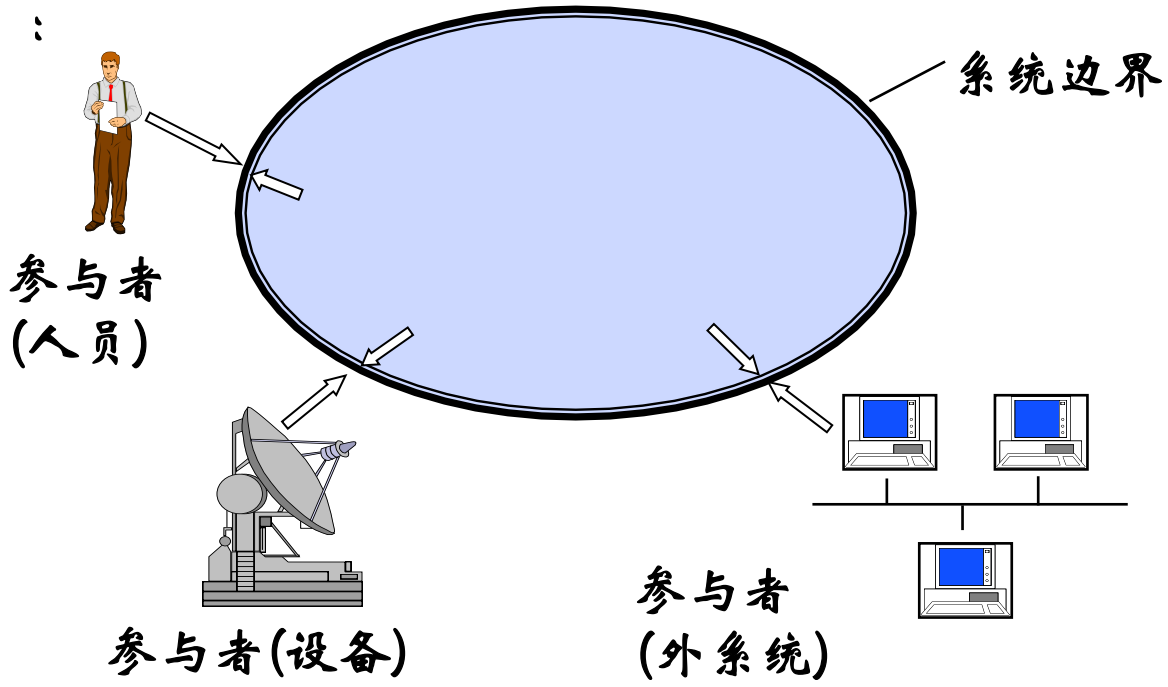
用例精讲

Use Case Modelling Details

清华大学软件学院 冯乐桐

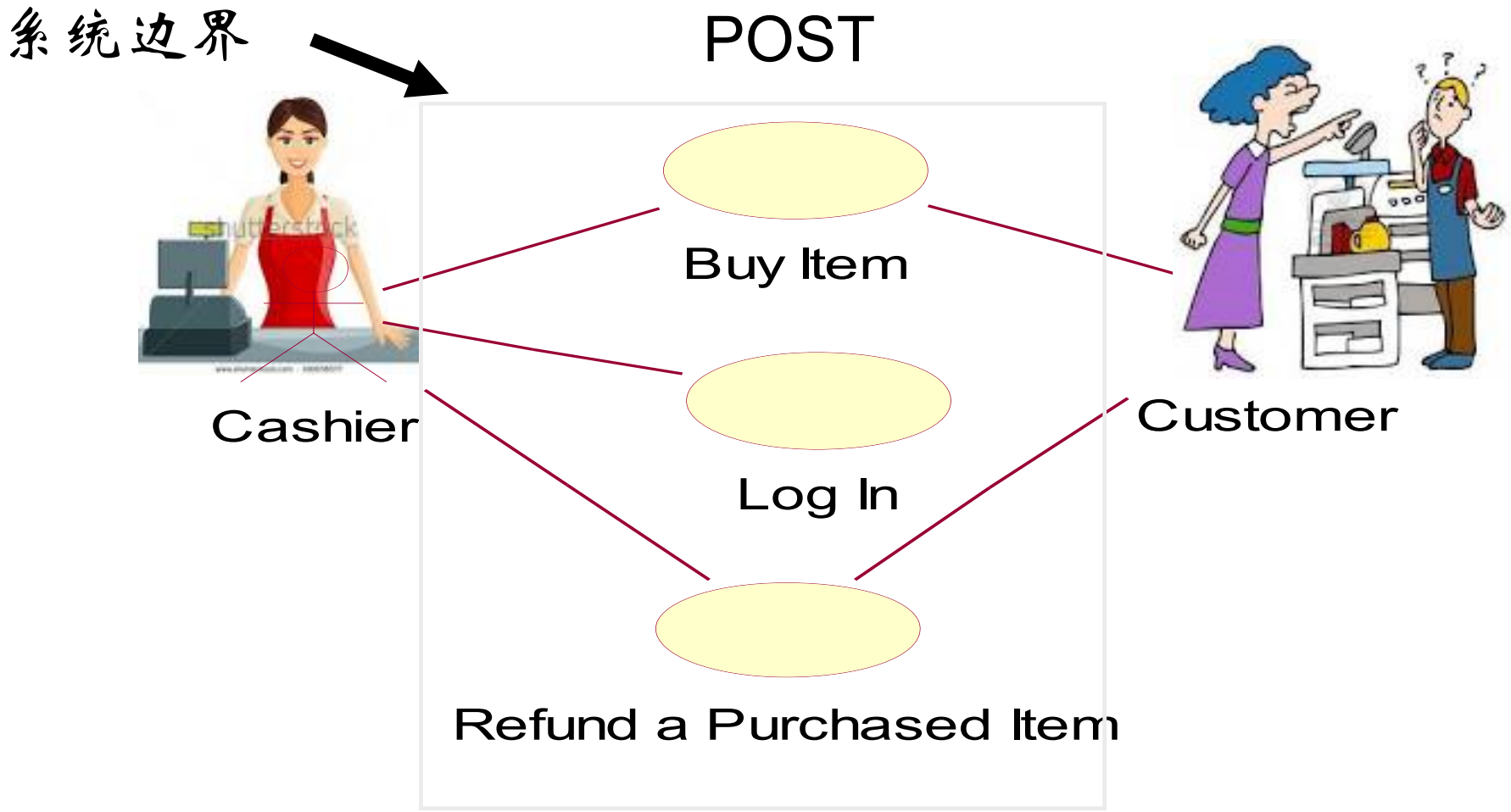
一、设定系统边界

- **系统边界**：一个系统所包含的所有系统成分与系统以外各种事物的分界线
- 系统边界会对用例以及Actor的定义有所影响
- 考虑用于零售店销售管理的系统的用例图：
  - 记录销售及付款情况的软硬件集成系统
  - 包括硬件设备，如计算机、条码扫描装置
  - 摆阔运行在系统上的软件
  - 系统目标包括：
    - 自动收款
    - 快速准确的销售情况统计及分析
    - 自动的库存管理



