→可用性的问题

# 书写用例文档—非功能需求：可用性

## ■ 可用性需求的表达

- 系统应易于使用 
- 第一次使用30分钟内能学会添加新员工（任务时间）
- 5次击键能完成客人入住服务，不需要使用鼠标（操作次数）
- 80%的用户认为系统易学，并且使用效率高（用户调查） 
- 系统界面应如××附件所表示的屏幕图像（小心！） 

# 书写用例文档—非功能需求：可靠性

## ■ 各种可靠性需求

- 系统应能防范磁盘故障（安全）
- 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)（稳定性）

# 书写用例文档—非功能需求：性能

## ■ 各种性能需求

- 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）

# 书写用例文档—非功能需求：可支持性

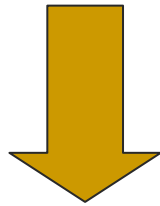
## ■ 各种可支持性需求

- 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  
×××
- 在两年内，以每个功能点××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

# 书写用例文档—设计约束

## ■ 界面样式

- 系统显示订单的统计信息



- 应采用附件××所示的屏幕格式....



# 书写用例文档—设计约束

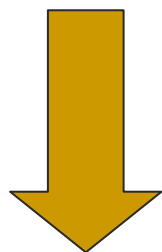
## ■ 平台性的约束：一般针对整个系统

- 由于劳动局目前有IBM x346 Pc服务器两台，还需要使用三年，所以系统应在这些PC服务器上运行。
- 由于财政厅的IT人员有BEA Weblogic的专长，而且目前其他的应用系统都使用Weblogic，所以本系统应使用Weblogic作应用服务器。

