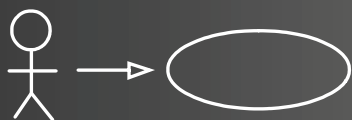


UML全程实作

类图

Think



<http://www.umlchina.com>

核心 workflow

*愿景

*业务建模

选定愿景要改进的业务组织

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

书写用例文档

提升
销售

*分析

类图

序列图

状态图

*设计

建立数据层

精化业务层

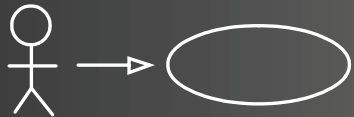
精化表示层

降低
成本

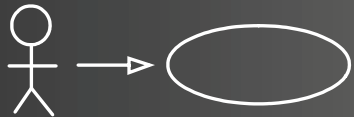
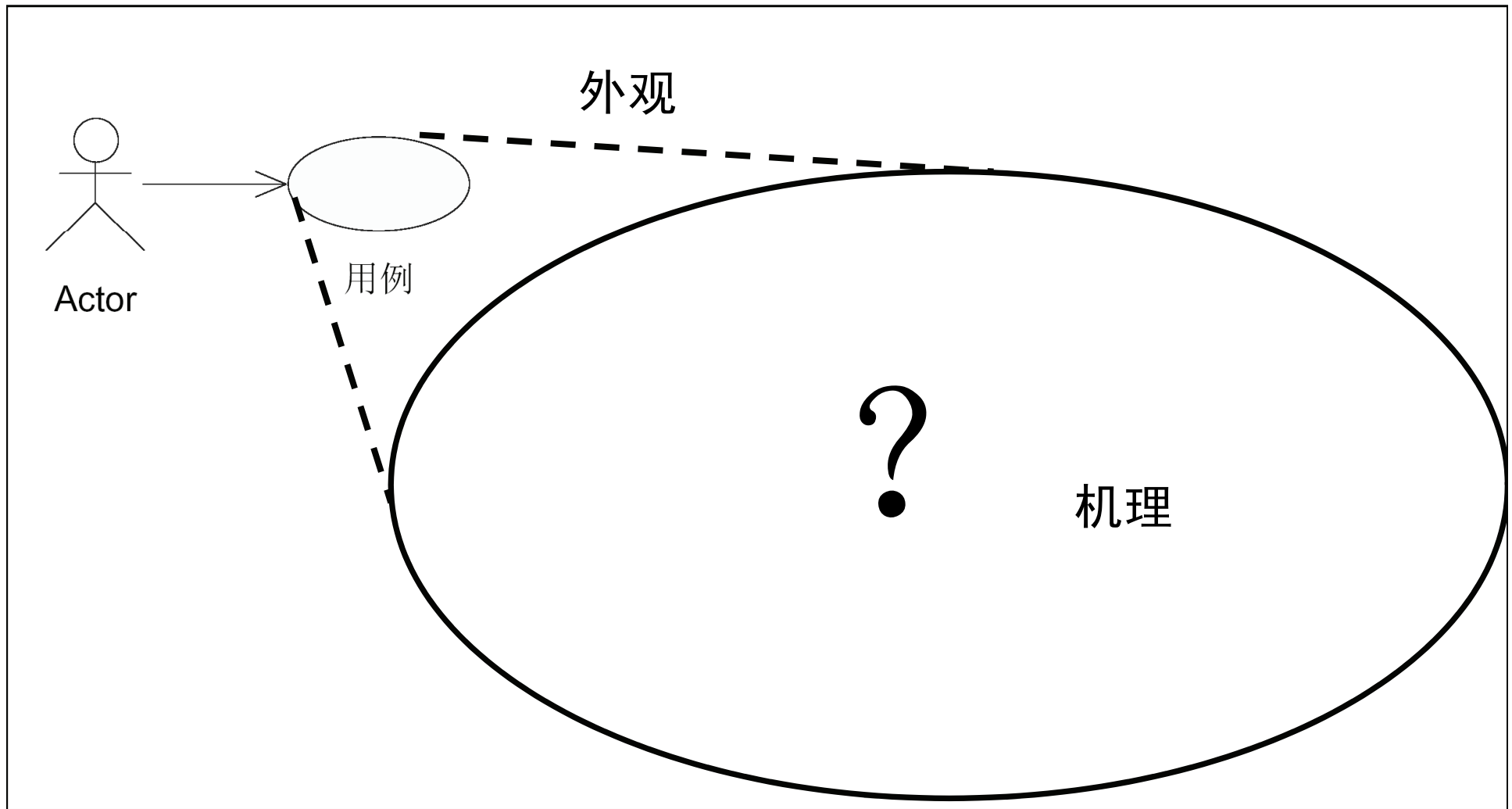


分析和设计

- 分析：提炼核心域知识
- 设计：添加非核心域知识



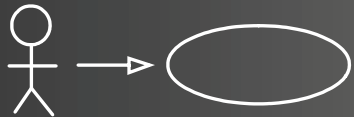
外观和机理



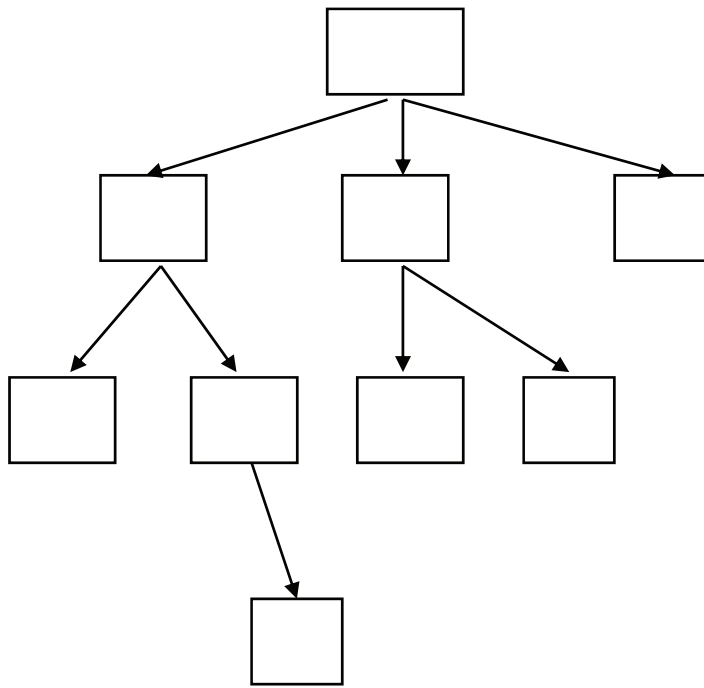
分解的必须

7 ± 2

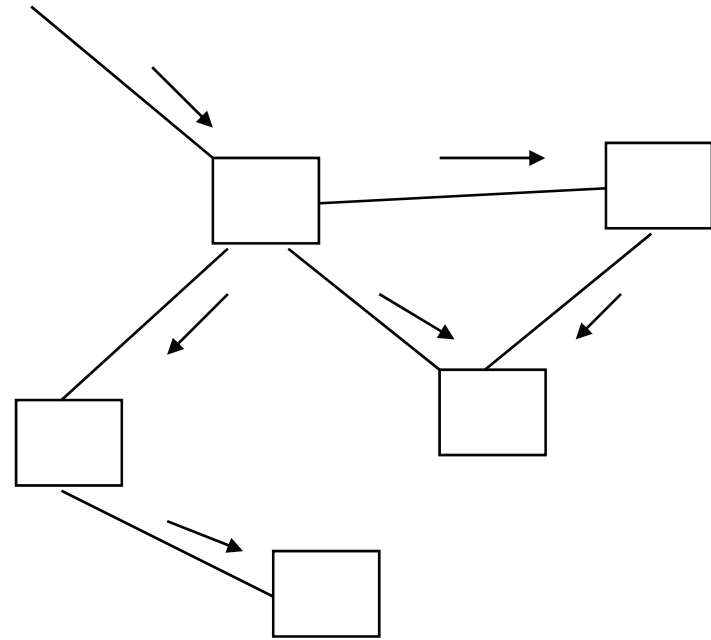
George Miller: 人脑的把握度有限



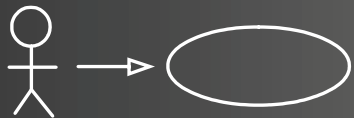
面向过程和面向对象



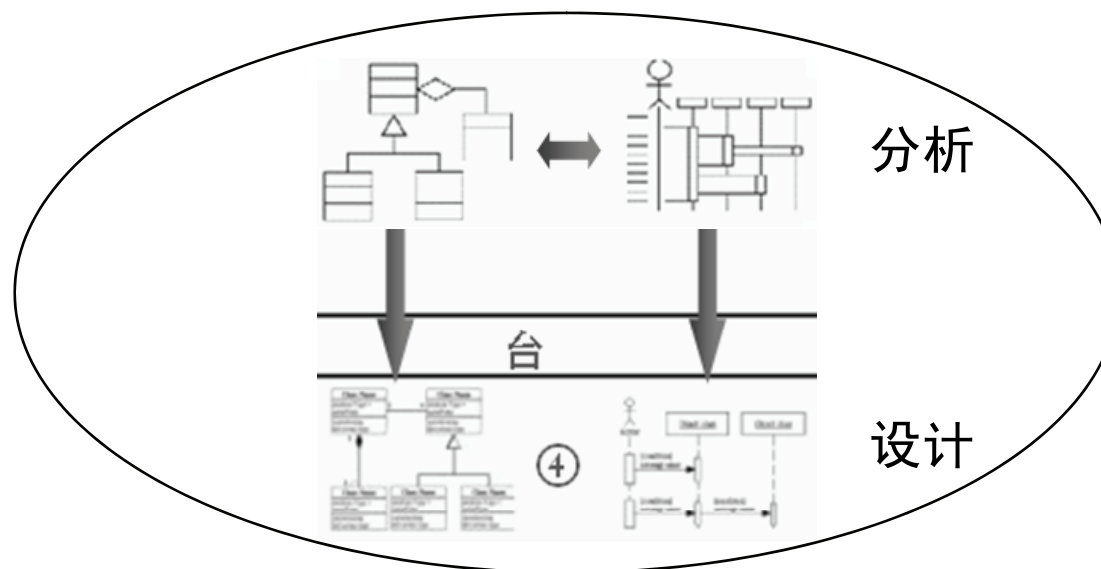
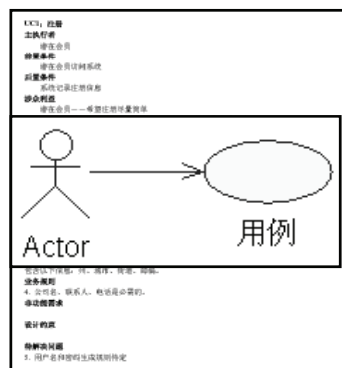
分解



协作

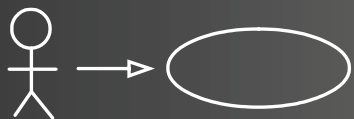


面向对象的分析设计



类、对象、关系、责任、协作

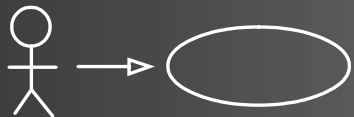
把不同变化趋势的知识分离



<http://www.umlchina.com>

对象

对象具有状态、行为和标识



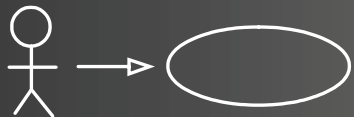
对象

详见状态图部分

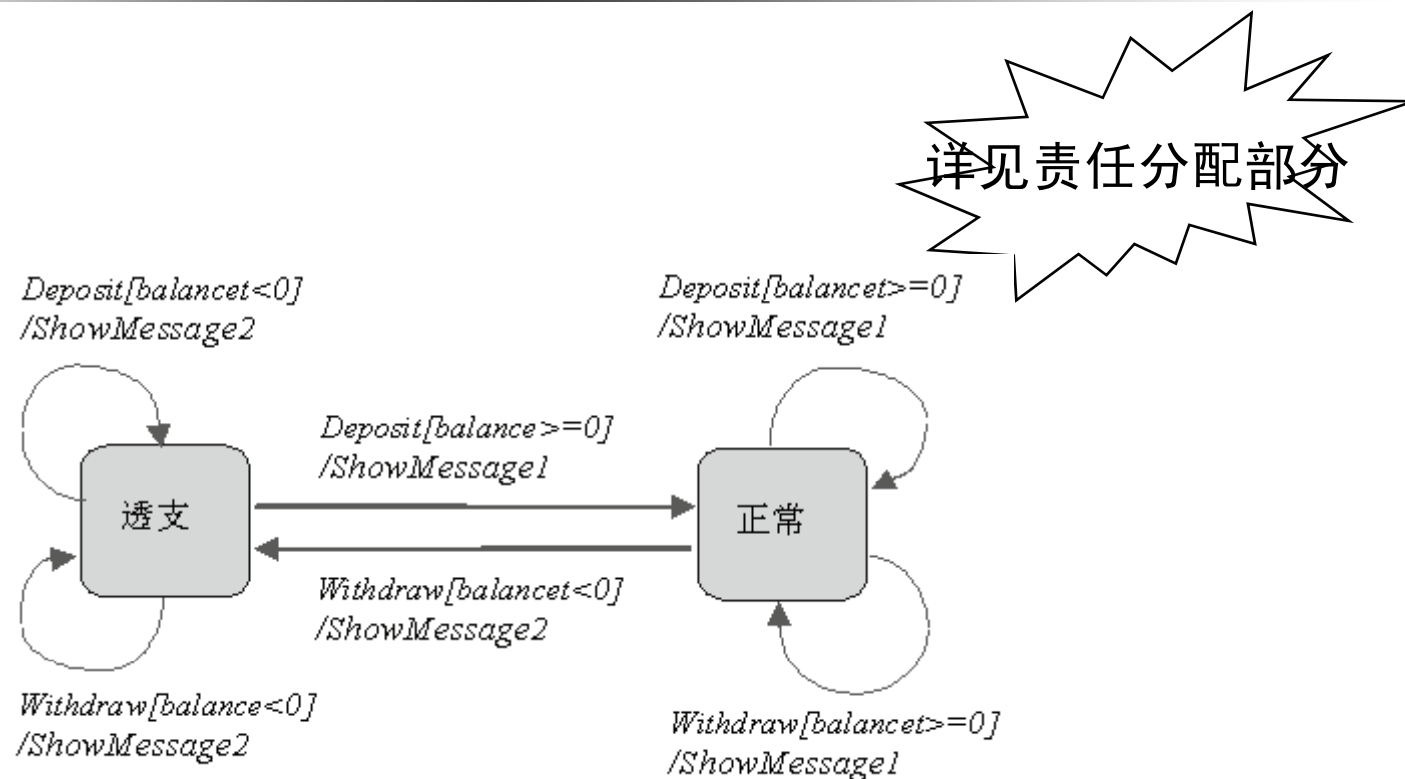
金贝贝:人	
年龄	= 4
身高	= 1.1
学历	= 1
婚姻状况	= 1

- ❖ 年龄: ...6, 7, 8, 9...17, 18...21, 22...34, 35...
- ❖ 身高: ...1.3, 1.4, 1.5, 1.6, 1.7, 1.8...
- ❖ 学历: ...高中, 专科, 本科, 硕士
- ❖ 婚姻状况: ...未婚, 已婚, 离异...