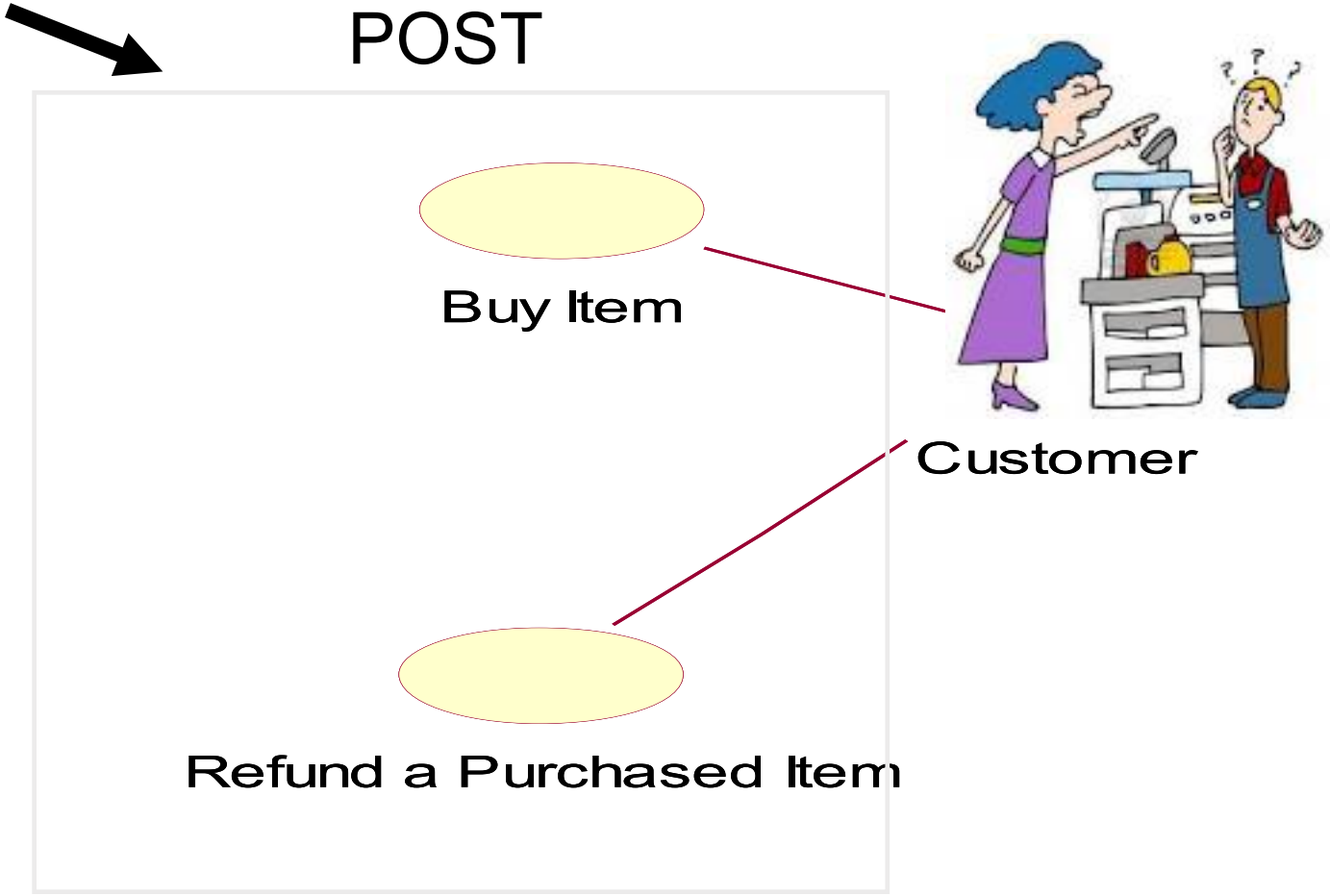
（本图选择自北大麻志毅老师教材）

系统边界定义之一

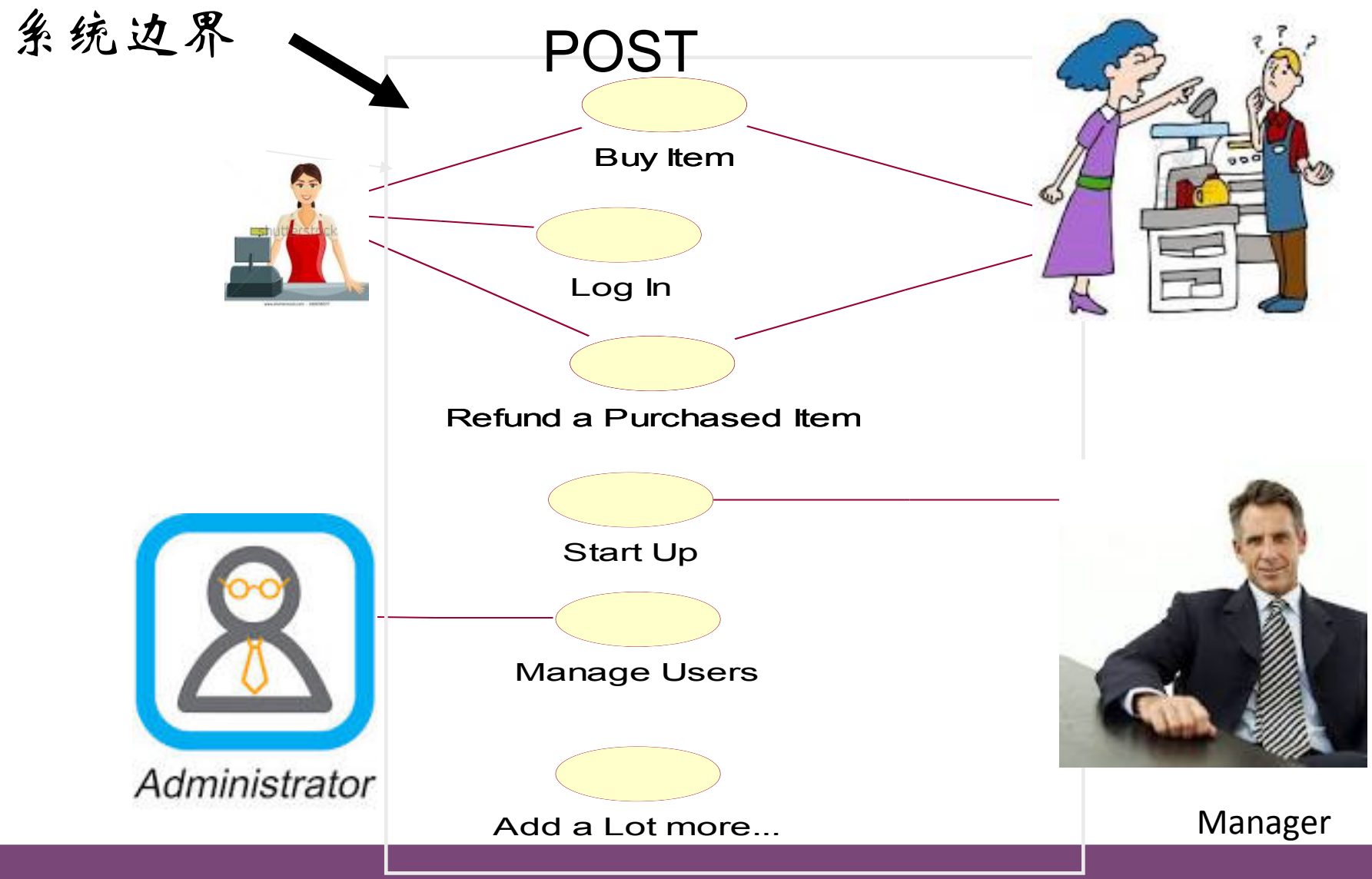

系统边界
POST
Buy Item
Log In
Refund a Purchased Item
Cashier
Customer

系统边界定义之二

系统边界



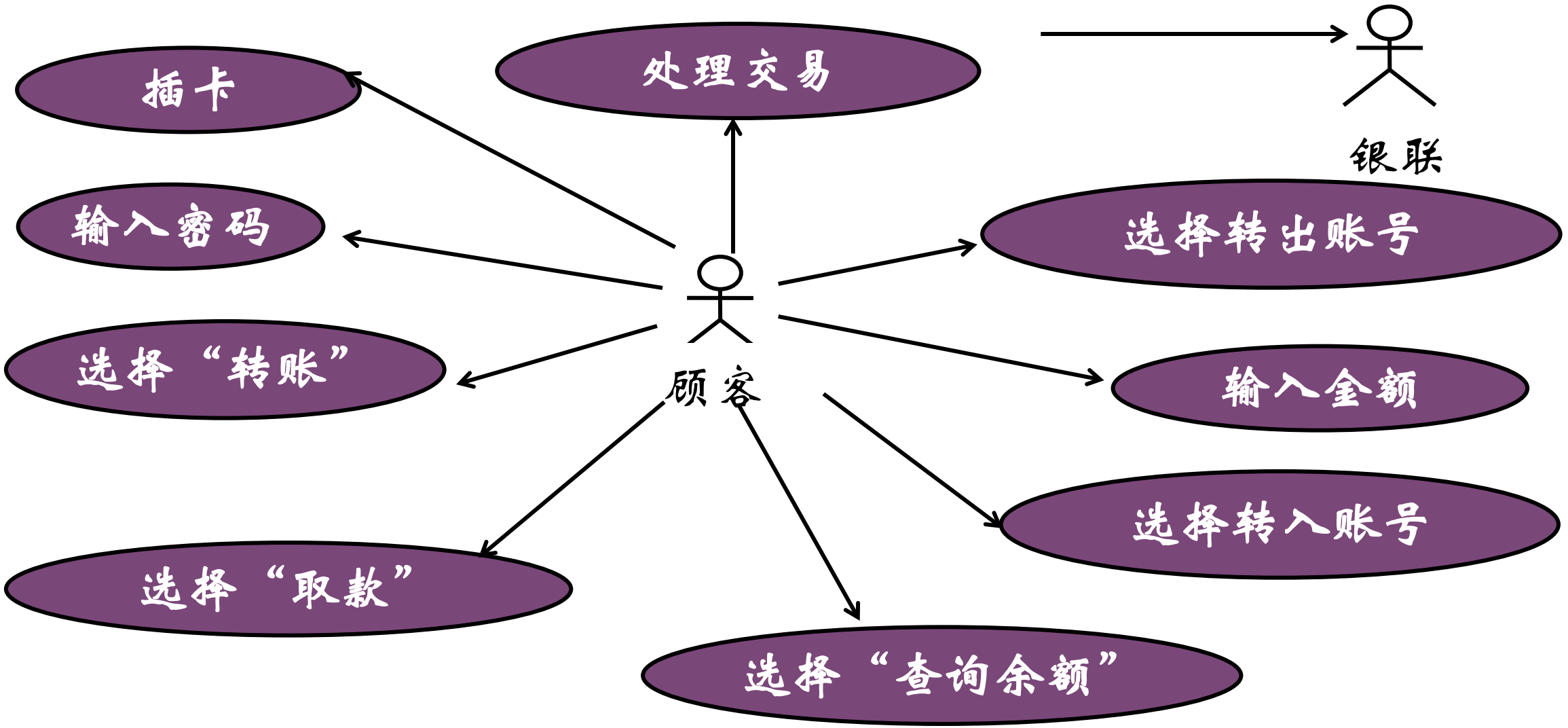
系统边界定义之三


系统边界
POST
Buy Item
Log In
Refund a Purchased Item
Start Up
Manage Users
Add a Lot more...
Administrator
Manager

二、不要把用例定义成功能分解

- 功能分解：将问题分解为粒度小，独立的部分。
  - 不同的模块协同工作，体现系统的功能。
    - 通常，一些功能分解并没有实际的意义。Often do not make sense in isolation.
- 用例：
  - 不是功能分解的过程！
  - 综合所有功能一起描述系统如何使用。
  - 需要包含语境信息

功能分解：一个例子



走出功能分解：正确的用例建模

如何避免功能性分解？

问题现象

- 非常细小的用例
- 用例过多
- 没有实际价值的用例
- 通过底层操作进行命名
  - “操作”+“对象”
  - “功能”+“数据”
  - 例如：“插入卡片”
- 很难理解整体模型