# 书写用例文档—设计约束

## ■ 技术接口

- 系统读取探测仪读数



- 通过RS-232接口读取

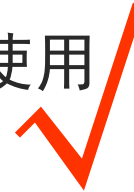


# 书写用例文档—设计约束

- 系统应采用三层架构方式搭建

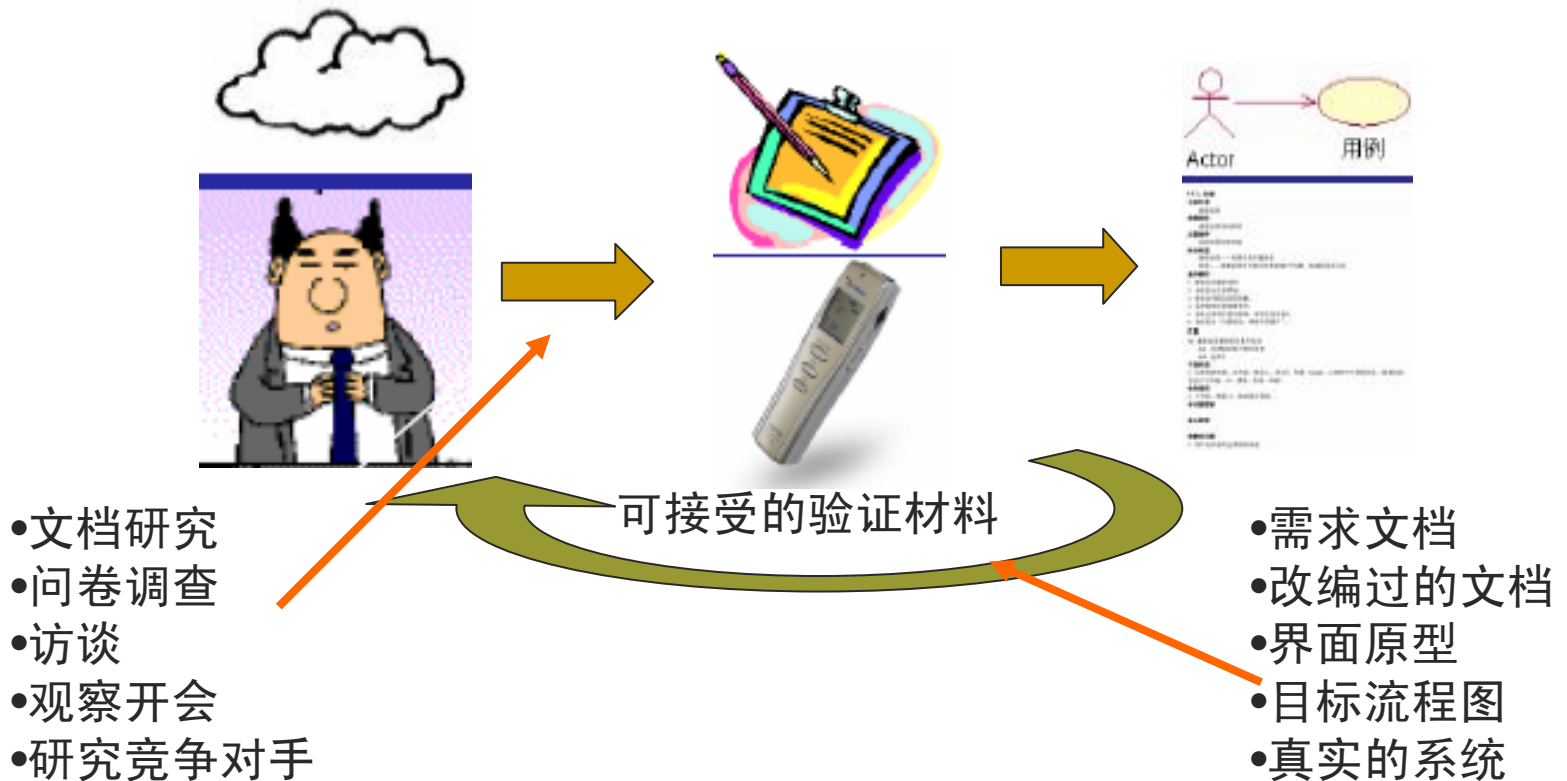


- 系统应允许分布在各地的多个用户并发使用



小心泛滥

# 书写用例文档—用例的验证



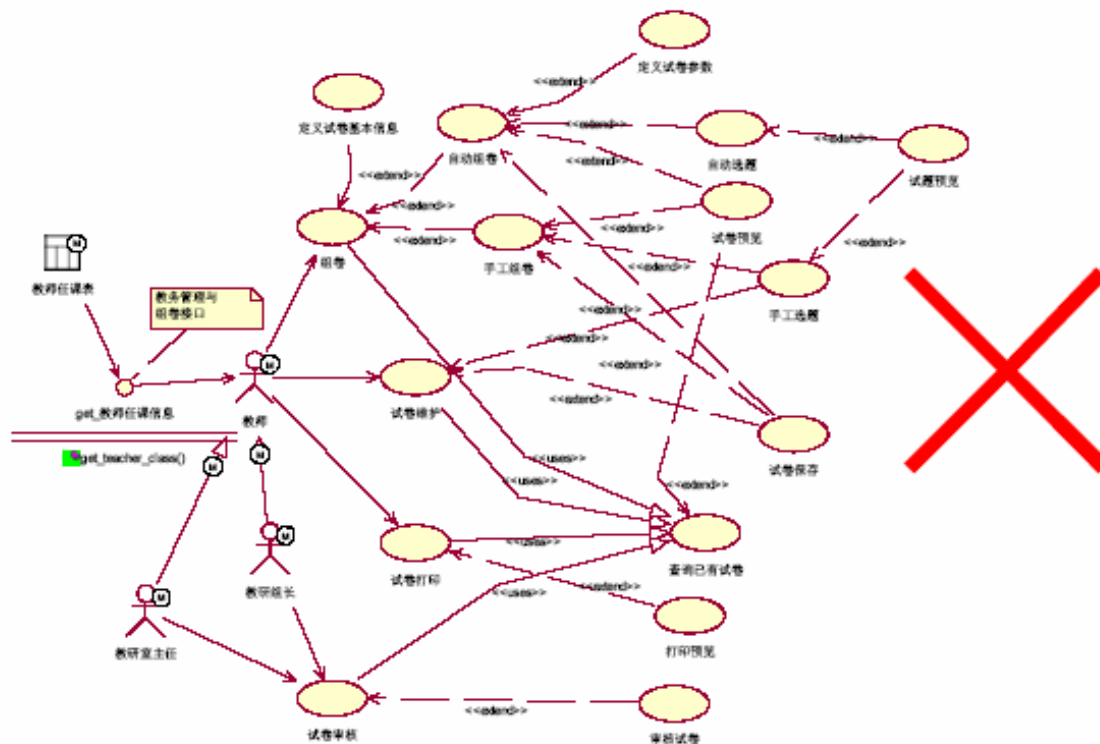
# [ 应该怎么做？——步骤 ]