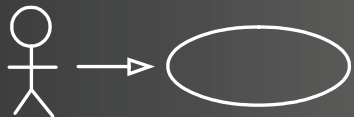
状态: 当前属性值的组合, 行为的累积结果



对象



行为：对象根据状态和接收消息作出的反应

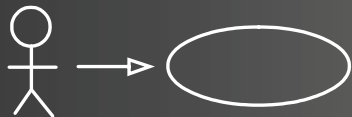


对象



对象在哪里？

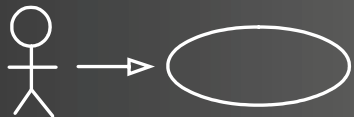
标识： 和其他对象相区分



类



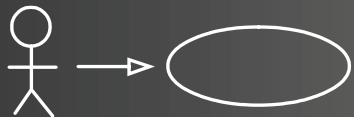
共享一个公用结构和公用行为的对象集合



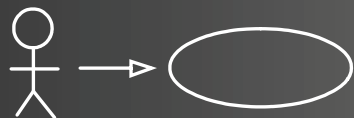
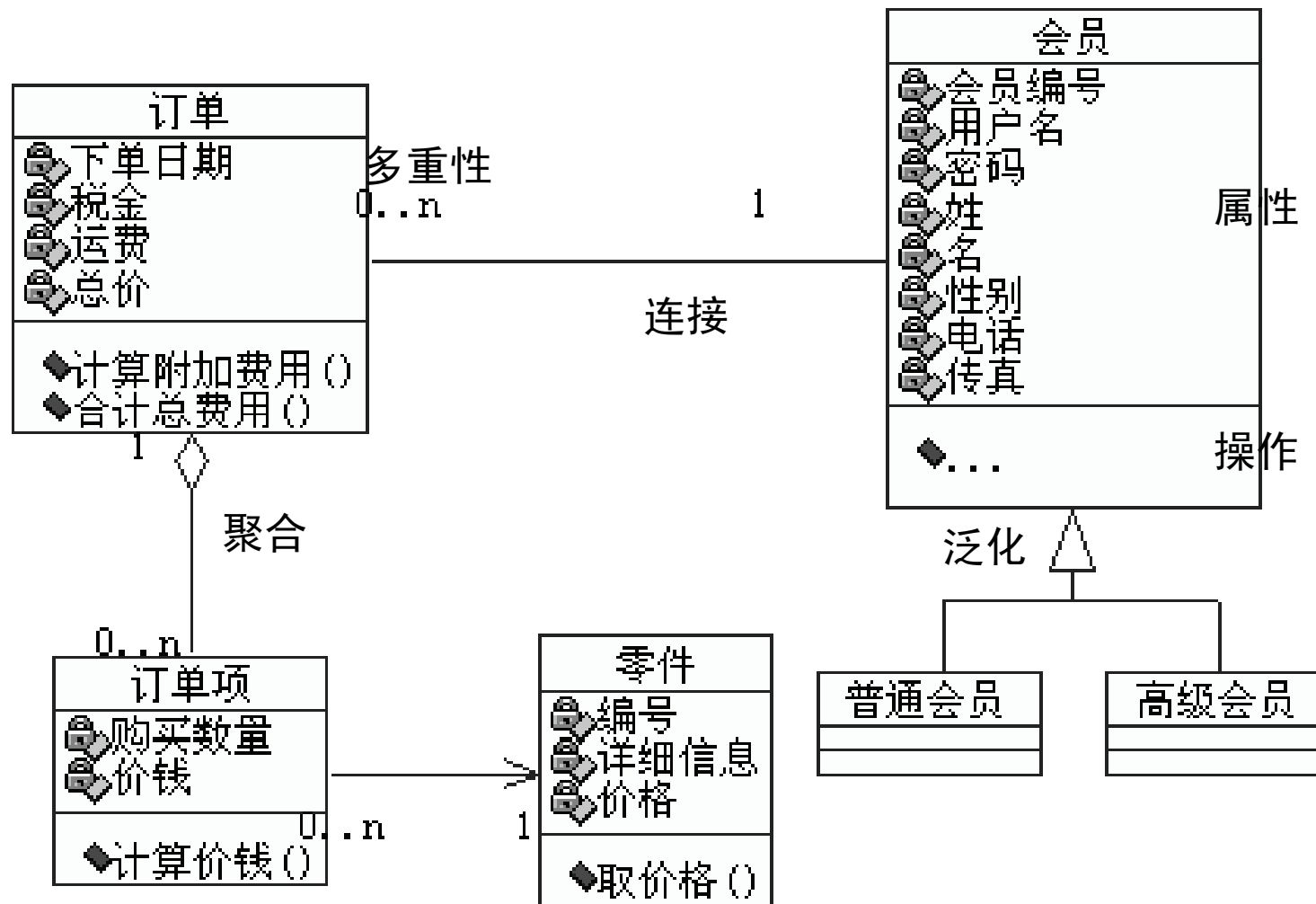
类



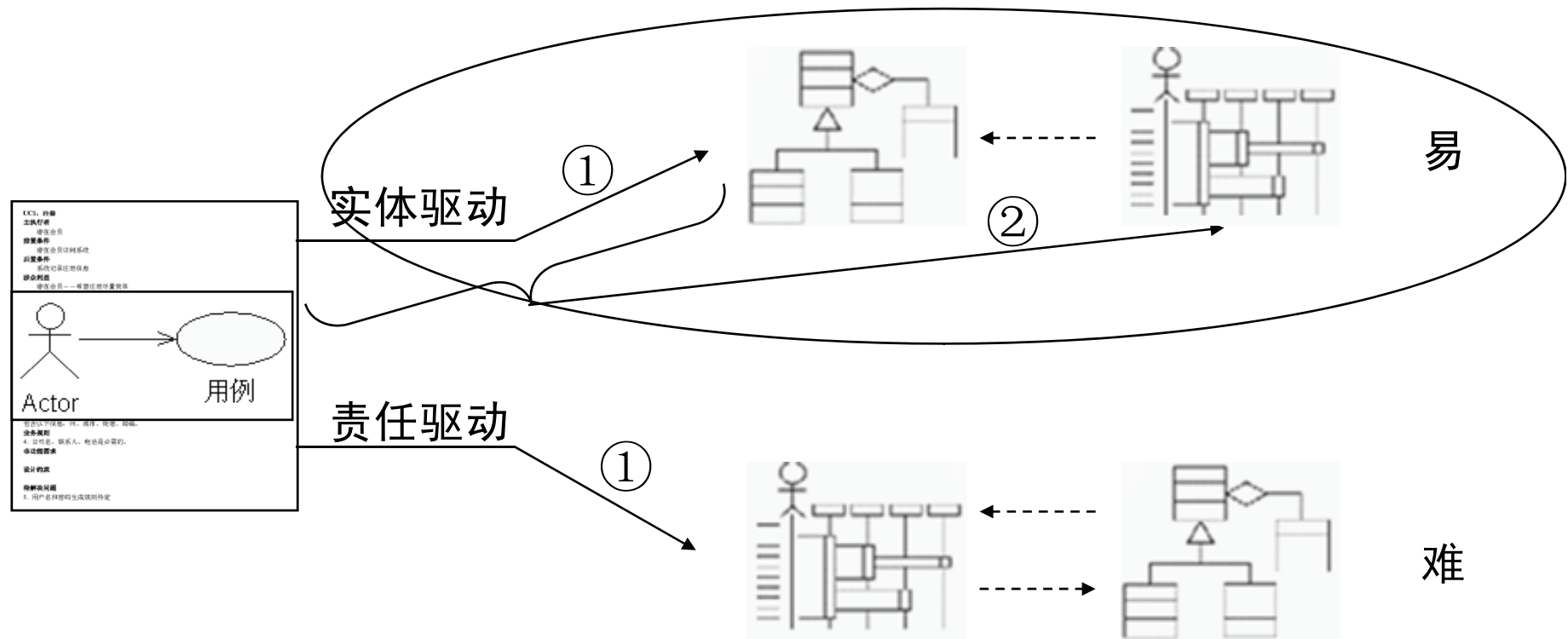
并不容易——这个对象的类叫什么？



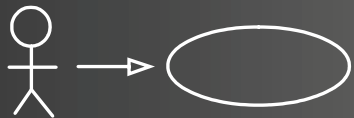
类图解说



路线



两条分析路线



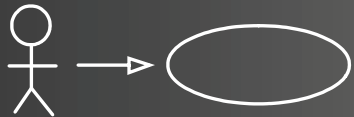
步骤

 识别类和属性



 识别泛化

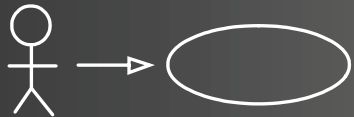
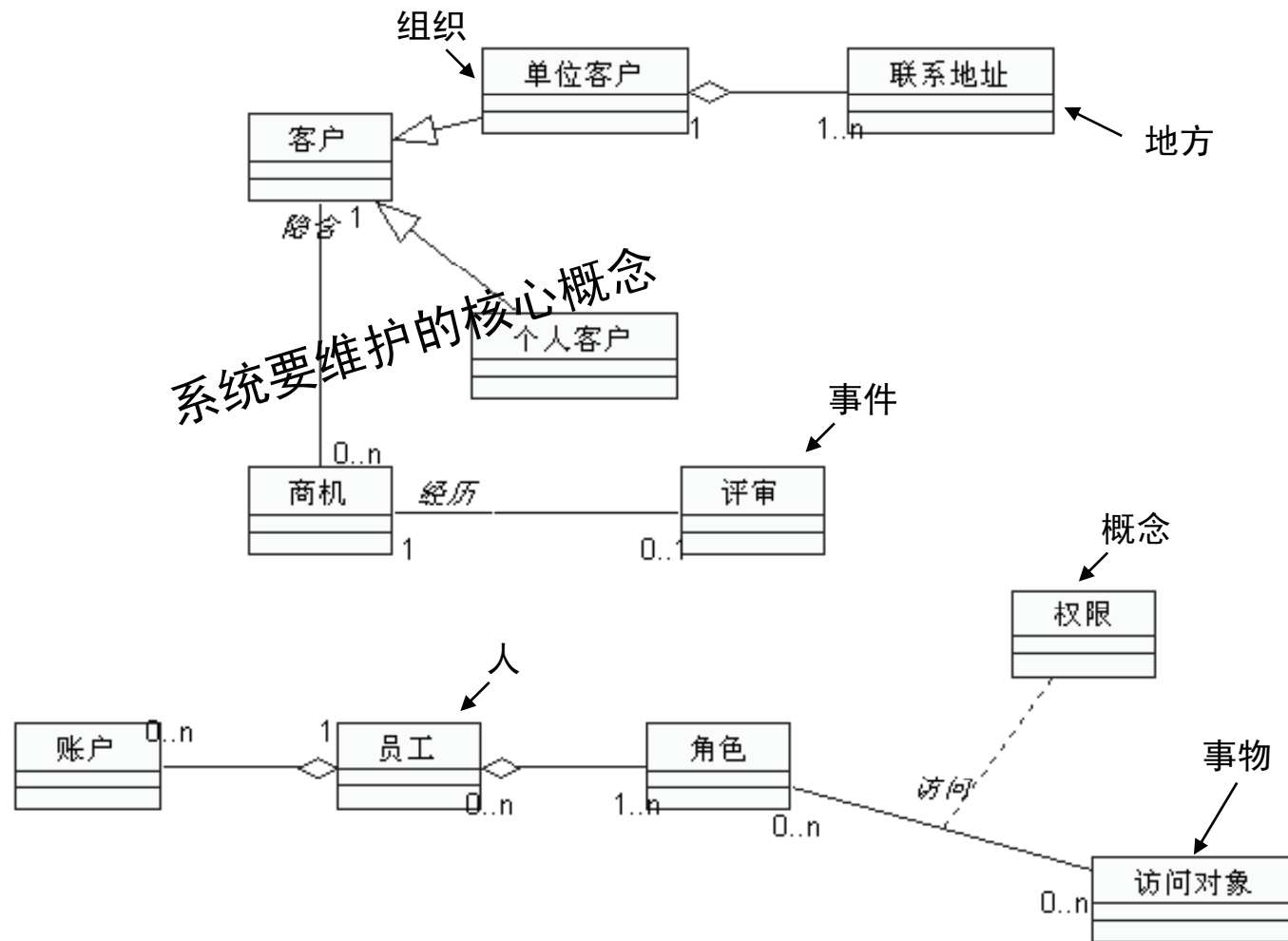
 识别关联



[illegible]

-

识别类和属性



基本路径

1. 当到达时间周期时，系统请求 BOSS 系统导出欠费清单
2. 系统把高价值号码和非高价值号码分类
3. 系统请求 MASA 系统取惯用号码
4. 系统分单，保存分单结果

扩展

1a. BOSS 系统无反馈：

1a1. 系统发送告警信息给帐务人员

3a. MASA 系统无反馈：

3a1. 系统发送告警信息给帐务人员

字段列表

1. 欠费清单=客户标识+手机号+欠费起始月+欠费金额+客户分类+客户状态

1. 客户分类={高价值|非高价值}

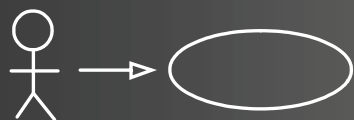
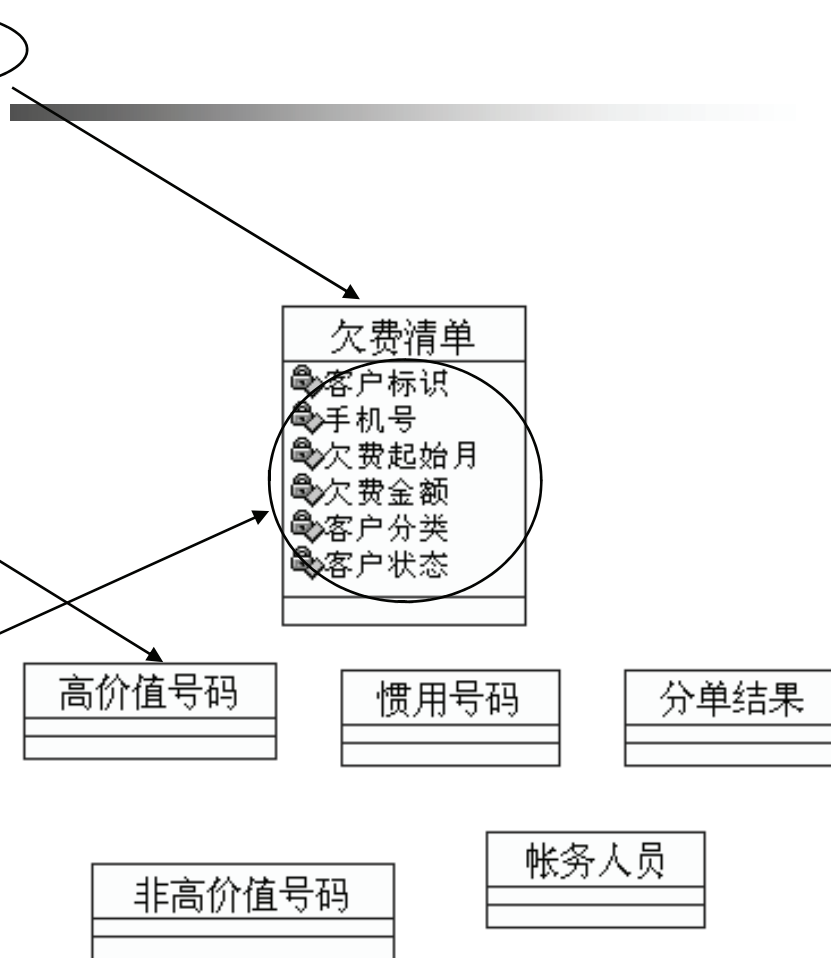
1. 客户状态={在用|停机.....}

3. 惯用号码=客户标识+手机号+惯用号码

4. 分单结果=追费员工号+客户标识

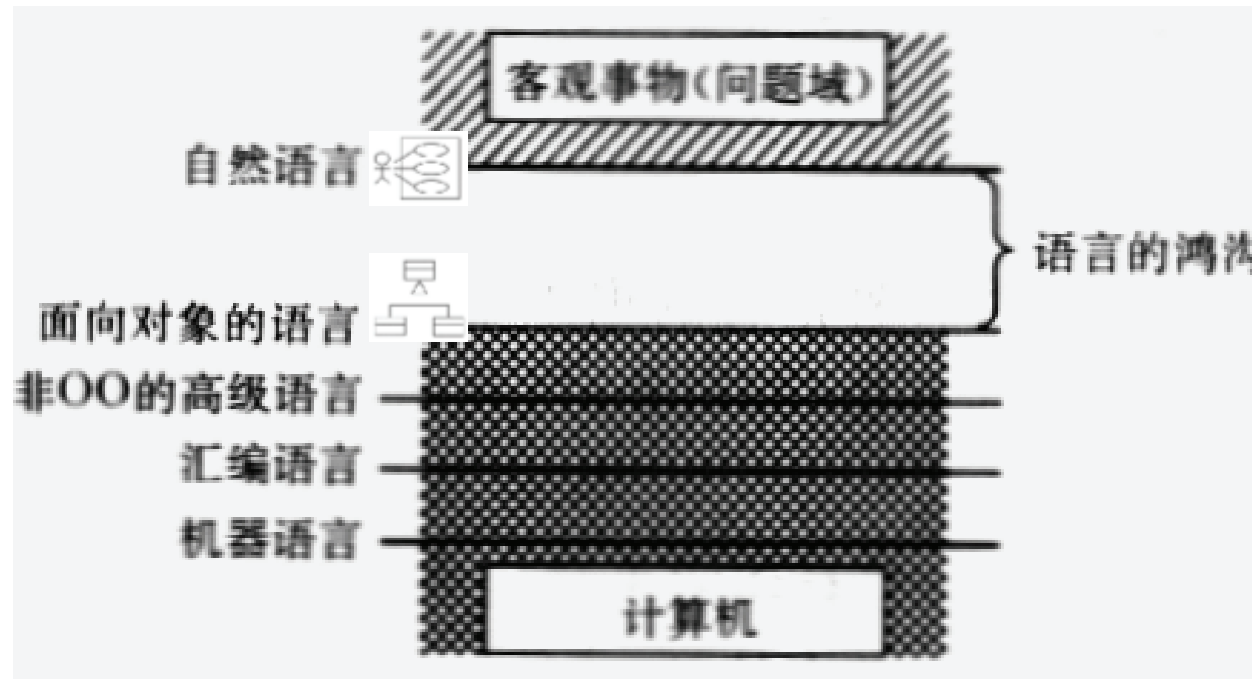
业务规则

4. 先按照起始月，再按本异地，使分配差最小。

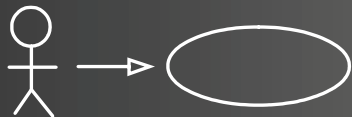


识别类和属性

——有没有秘诀？

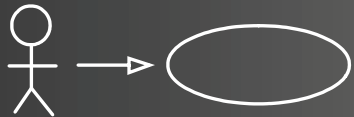


没有寻找类的简单算法，靠分析员的头脑跨越鸿沟



识别类和属性

- 需要识别的是实体类（核心域概念）
 - 电梯调度：楼层、电梯、行进方向...
 - 芯片加工：Wafer、Cassette、Recipe...
- 对象在大脑中运行



识别类和属性

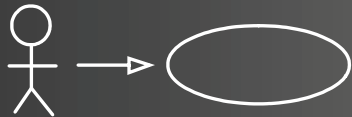
➤ 很多名词并非系统要保存的实体

ATM卡、钞票、收据...