修改思路：

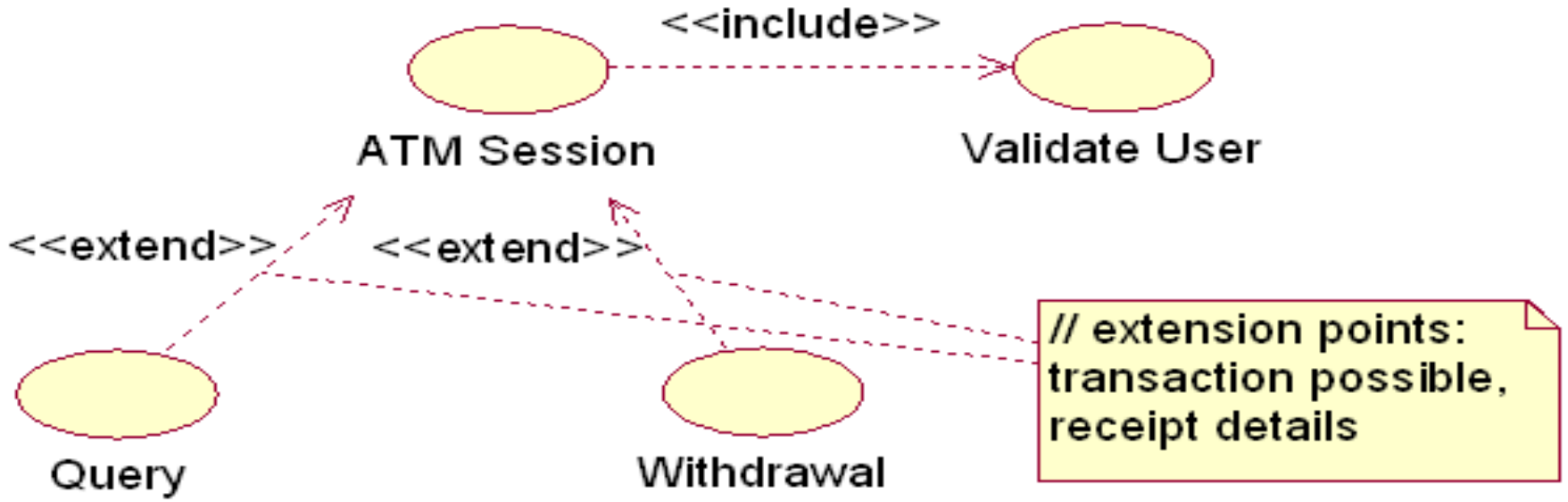
- 寻找更大的应用场景

  “为什么要构建这个系统？”

- 从一个用户的角度出发
  - “用户希望达到什么目的？”
  - “这个用例可以满足谁的目标？”
  - “这个用例的意义是什么？有什么价值？”
  - “这个用例背后的用户故事是什么？”

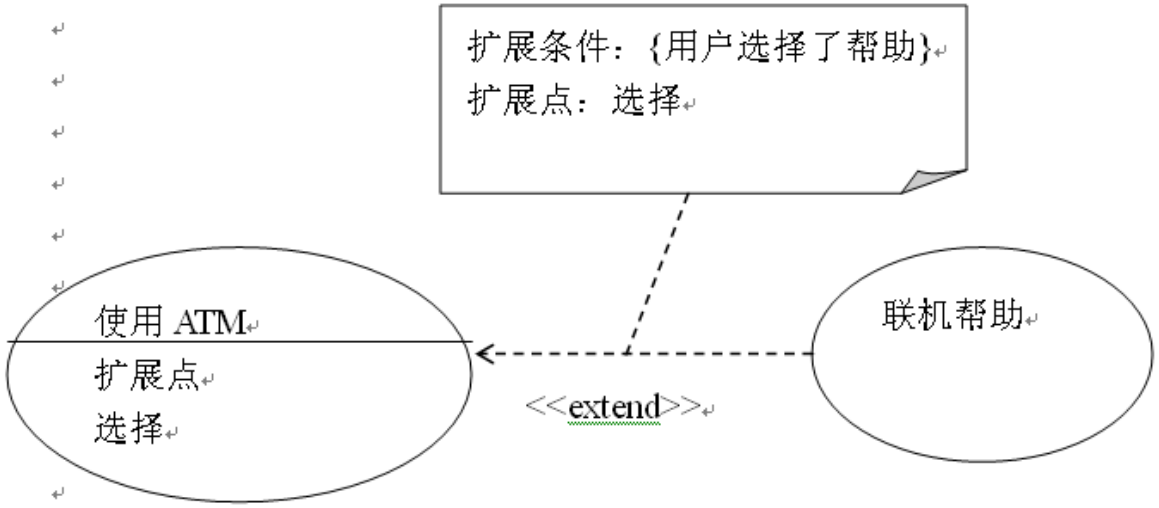
三、何时使用包含关系？

- 当多个用例有共享行为时，使用包含关系
- 为共享行为单独创建用例，被相关用例“包含”
- 例如：
  - 估价
  - 权限检查

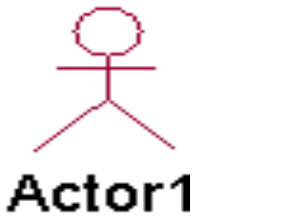


四、何时使用扩展关系？

- 一个用例与另外一个用例近似，只有少许额外的活动
- 将代表普遍或基本行为的情况定义为一个用例
- 将特殊的、例外的部分定义为扩展用例

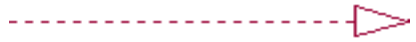


五、用例图中的主要图标



注释

注释连接



Realize

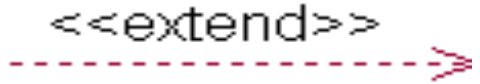
Association

关联

use-case realization

Generalization

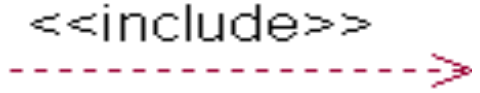
泛化



extend use case

Dependency

依赖



include use case

说明：UML中不使用颜色来作为图形语义的区分标记。