- 识别Actor;
- 识别用例;
- 书写用例文档
- 通过关系整理用例



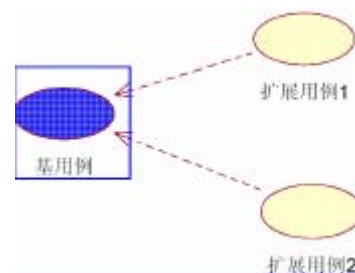
# 识别用例的关系

## ■ 不要滥用用例关系

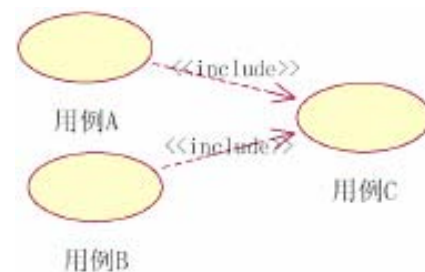


# 识别用例的关系—用例的关系

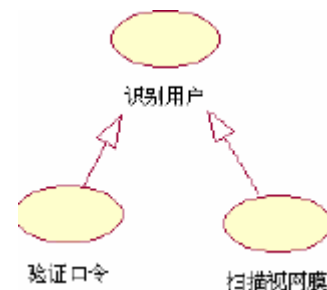
- 扩展：分离次要事件流



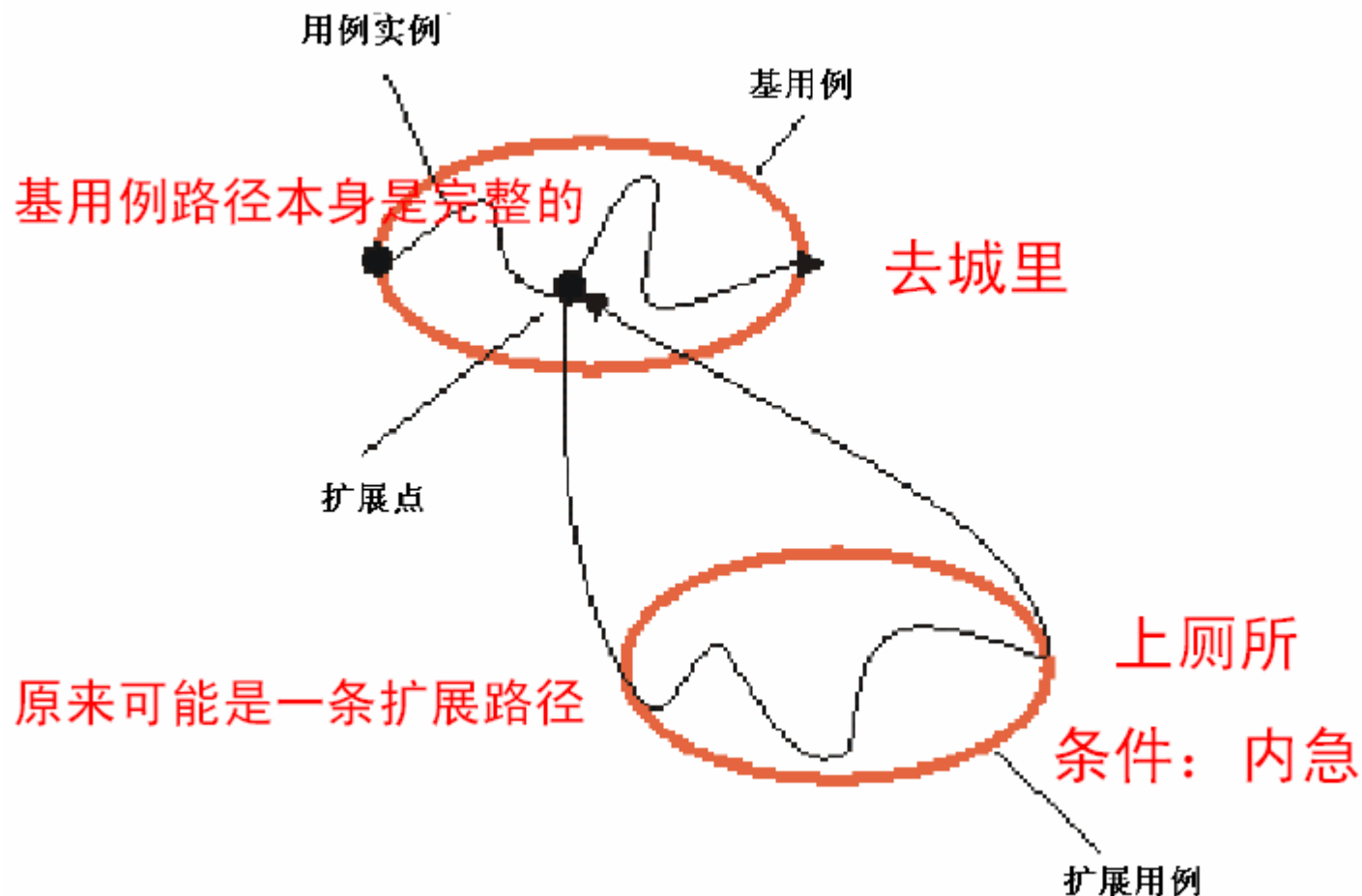
- 包含：提取公共步骤，便于复用



- 泛化：同一业务目的的不同技术实现



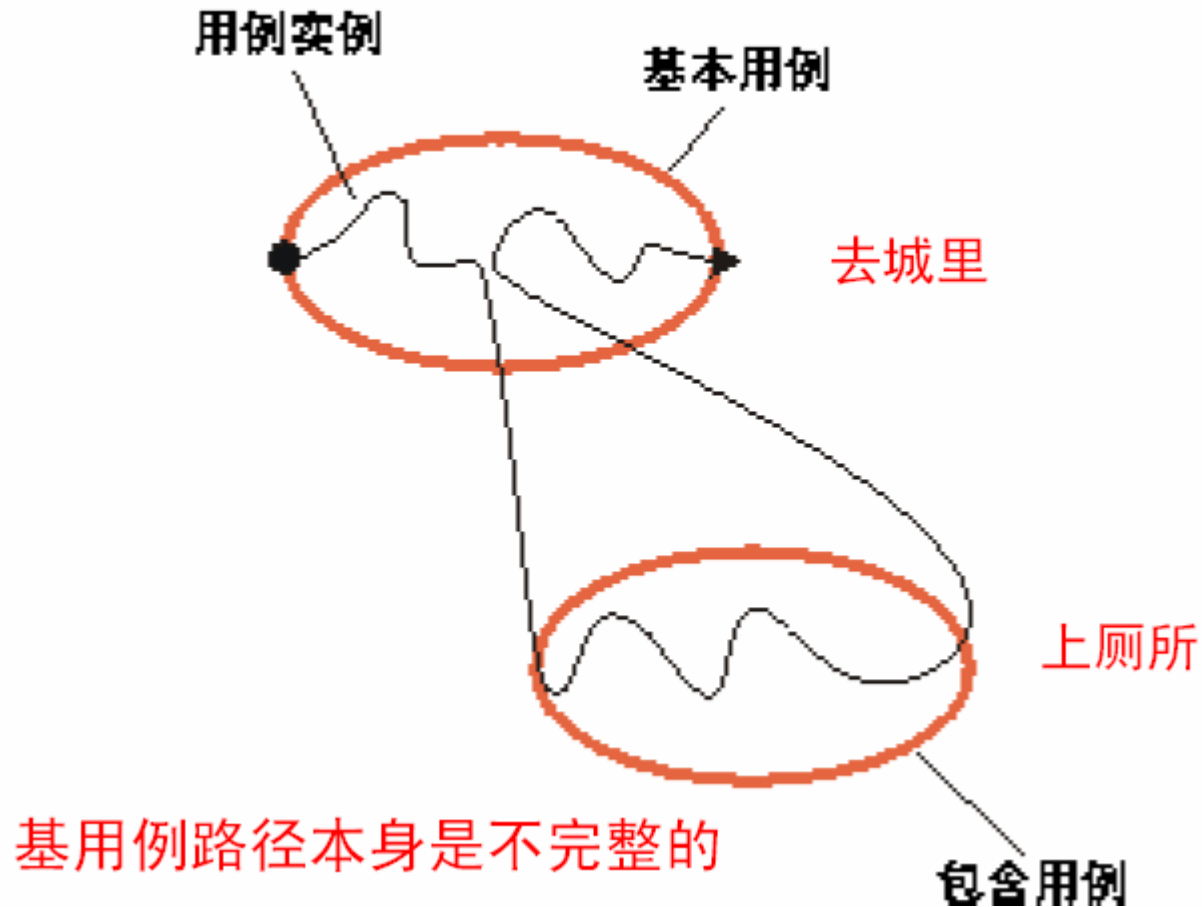