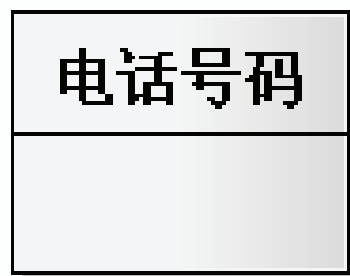
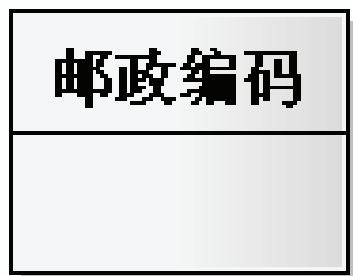
发货单、账单、统计报表....

➤ 事件可能也是类

评审、雇用...



识别类和属性

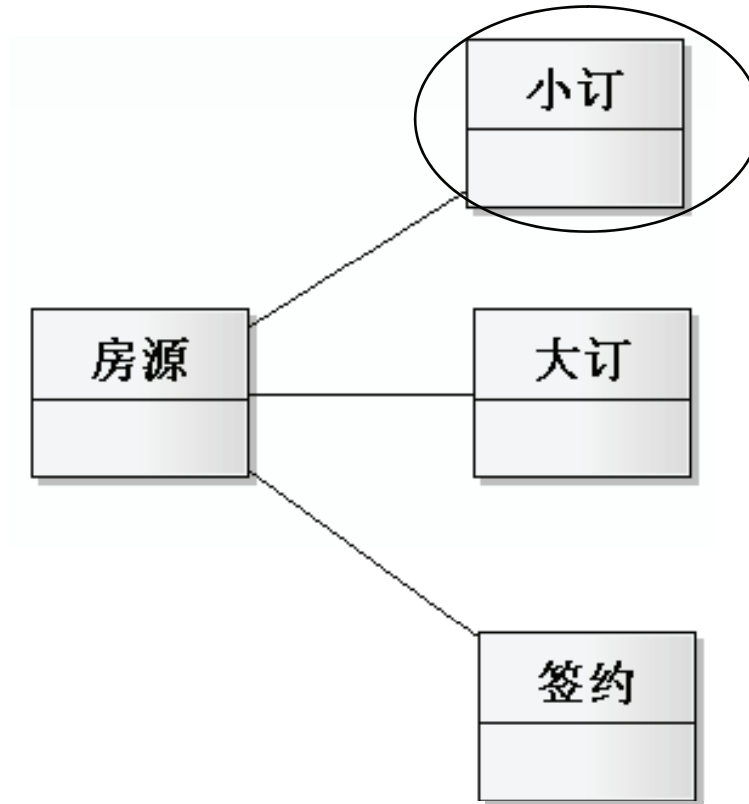


快递路线
抽取区号、格式化

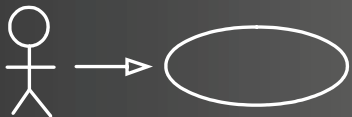
类还是属性？看封装知识



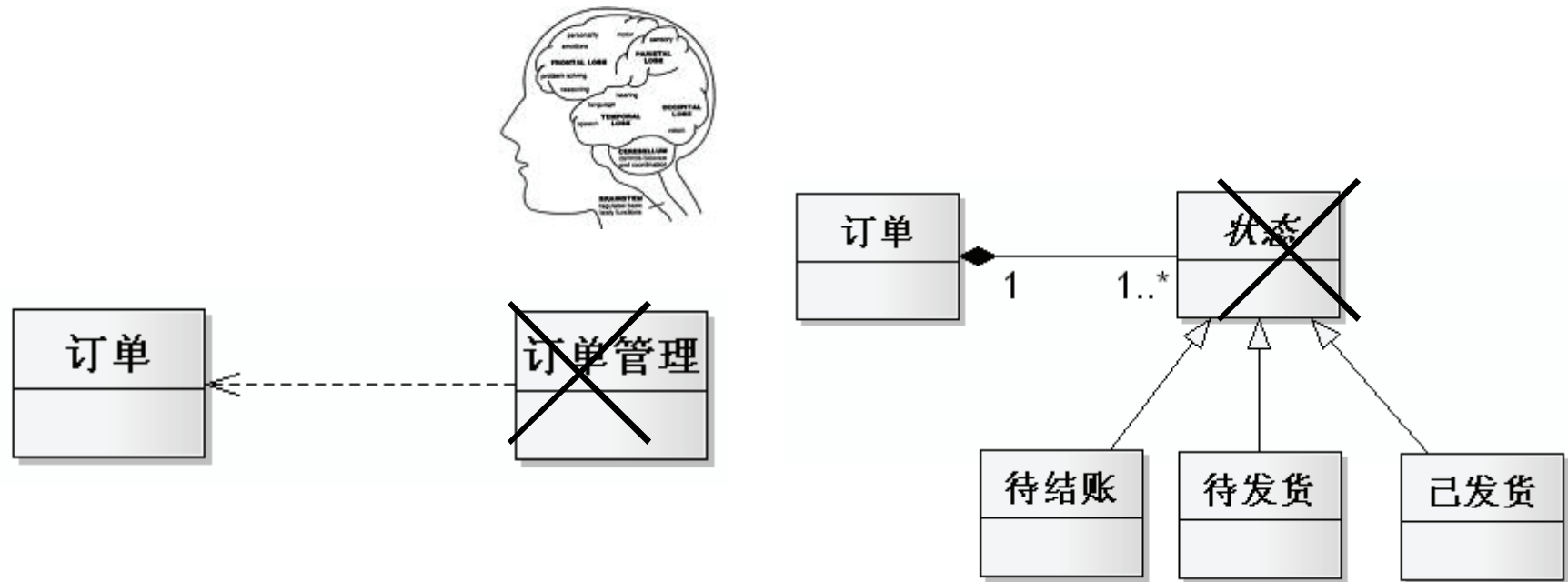
识别类和属性



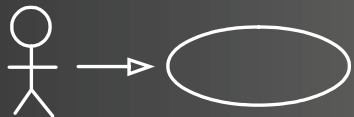
非有形概念也是类



识别类和属性

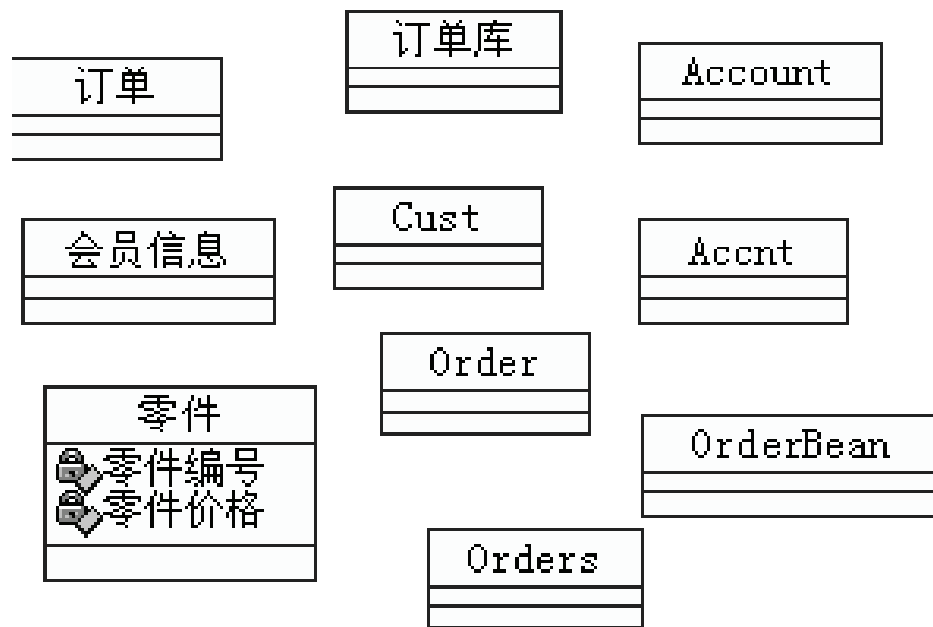


不同抽象级别



识别类和属性

——类和属性的名字



命名规则:

名词、形容词—名词

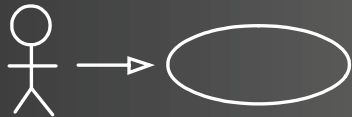
使用业务词汇

没有“与”、“或”

小心：“表”、“信息”、“数据”...

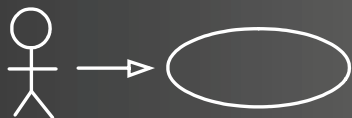
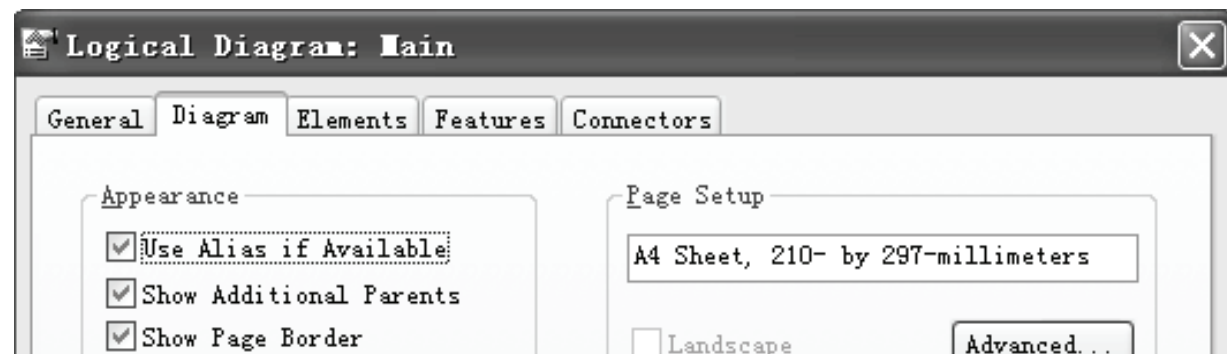
属性名不要类前缀

英文：不用缩写、单数

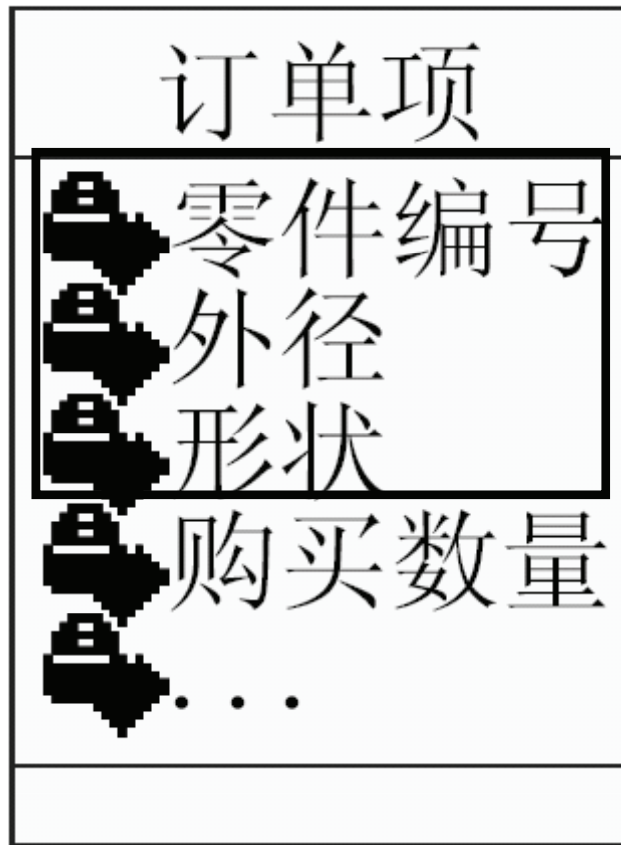


识别类和属性

——类图的位置

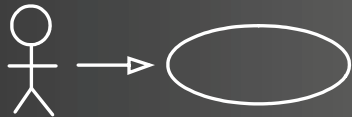


识别类和属性——审查

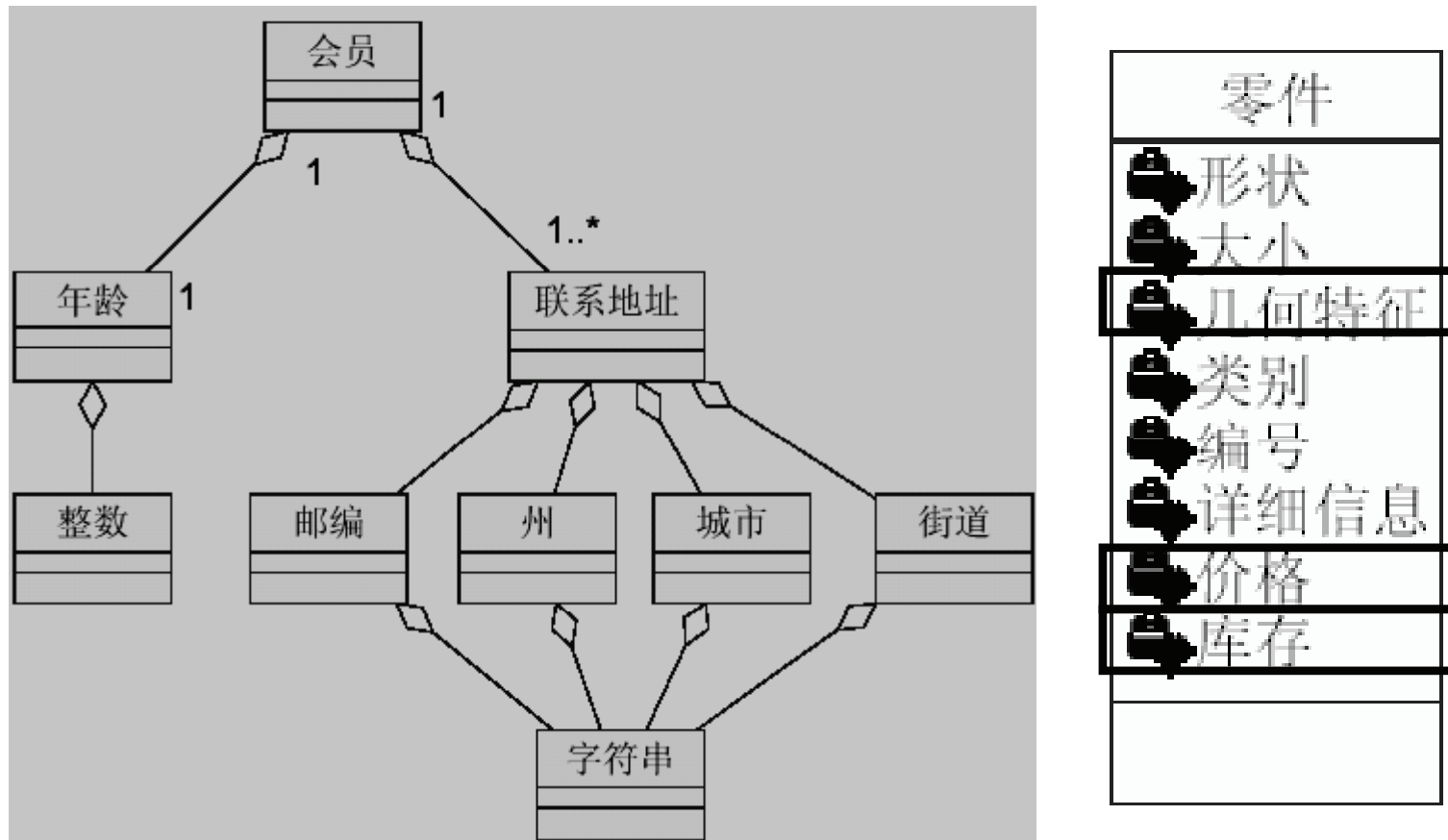


什么的什么
什么的什么的什么
什么的什么的什么的什么

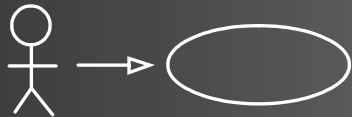
属性是否直接描述类的特征



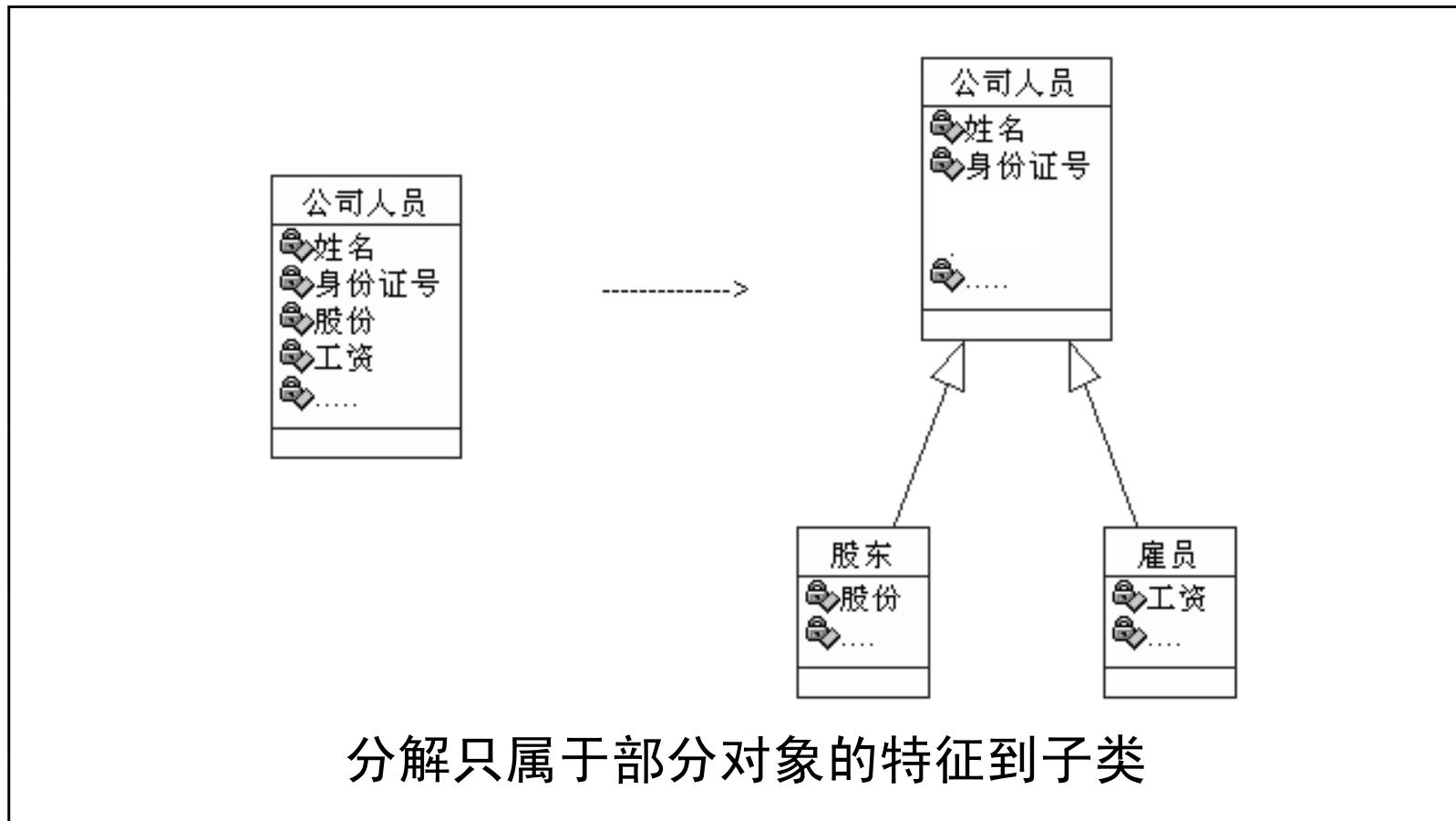
识别类和属性——审查



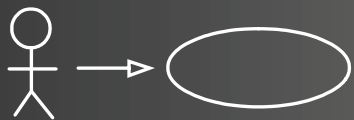
是否有复杂结构或1对多的属性



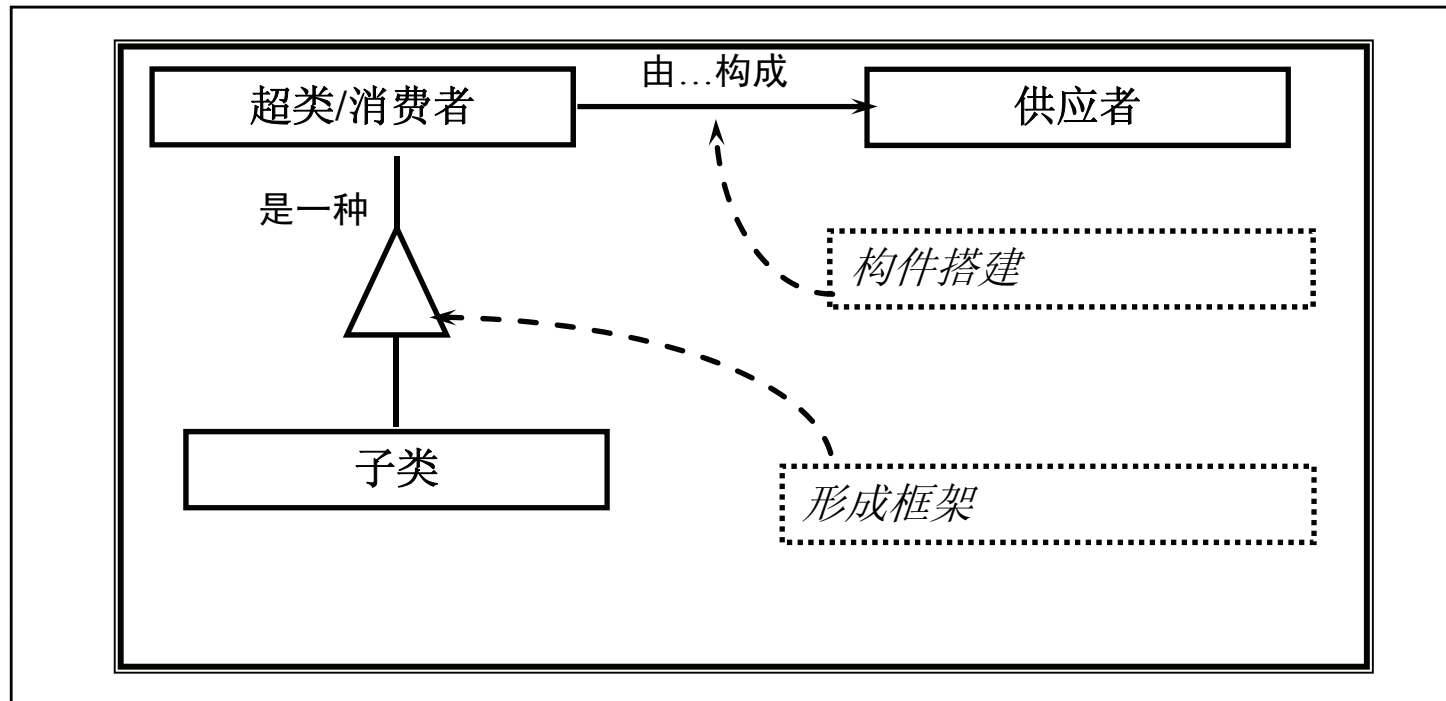
识别类和属性——审查



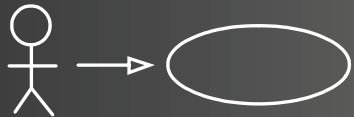
属性是否对类的所有对象都有意义



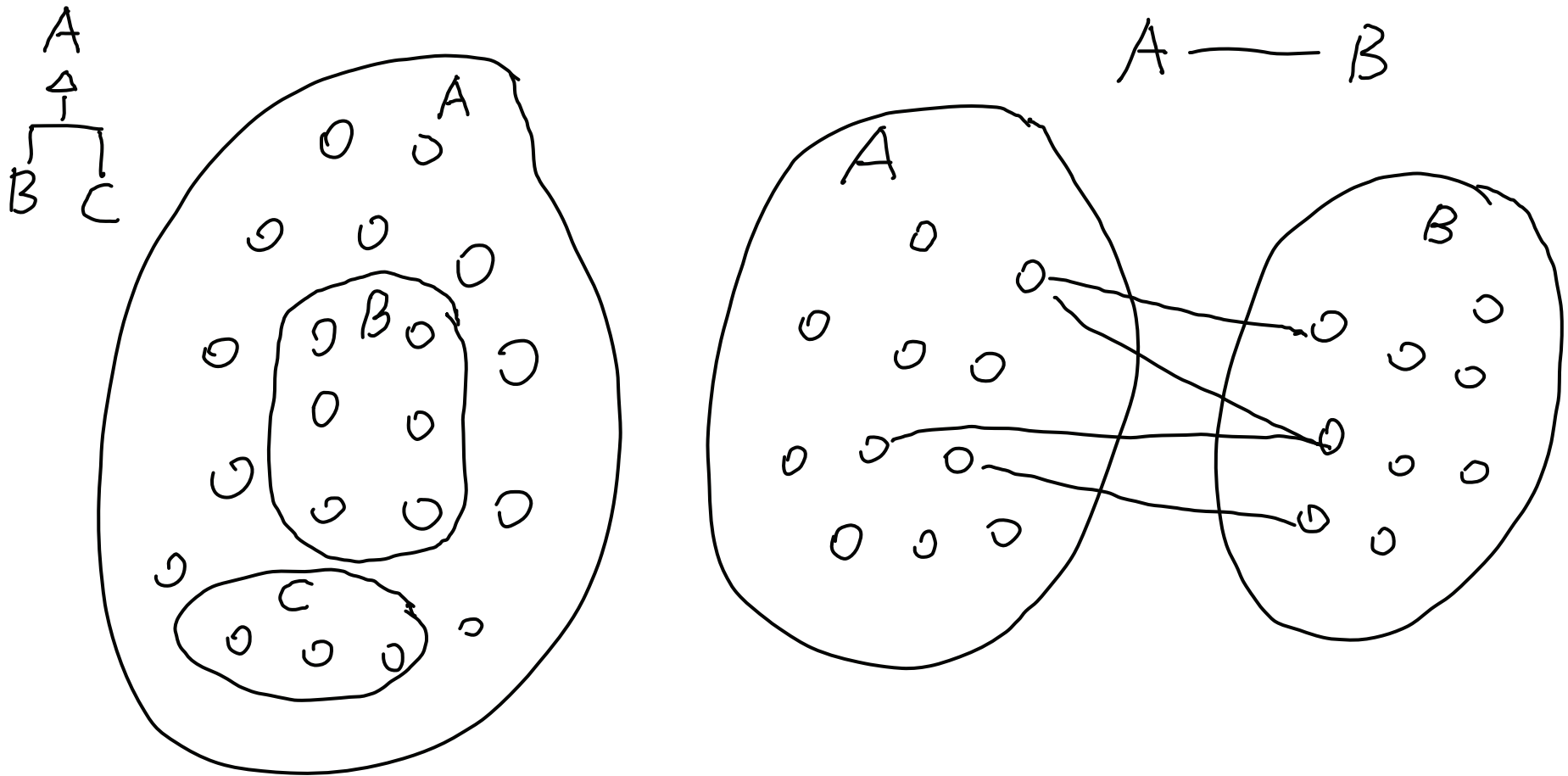
类的关系



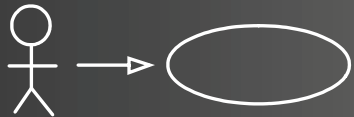
- 泛化：子类通过继承复用超类的特征（集合关系）
- 关联：对象通过组装复用其他对象的特征（个体关系）



类的关系



泛化和关联

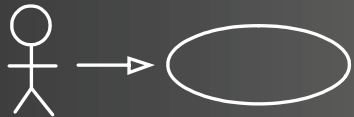
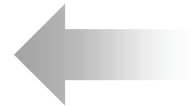