# 识别用例的关系—扩展



# 识别用例的关系—何时使用扩展关系

- 次要事件流路径步骤多
- 次要事件流路径内部还有扩展点
- 次要事件流路径未定或容易变化—分离以“冻结”基本用例

# 识别用例的关系—包含

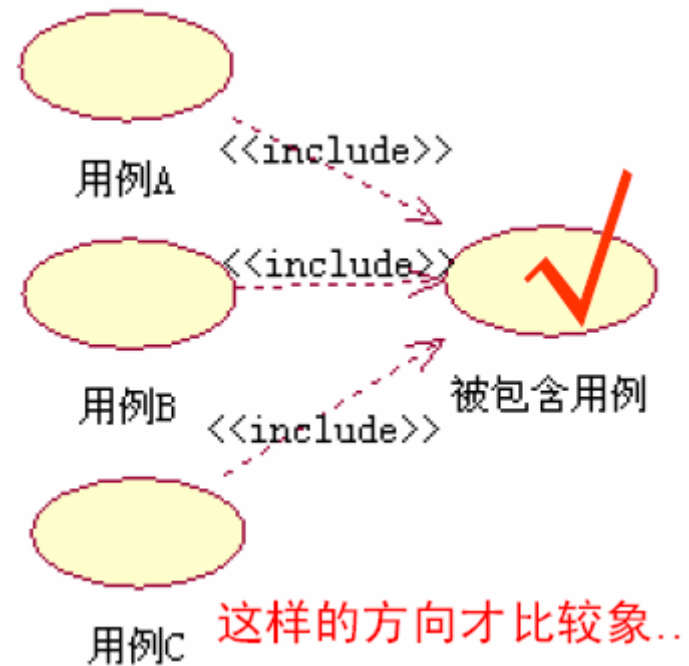
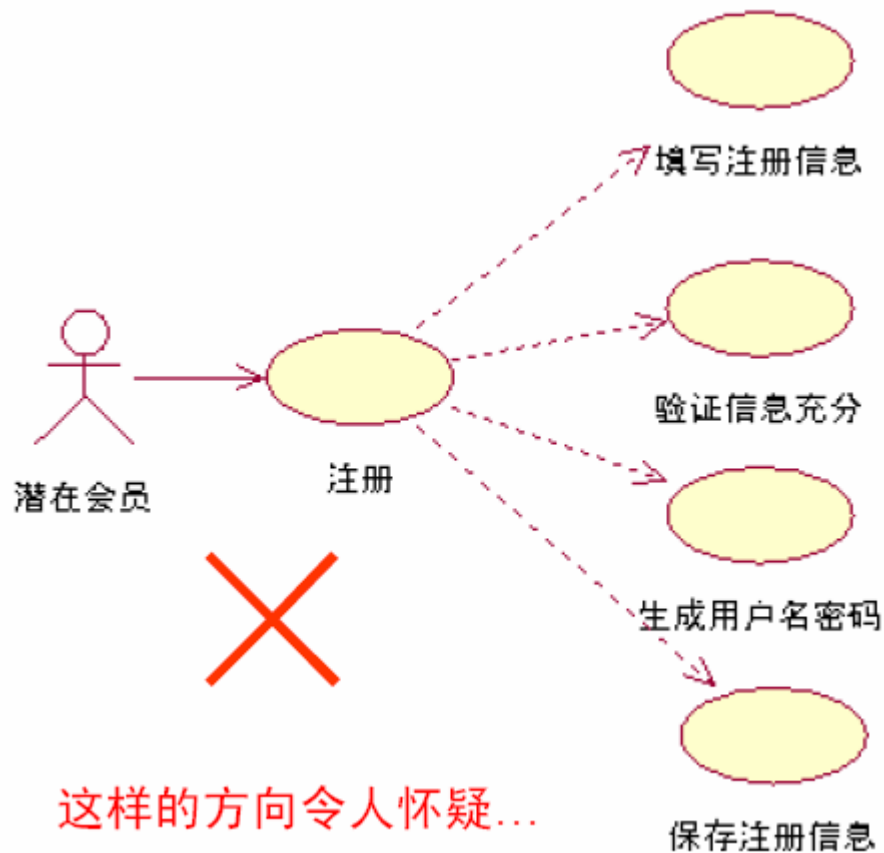