步骤

 识别类和属性

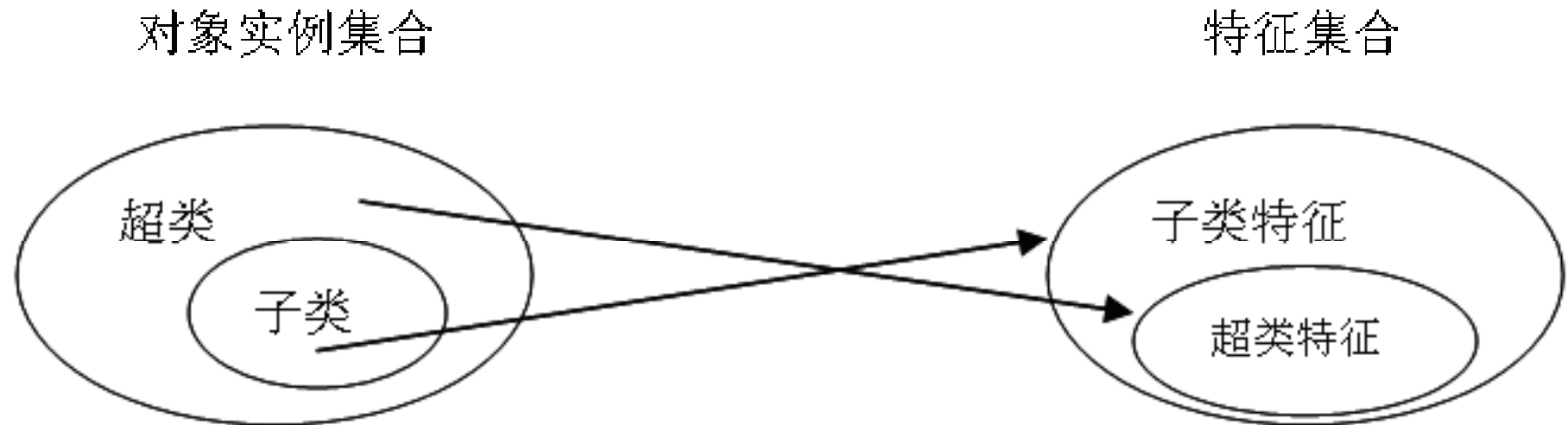
 识别泛化

 识别关联

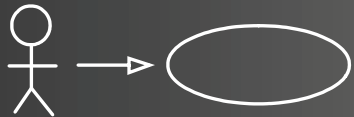


识别泛化

——泛化

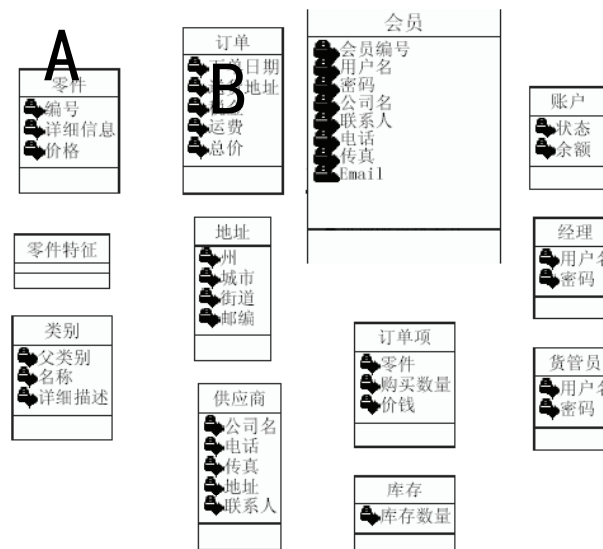


超类的对象集合 包含 子类的对象集合

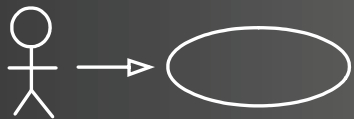


识别泛化

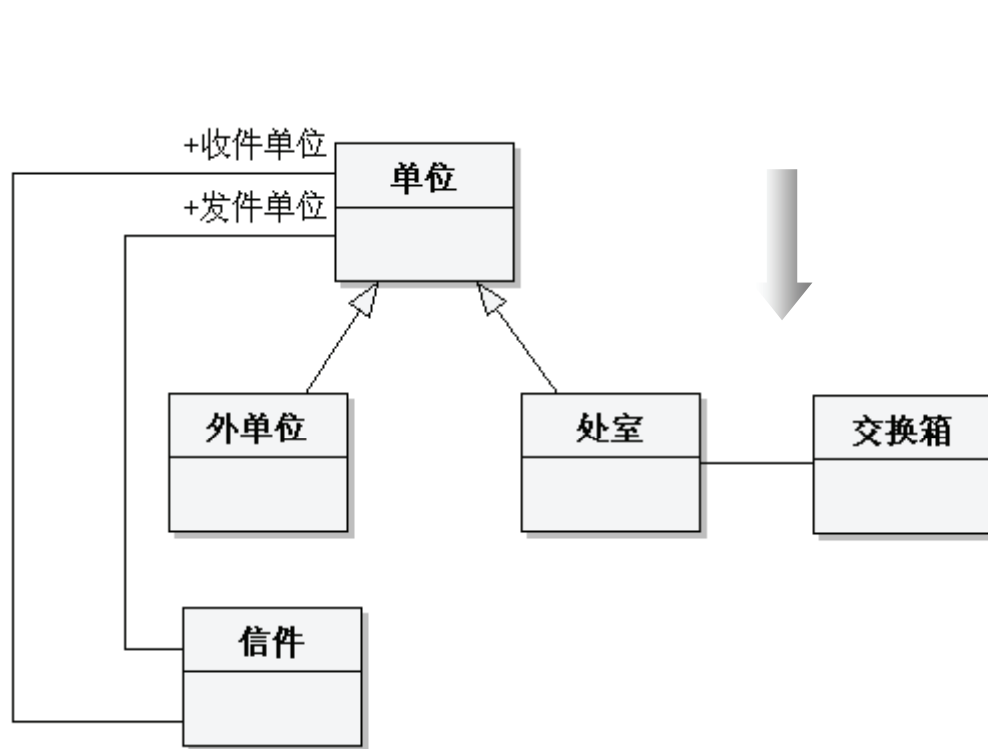
——思路



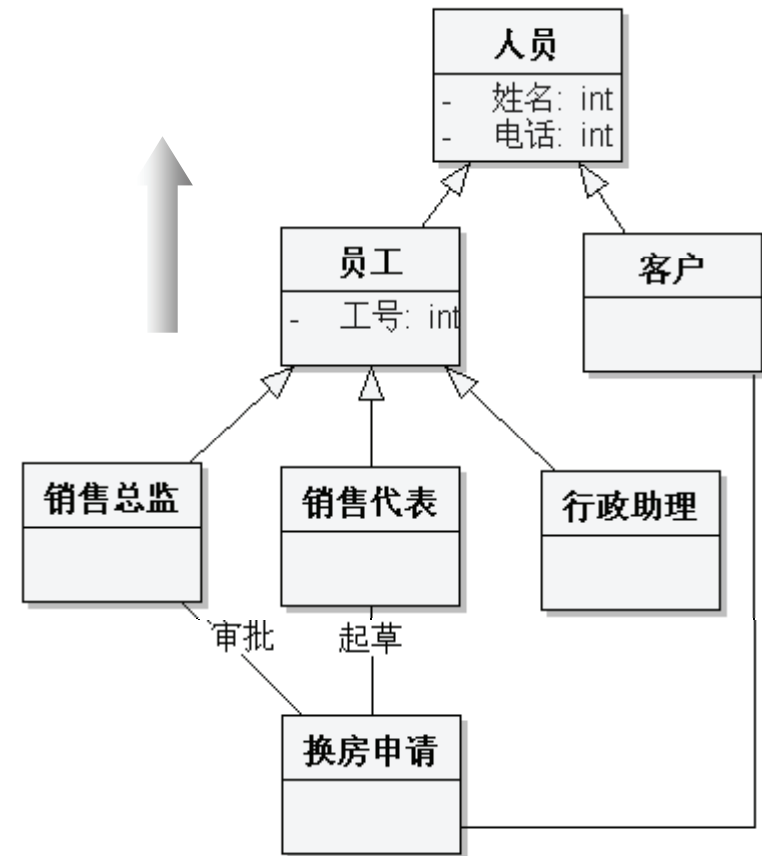
- A的对象总是B的对象，B的对象也总是A的对象
- A的对象总是B的对象，B的对象有时是A的对象 (*)
- A的对象从来不是B的对象，B的对象从来不是A的对象
- A和B之间有一些共同的特征 (*)



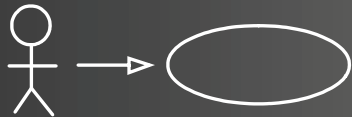
识别泛化



自上而下

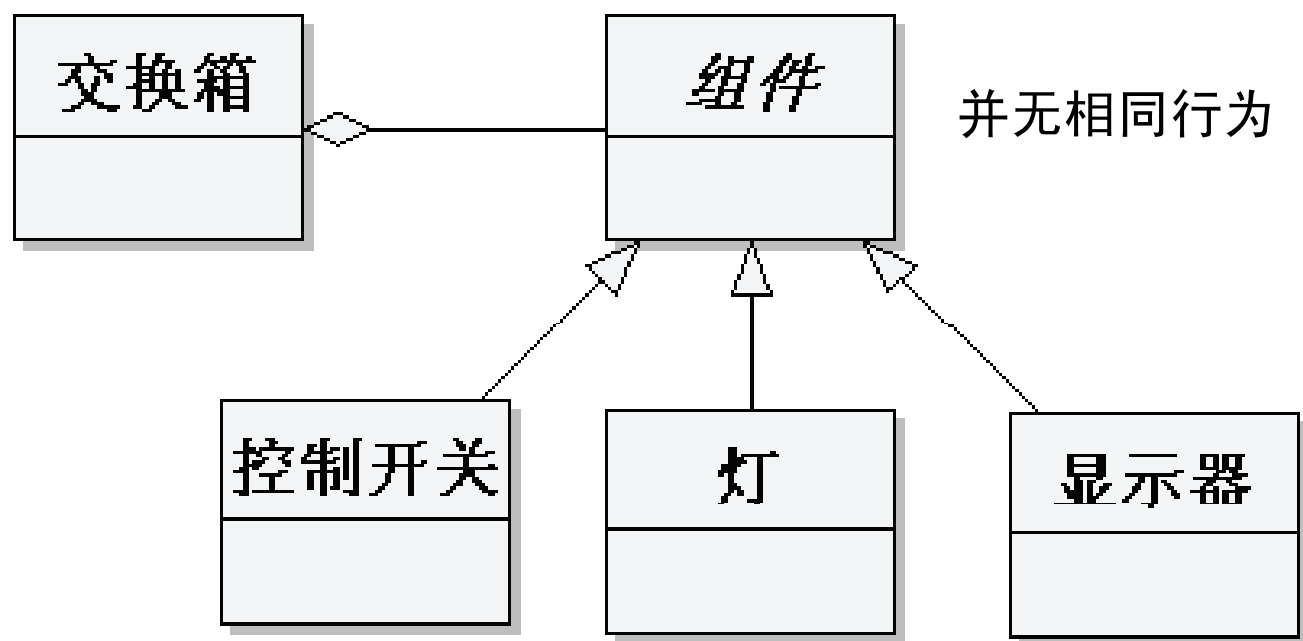


自下而上

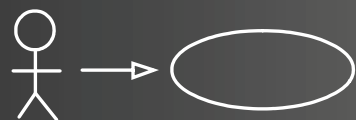




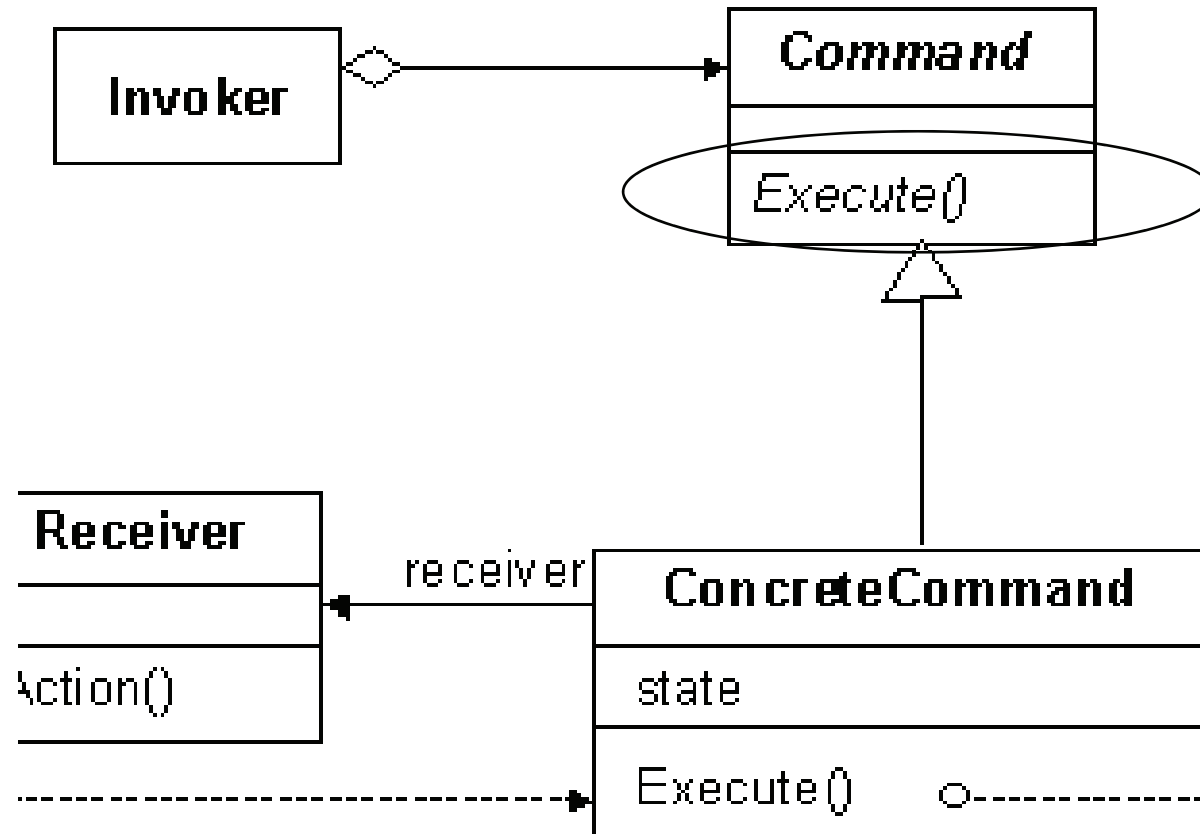
识别泛化



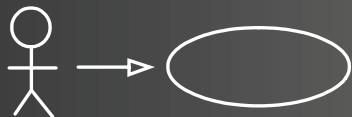
不要为了抽象而抽象



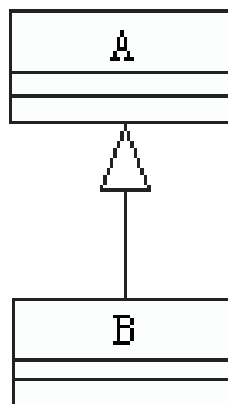
识别泛化



对比



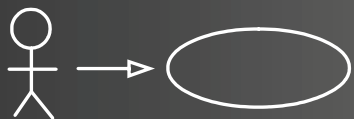
识别泛化



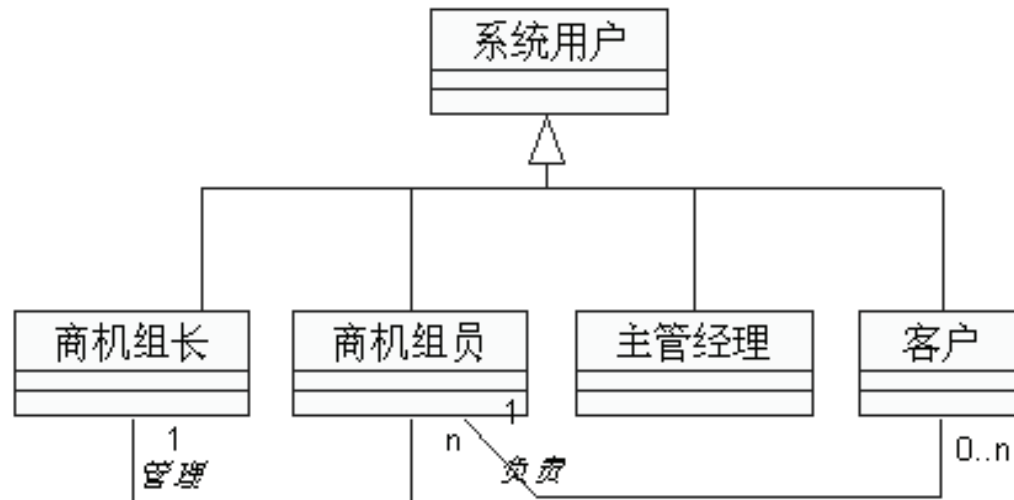
B是一种A吗→是

A是B的一部分吗→不是

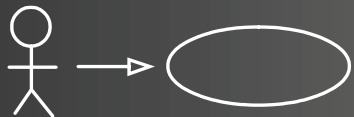
泛化还是关联？



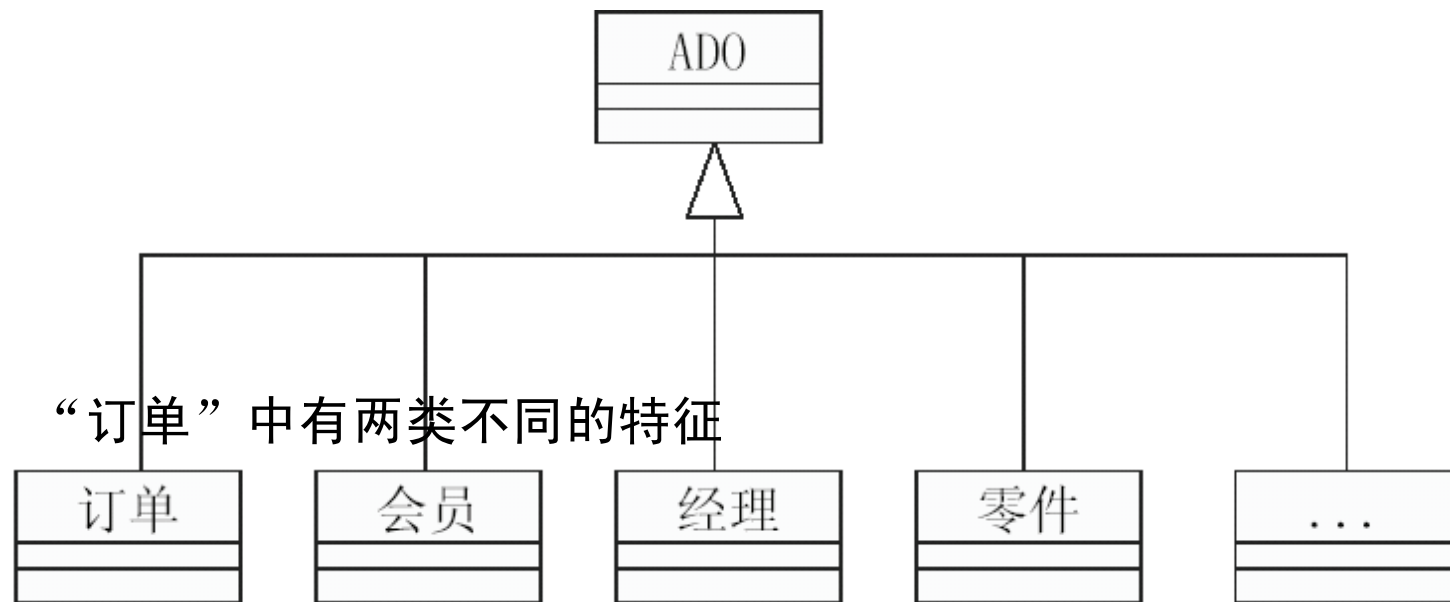
识别泛化



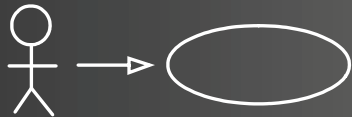
泛化的子类对象之间可能会互有关联



识别泛化



不同领域的类之间不应形成泛化关系

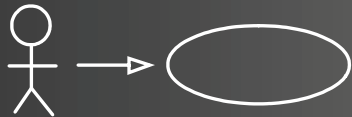




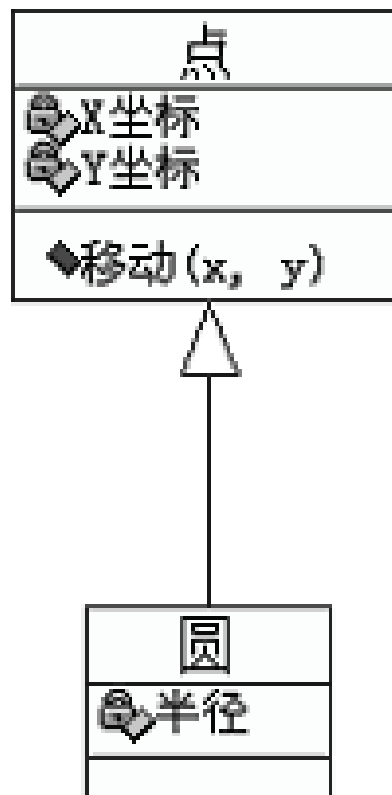
识别泛化

- Liskov Substitution Principle
- 子类型应该能替换基类型
- -----Barbara Liskov

Liskov替换原则——LSP

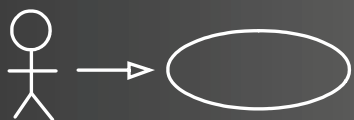


识别泛化

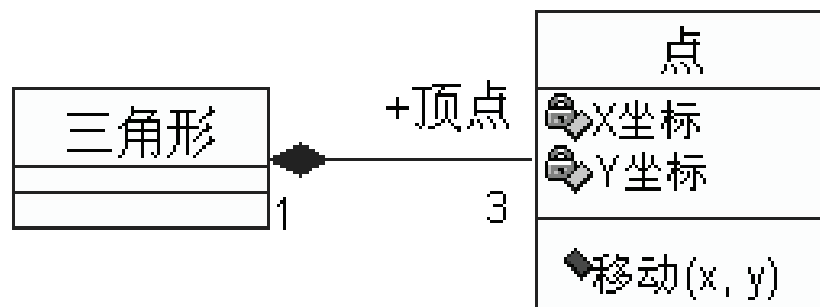


圆是一种有半径的点

这样合适吗

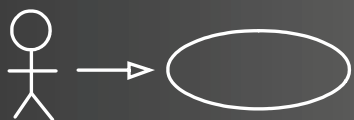


识别泛化

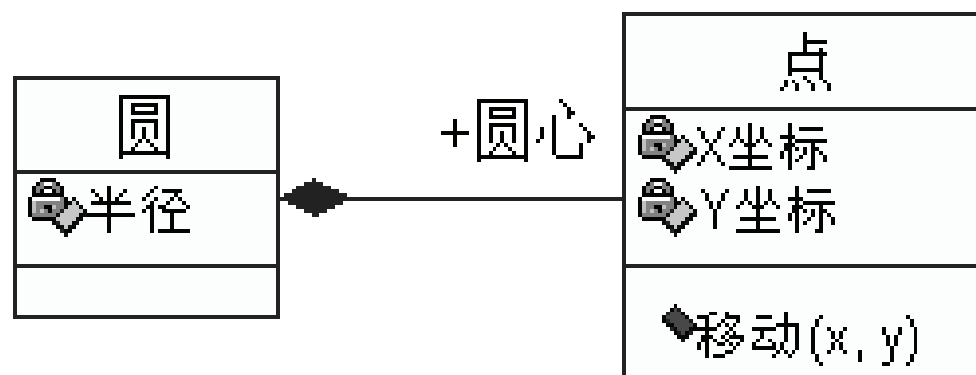


三个圆能组成三角形吗？

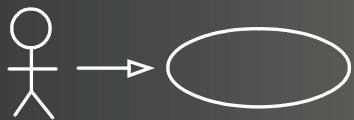
LSP——子类应能替换超类



识别泛化



关联更能反映业务内涵

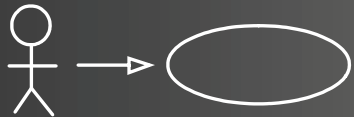


步骤

 识别类和属性

 识别泛化

 识别关联



识别关联

——关联的几种表现形式

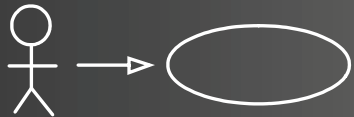
连接



聚合

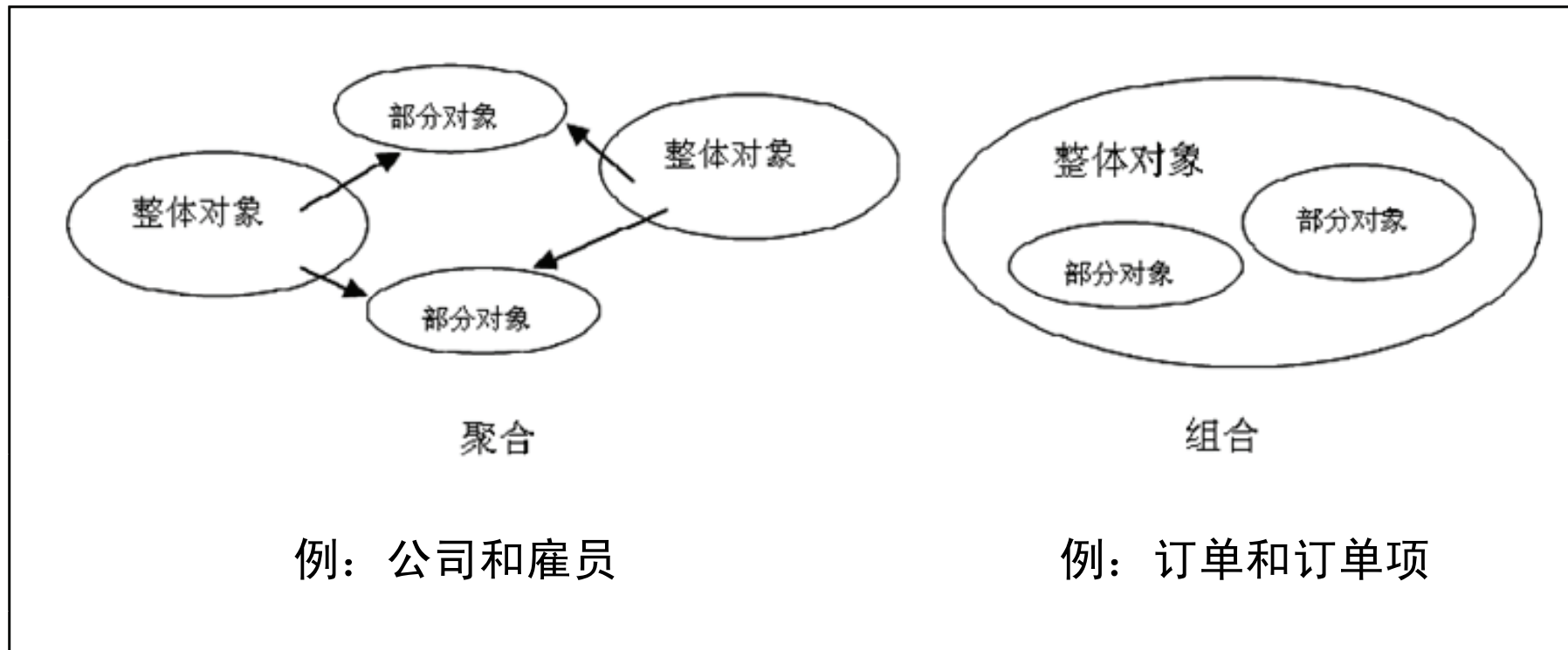


组合

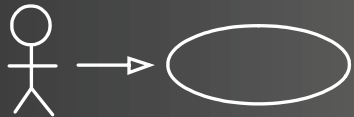


识别关联

——聚合 vs. 组合

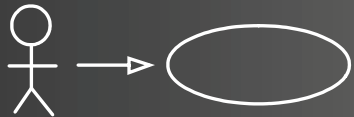