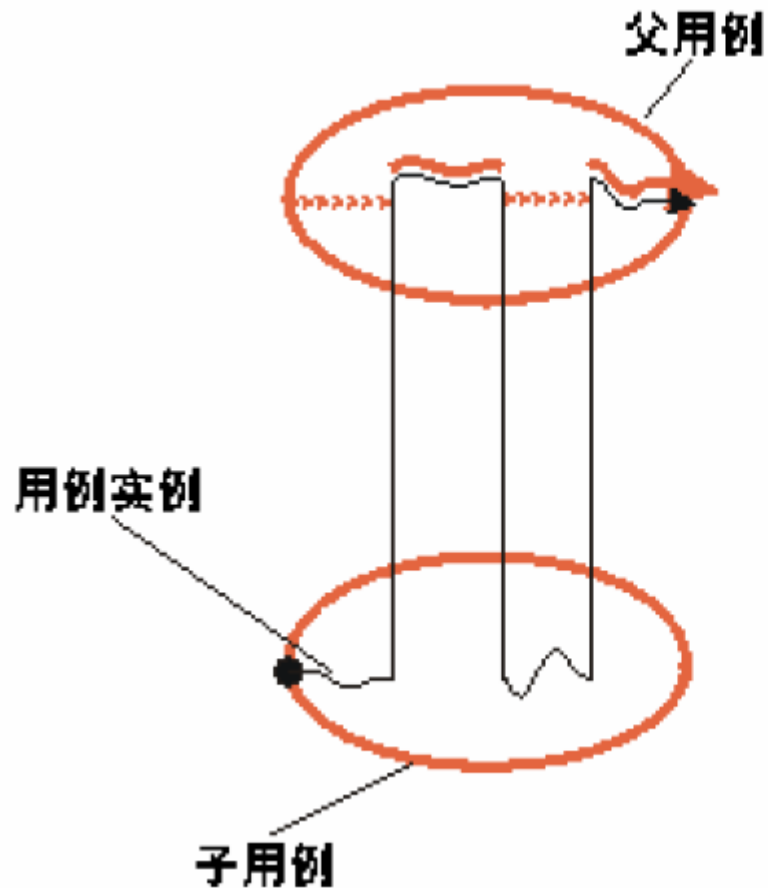
# 识别用例的关系—何时使用包含关系

- 某些步骤在多个用例重复出现，且单独形成价值
- 用例的步骤较多时，可以用Include简化（慎用）

# 包含关系的误用



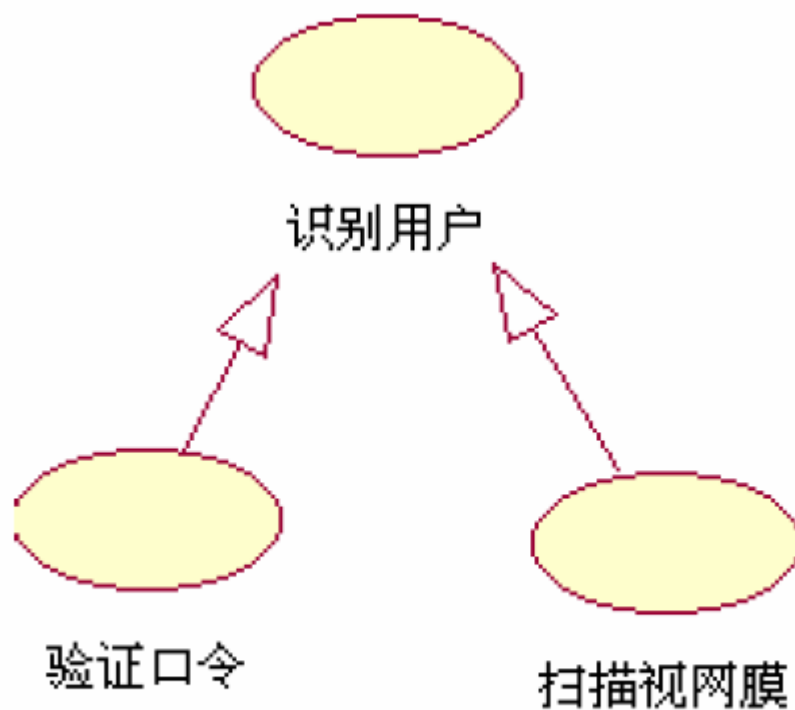
# 识别用例的关系—泛化



子用例继承父用例的一切:

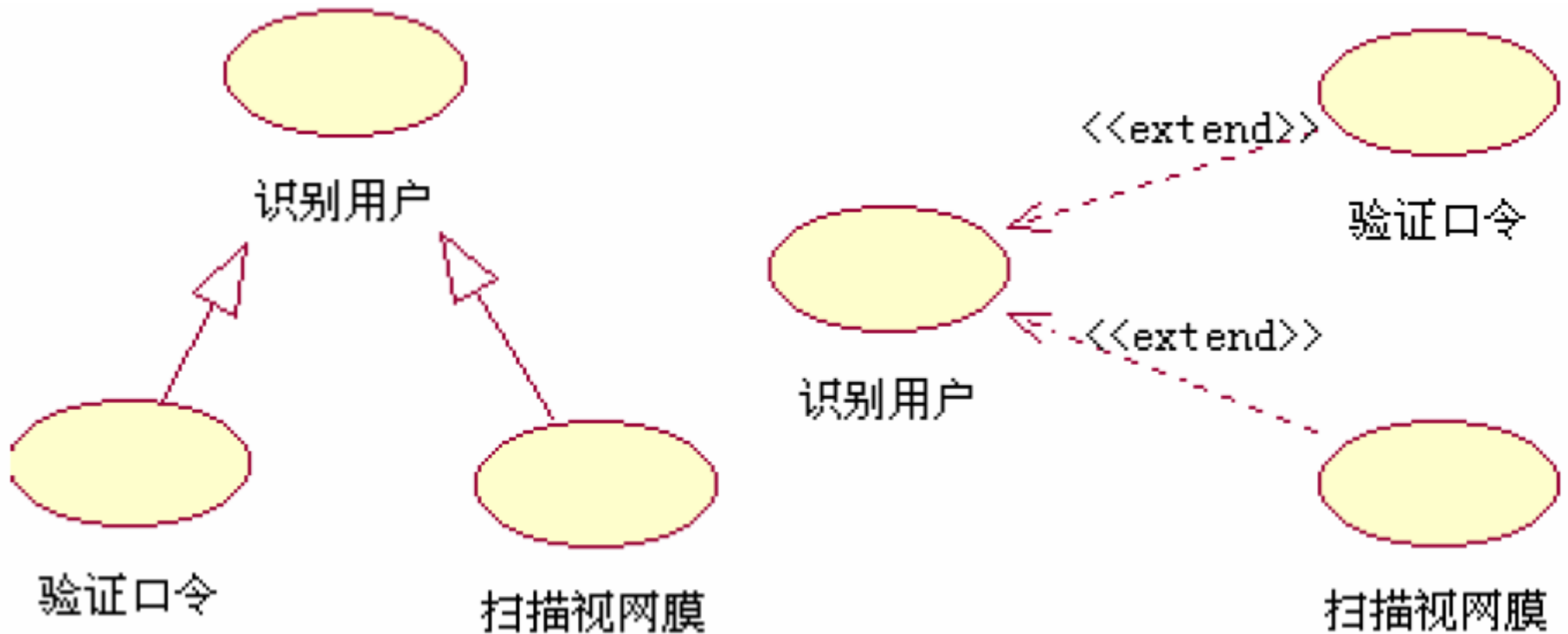
- ❖ 关系
- ❖ 前置条件
- ❖ 后置条件
- ❖ 路径
- ❖ 步骤

# 识别用例的关系—泛化关系举例



同一业务目的不同技术实现

# 识别用例的关系—泛化？扩展？



采用关系不同，文档结构不同

# 识别用例的关系—关系的文档示例

3d. 会员取消订单:

3a1. 会员请求取消一张订单

3a2. 系统删除该订单

3a3. 返回 2

6a. 会员从订单中删除订单项:

6a1. 会员请求从订单中删除某个订单项

6a2. 系统删除该订单项

6a3. 返回 5

6b. 会员修改订单项的购买数量:

6b1. 会员修改某个订单项的购买数量, 请求更改

6b2. 系统修改该订单项的购买数量

6b3. 返回 5

6c. 会员修改订单的送货地址:

6c1. 会员修改订单的送货地址, 请求更改

6c1a. 会员从地址簿中取地址

6c2. 系统修改订单的送货地址

6c3. 返回 5

6d. 会员选择结账:

6d1. 结账

6e. 会员取消订单:

6e1. 会员请求取消一张订单

扩展

UC20 检索螺钉

父用例

UC03

主参与者

潜在会员

前置条件

参与者访问系统

后置条件

参与者查询到所要的零件

基本路径

1. 参与者提交查询条件

2. 系统按查询条件检索螺钉

3. 系统显示搜索到螺钉的编号、类别、价格

4. 参与者选中某个螺钉

5. 系统显示该螺钉的详细信息

扩展点

2a. 系统没有检索到所需螺钉:

2a1. 系统显示“没有找到适合条件的螺钉”