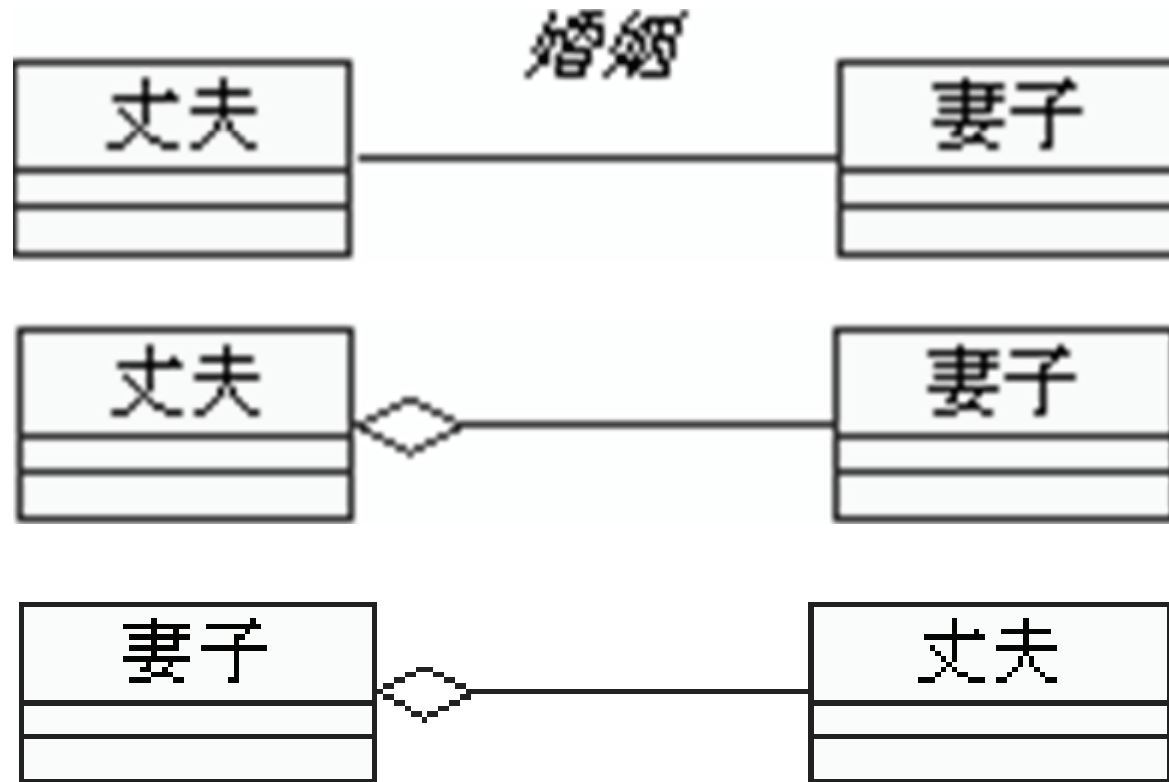


分析 workflow 不必区分



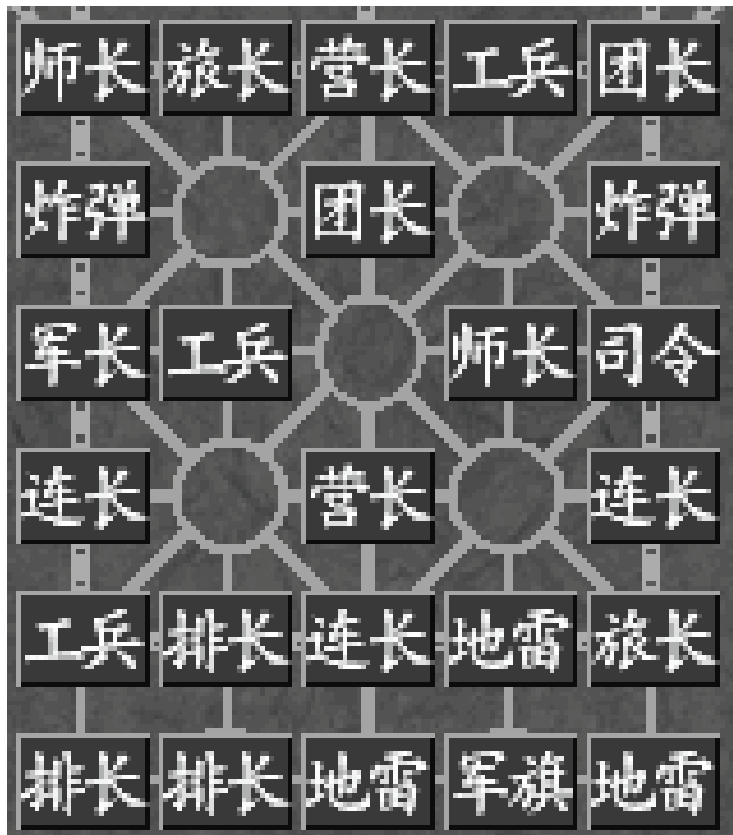
识别关联

——连接 vs. 聚合？

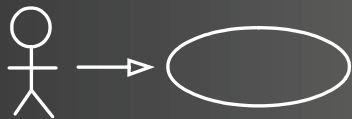


识别关联

——识别聚合（组合）结构的意义

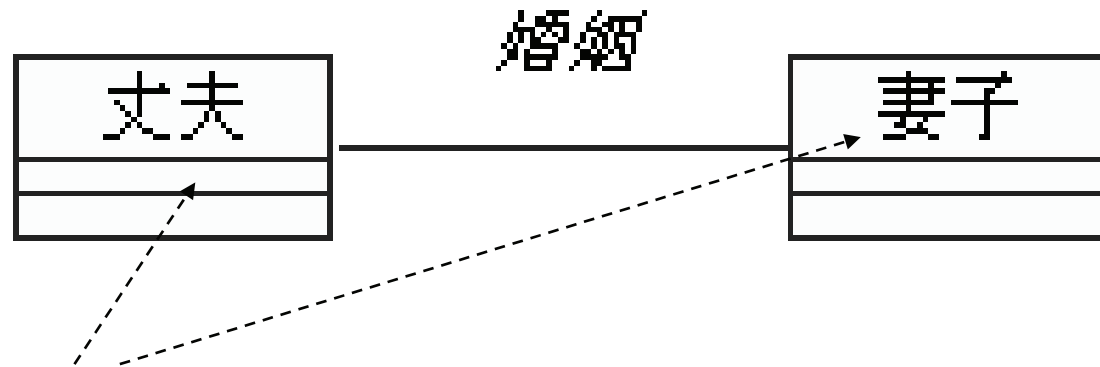


在动态建模时帮助责任分配

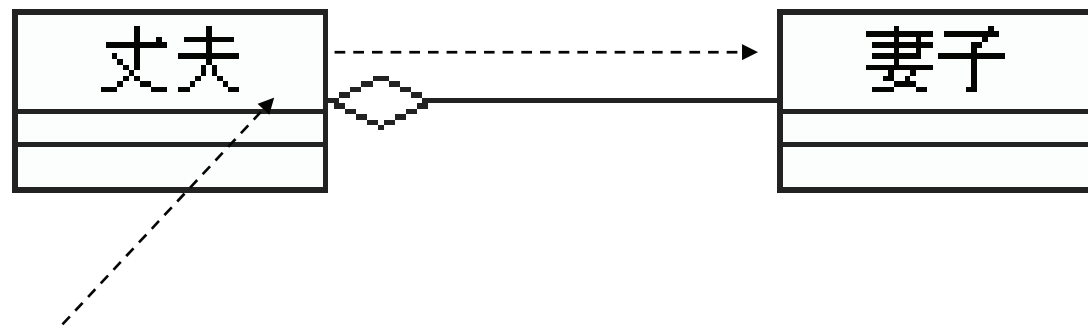


识别关联

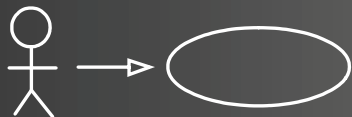
——连接？还是聚合？



男女平等



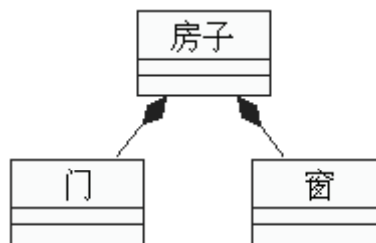
男尊女卑



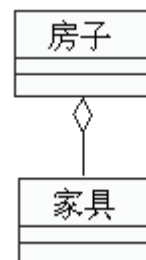
识别关联

——聚合/组合情况

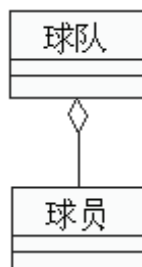
❖ 组合/部分



❖ 容器/内容



❖ 集合/成员



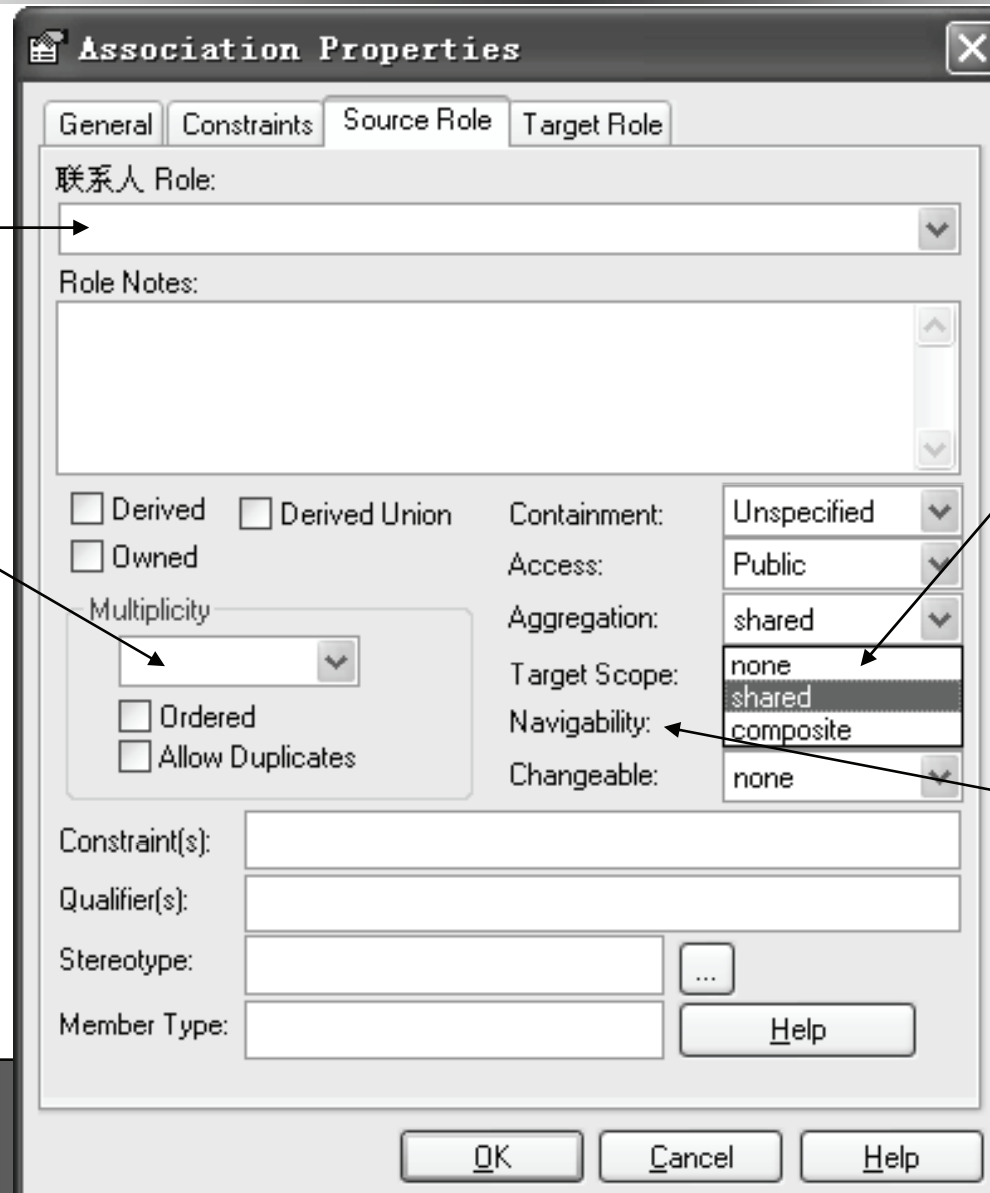
识别关联

角色

多重性

聚合组合

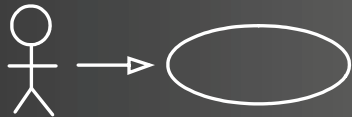
导航关系



The image shows a 'UML Association Properties' dialog box with the following sections:

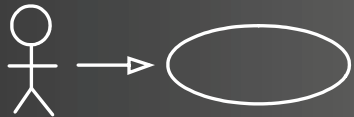
- General** (selected tab):
 - 联系人 Role: [Empty dropdown]
 - Role Notes: [Empty text area]
 - ☐ Derived ☐ Derived Union ☐ Owned
 - Multiplicity: [Empty dropdown]
 - ☐ Ordered ☐ Allow Duplicates
 - Containment: [Unspecified]
 - Access: [Public]
 - Aggregation: [shared]
 - Target Scope: [none, shared, composite]
 - Navigability: [Empty dropdown]
 - Changeable: [none]
 - Constraint(s): [Empty text area]
 - Qualifier(s): [Empty text area]
 - Stereotype: [Empty text area]
 - Member Type: [Empty text area]
- Source Role**
- Target Role**

Buttons at the bottom: OK, Cancel, Help.

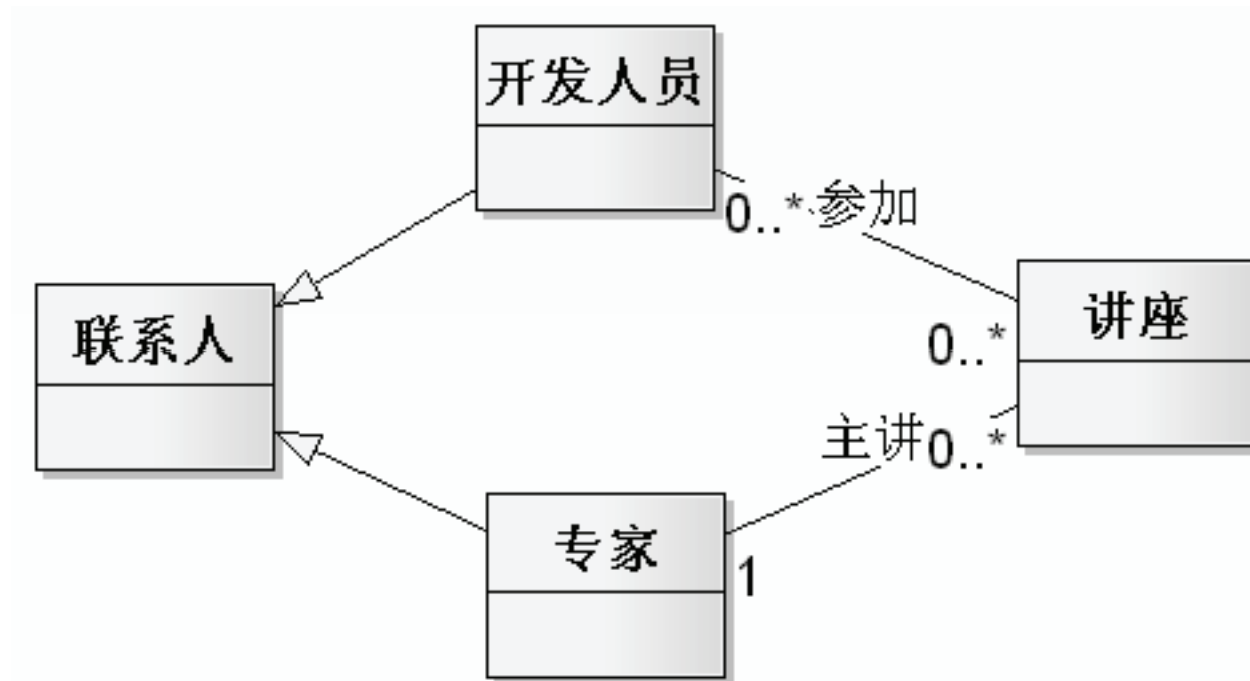


识别关联

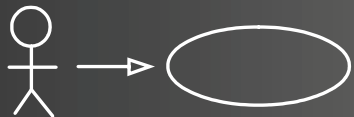
- 逐一考虑类图上各类之间关系，以及类与类自身的关系



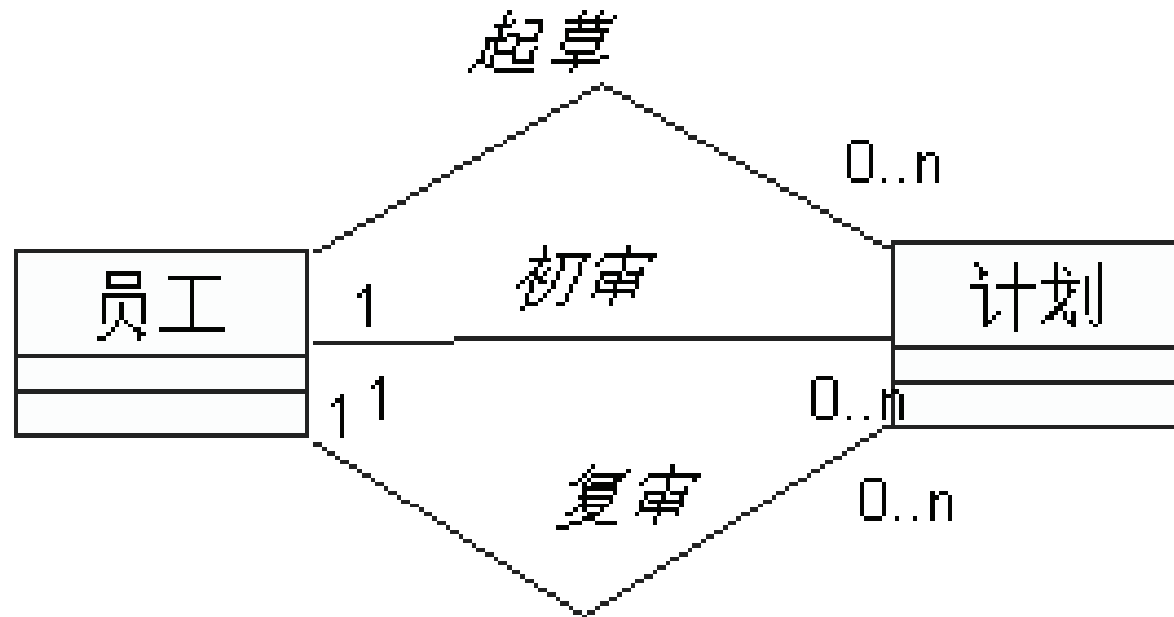
识别关联



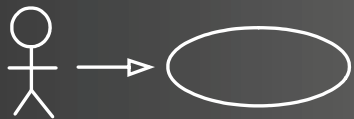
关联的名字胜过多重性



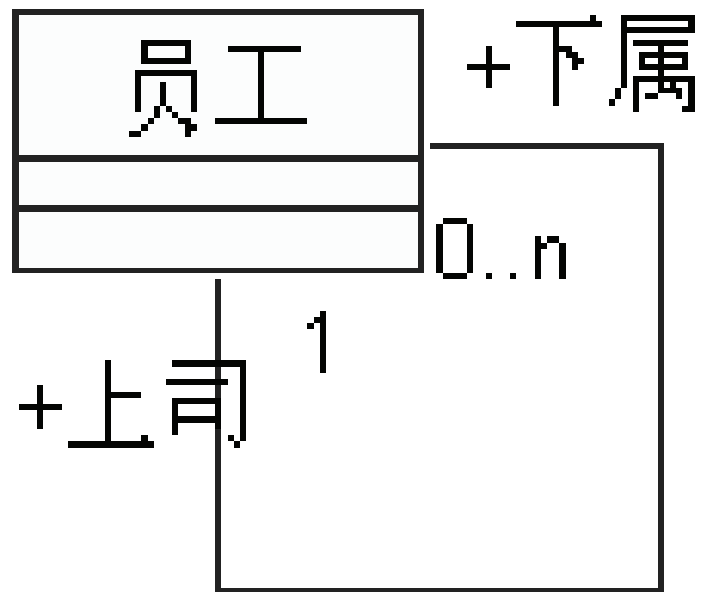
识别关联



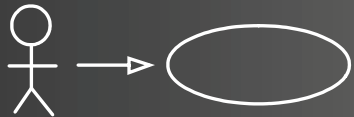
类之间关联可以有多种



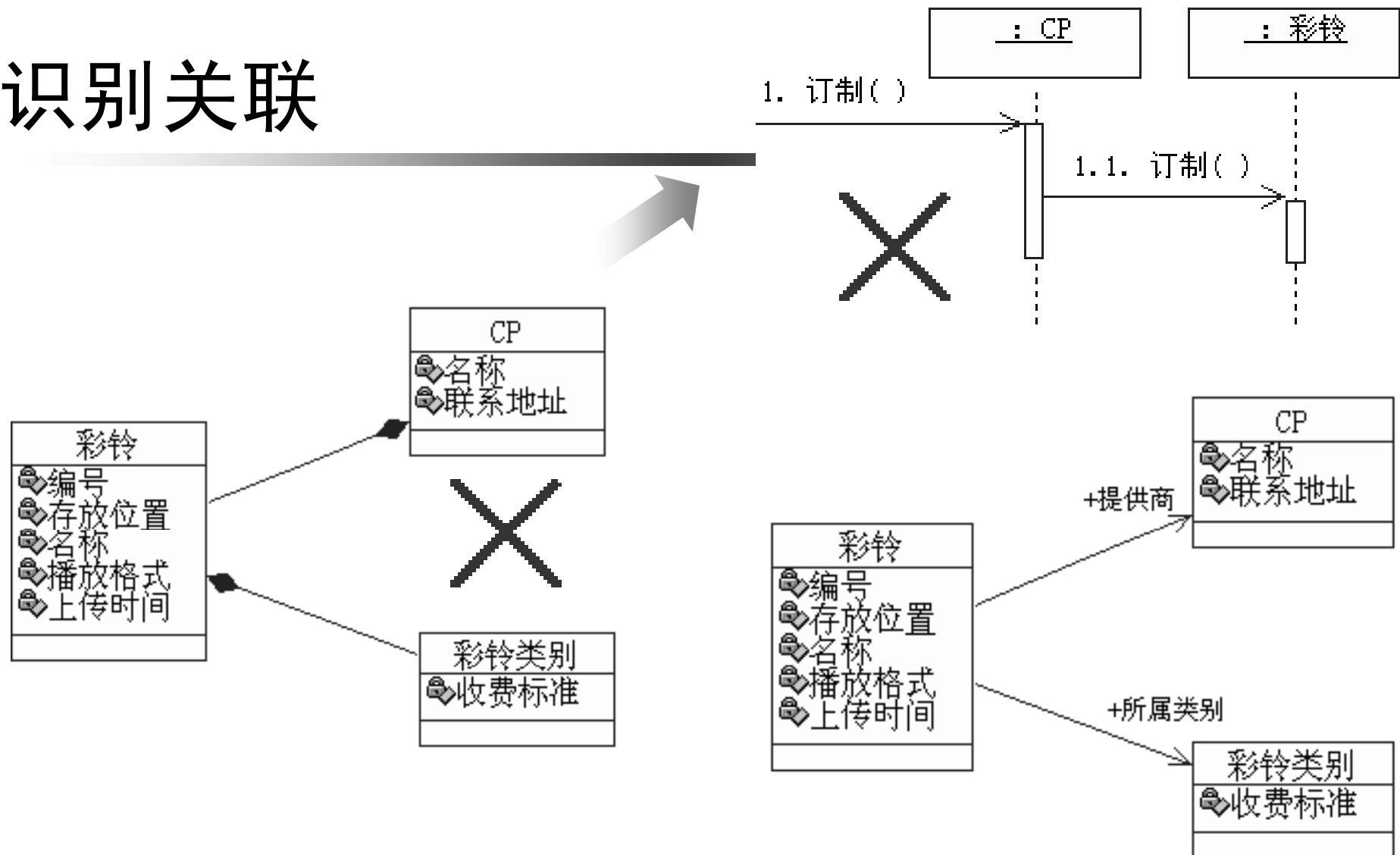
识别关联



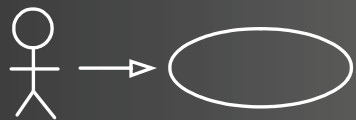
自反关联



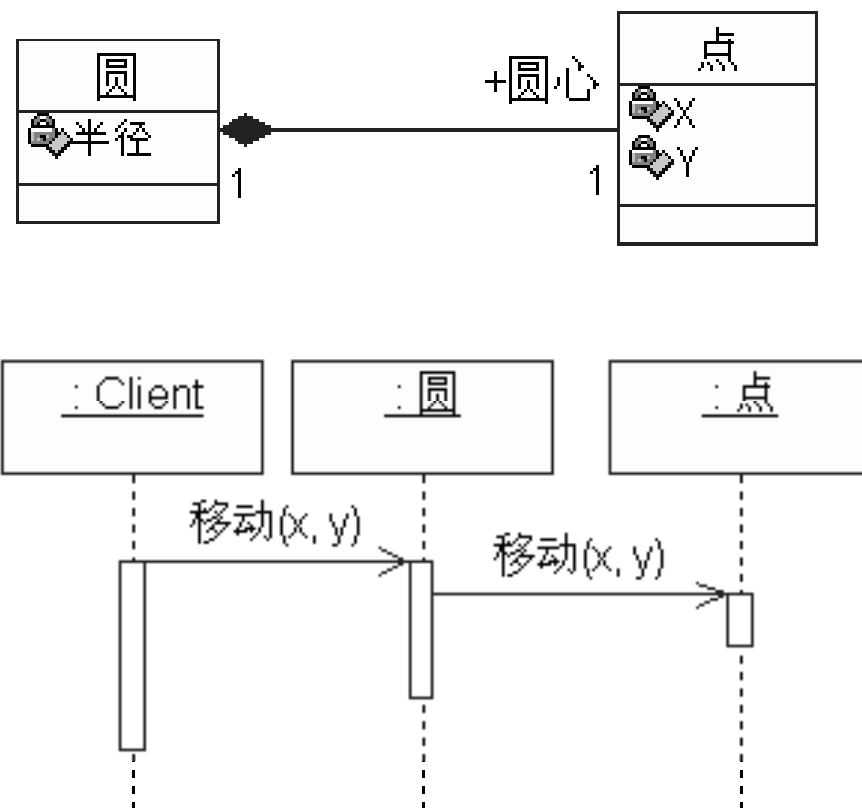
识别关联



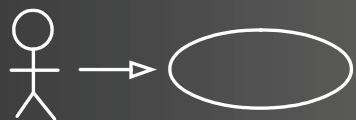
聚合组合不是万金油



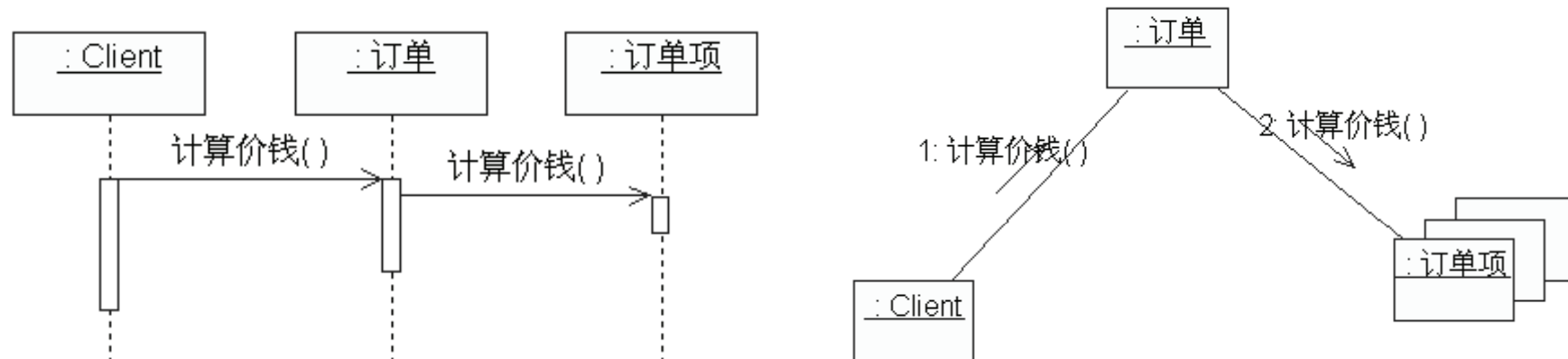
识别关联



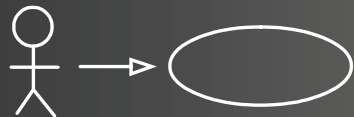
聚合/组合意味着传递性



识别关联

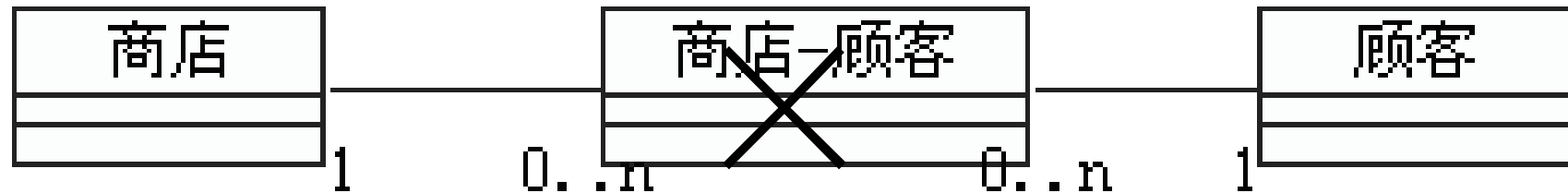


聚合/组合意味着传递性

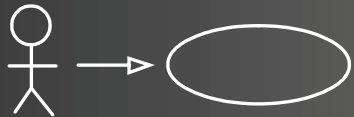


识别关联

——警惕“数据库”习惯

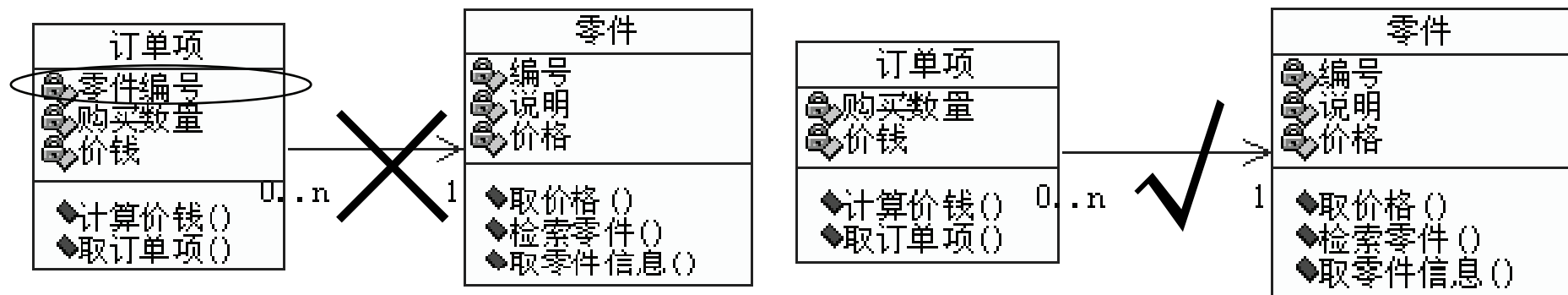


关联类也要有业务意义



识别关联

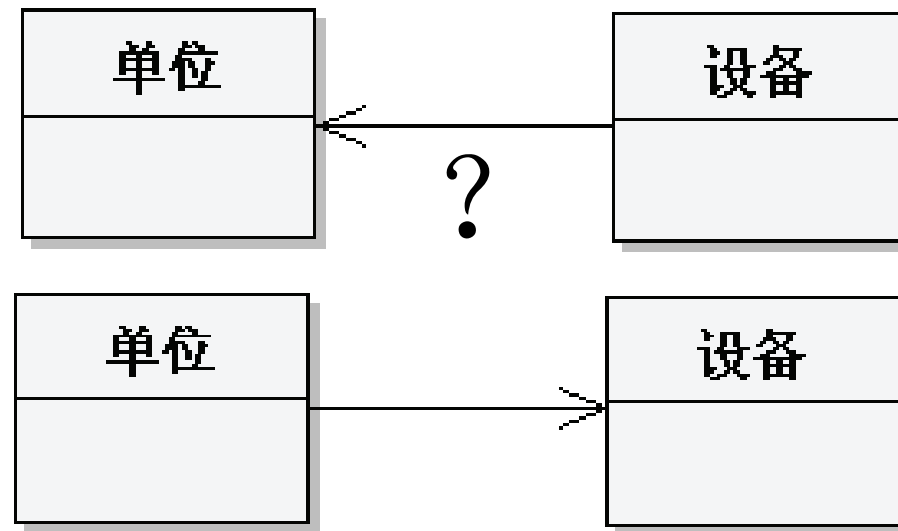
——警惕“数据库”习惯



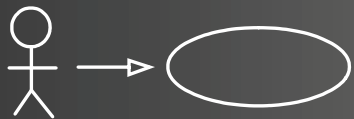
不是拥有“外键”，而是拥有“对象”



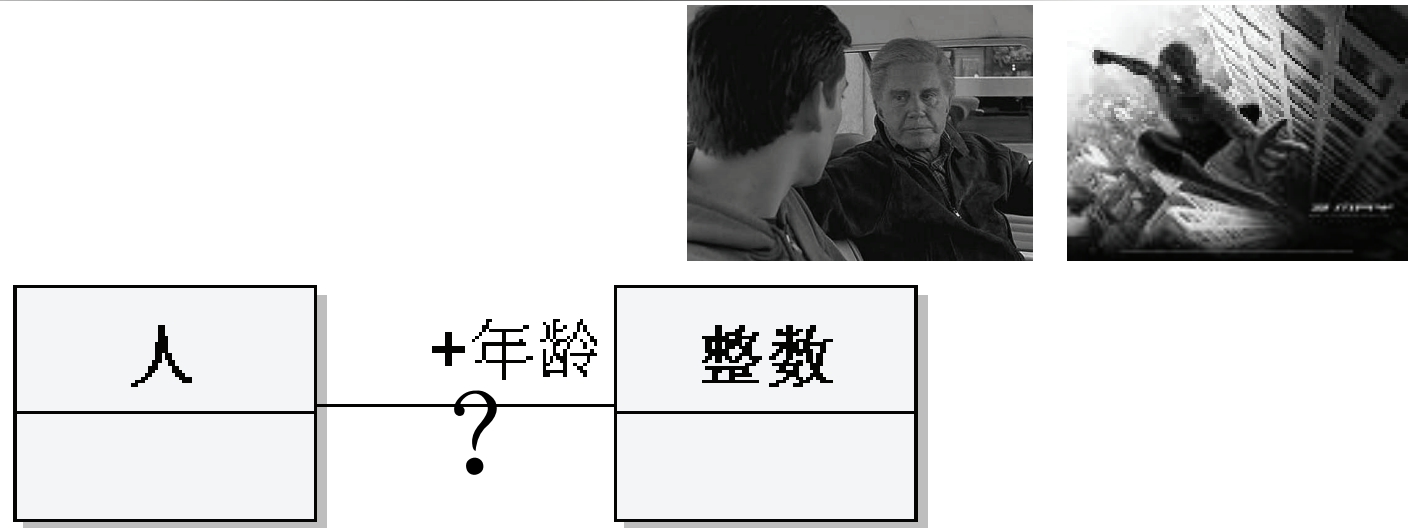
识别关联



关联的导航方向



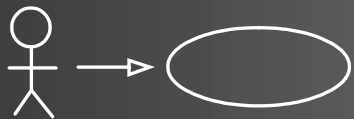
识别关联



能力越大，责任越大

With great power comes great responsibility

关联的导航方向一极端的例子

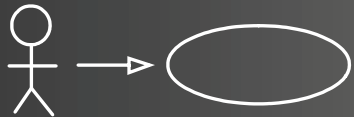


识别关联

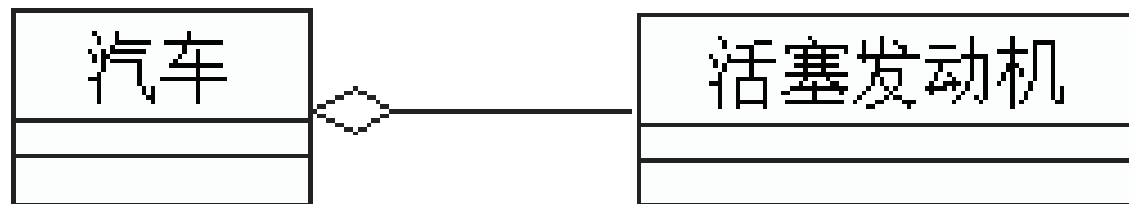
× × 锅检所，了解哪些设备过期未检验

设备的状态比单位的状态丰富

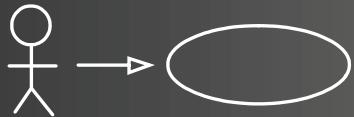
关联的导航方向—结合上下文



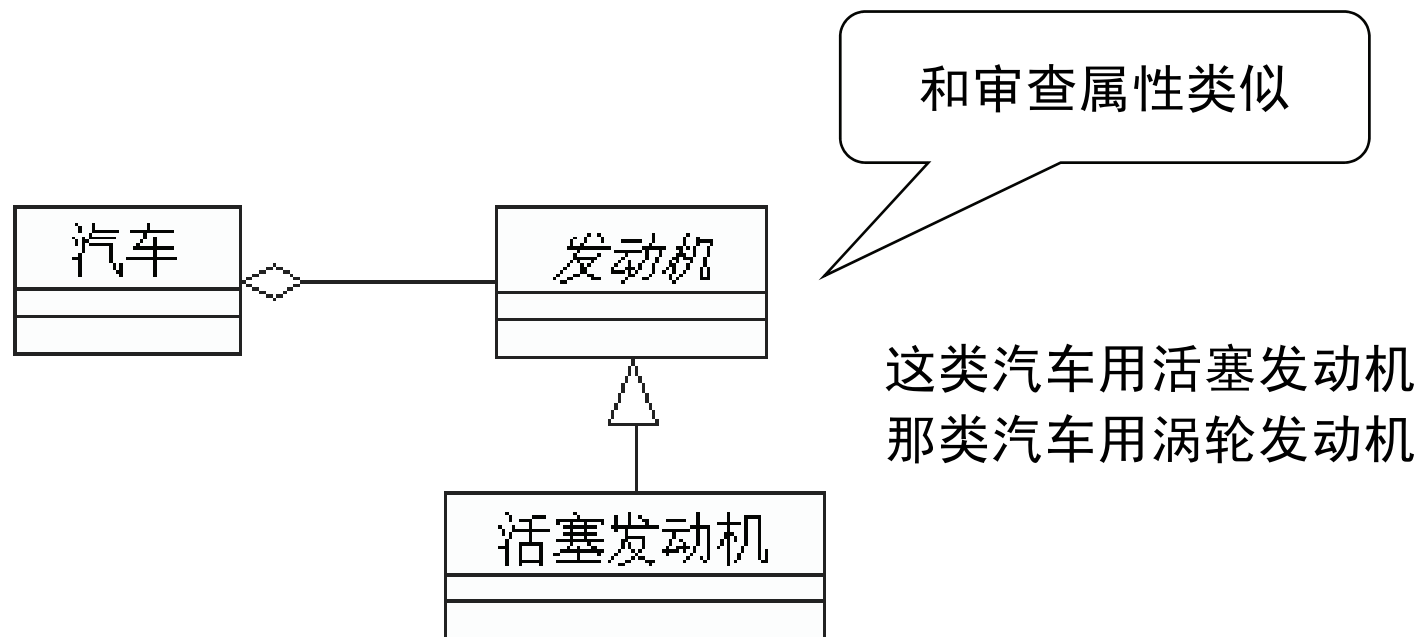
识别关联



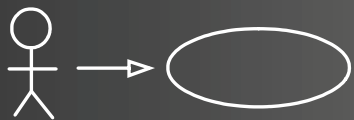
关联两端抽象级别要合适



识别关联

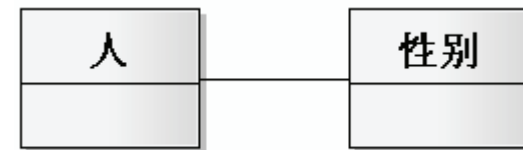


调整

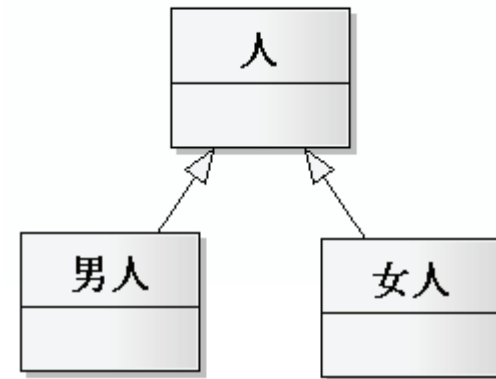


识别关联

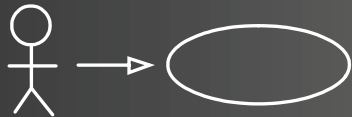
- 共享数据——关联优先



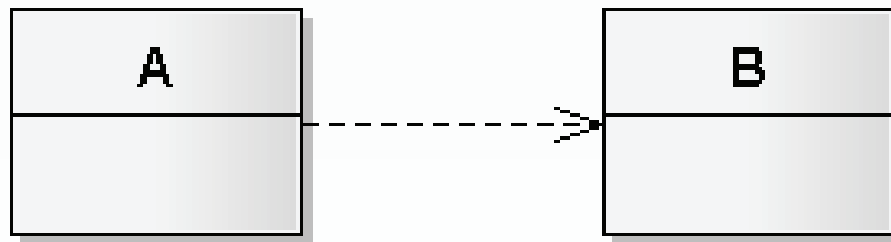
- 行为变异——泛化优先



泛化或关联的选择



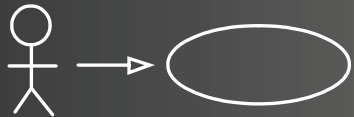
类的依赖



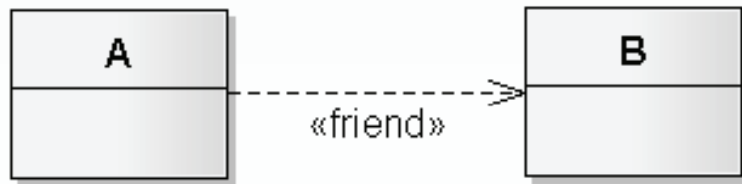
一个大杂烩
B变化A也要变化
但又不是泛化和关联

（不主张在类图上画依赖关系）

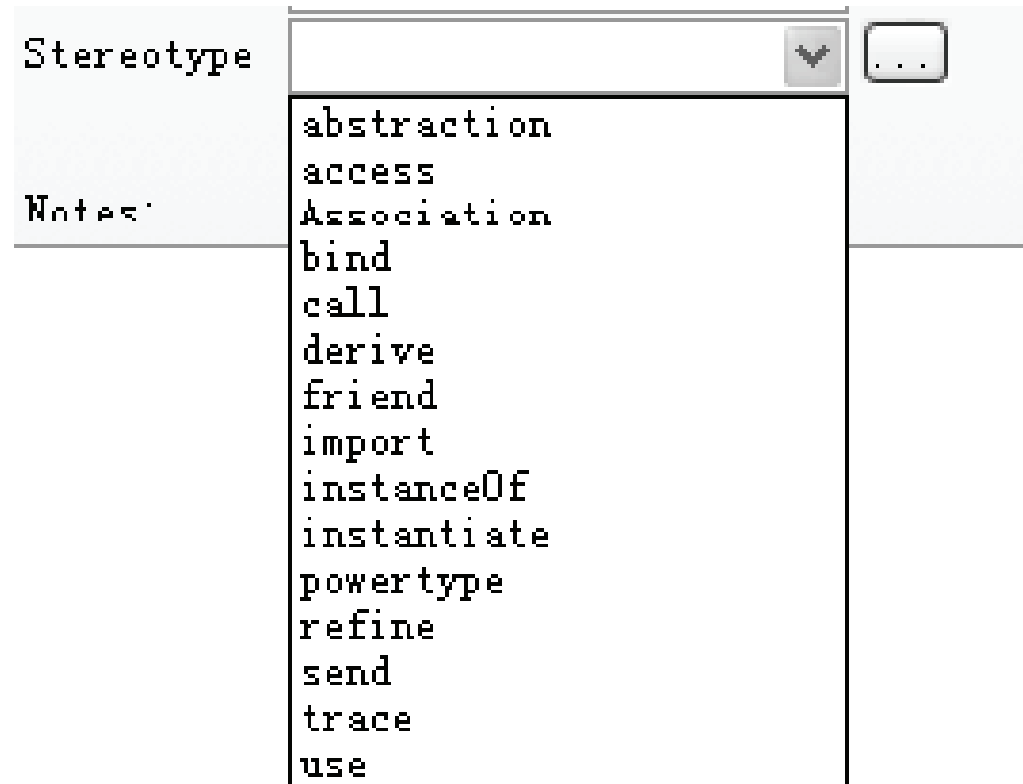
依赖



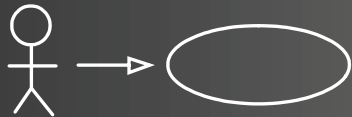
类的依赖



依赖的构造型随着语言不同，
列表还会增加



依赖



类的依赖