2a2. 用例结束

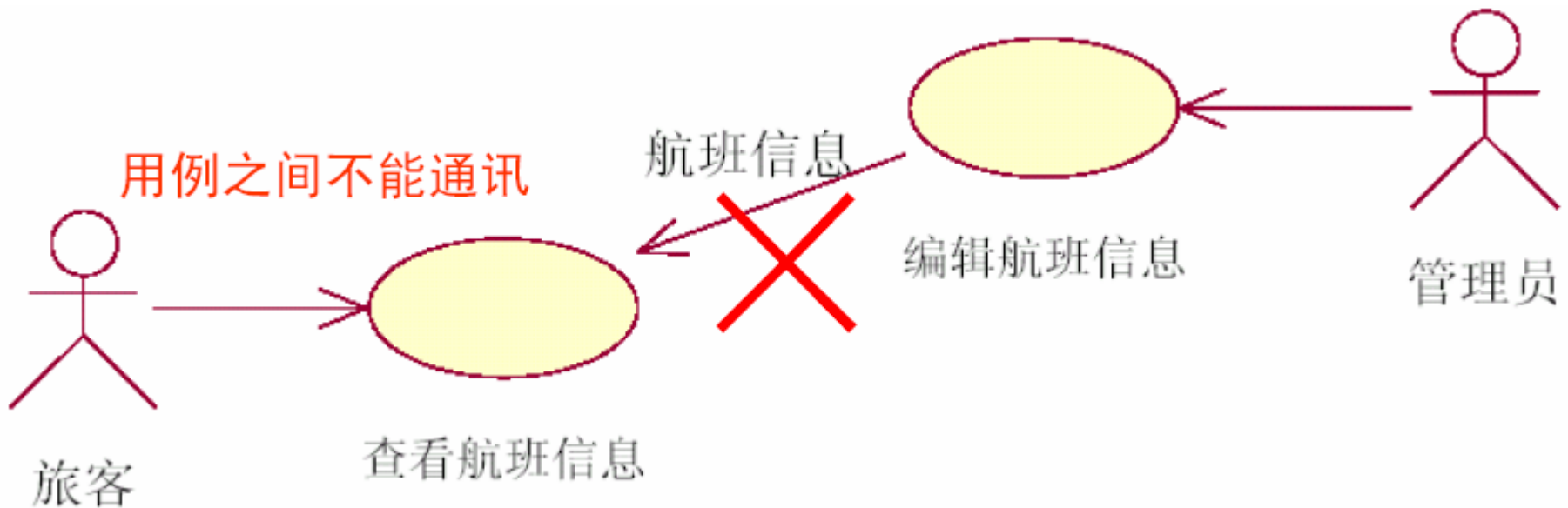
补充说明

.....

泛化

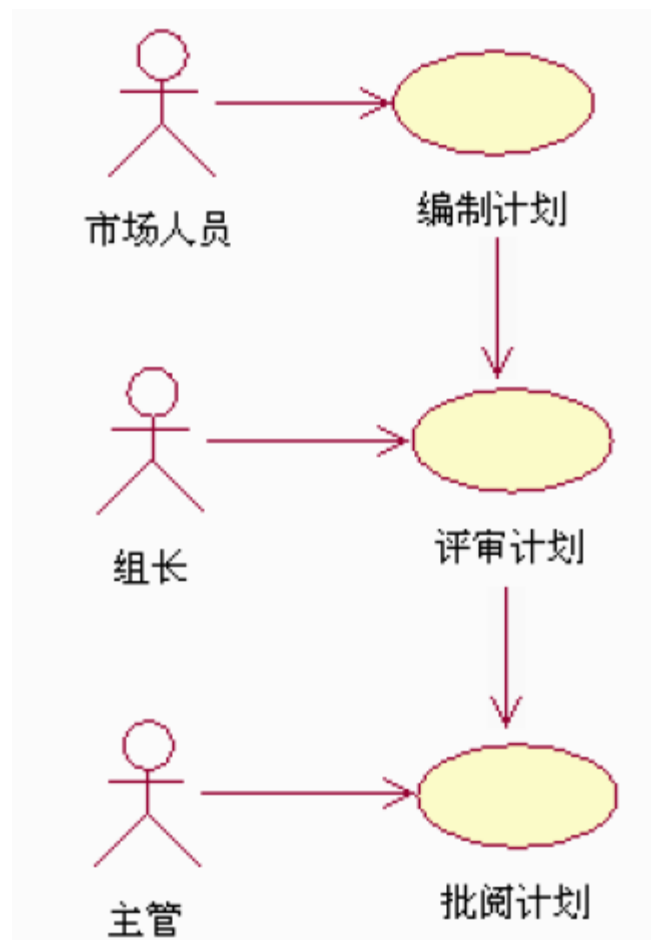
# 识别用例的关系

- 除此之外，不能有别的关系



# 识别用例的关系

- 小心“羊肉串”！
- 用例是执行者一份一份地和系统签订的契约





# 用例参考书籍

