

如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节

四、如何进行问题域部分的设计？

1、继续运用OOA的方法
——概念、表示法及策略

2、使用OOA结果，并加以修改
——需求的变化，新发现的错误

3、使用OOA结果，并进行补充与调整（本节的重点）

- (1) 为复用设计与编程的类而增加结构
- (2) 增加一般类以建立共同协议
- (3) 按编程语言调整继承
- (4) 提高性能
- (5) 为实现对象永久存储所做的修改
- (6) 为编程方便增加底层细节



如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节



(1) 为复用设计与编程的类而增加结构

----OOA识别和定义的类是本次开发中新定义的，需要进行编程。

----如果已存在一些可复用的类，而且这些类既有分析、设计时的定义，又有源程序，那么，复用这些类即可提高开发效率与质量。

----可复用的类可能只是与OOA模型中的类相似，而不是完全相同，因此需对二者进行修改。

目标：尽可能使复用成分增多，新开发的成分减少



如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节



不同程度的复用

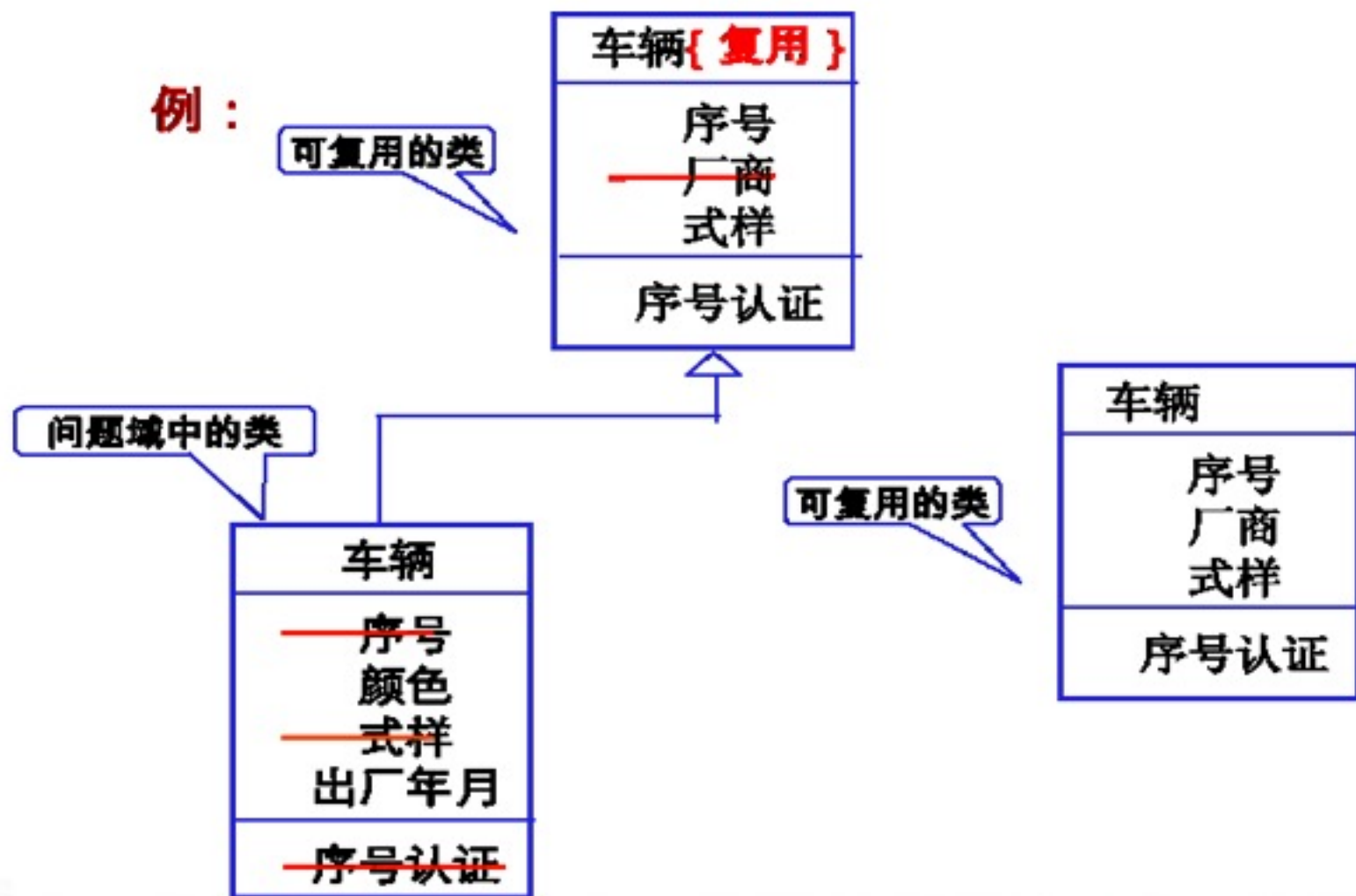
定义的 可复用类 的信息	比	当前所需 的类的 信息	=	直接复用
			<	通过继承复用
			>	删除可复用类的多余信息
			≈	删除多余信息，通过继承而复用

对第四种情况的做法：

- (1) 把要复用的类加到问题域。
- (2) 在类名后加{复用}，划掉不用的属性与操作。
- (3) 建立从复用类到问题域原有的类之间的泛化关系
- (4) 由于问题域类继承了复用类的特征，所以有些属性和操作不需要了——划掉。
- (5) 修改问题域原有类的结构和关联，必要时移到复用类。

如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节





如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节

(2) 增加一般类以建立共同协议

- 增加一般类：将所有的类组织在一起
提供全系统通用的协议
例：提供创建、删除、复制等操作
- 增加一般类：提供局部通用的协议
例：提供永久存储及恢复功能

注意：

- 1、如果新增加的类是自己定义的，则表示法和其它类一样；
- 2、如果新增加的类是编程语言提供的预定义类，则只在类符号的名字栏填写一个和语言提供的类完全相同的类名，并标上{复用}的字样，属性和操作栏不必填写任何属性和操作；
- 3、在类描述模版的“类整体说明”部分的“其它”项中用文字加以说明，比如“利用编程语言提供的同名类”。

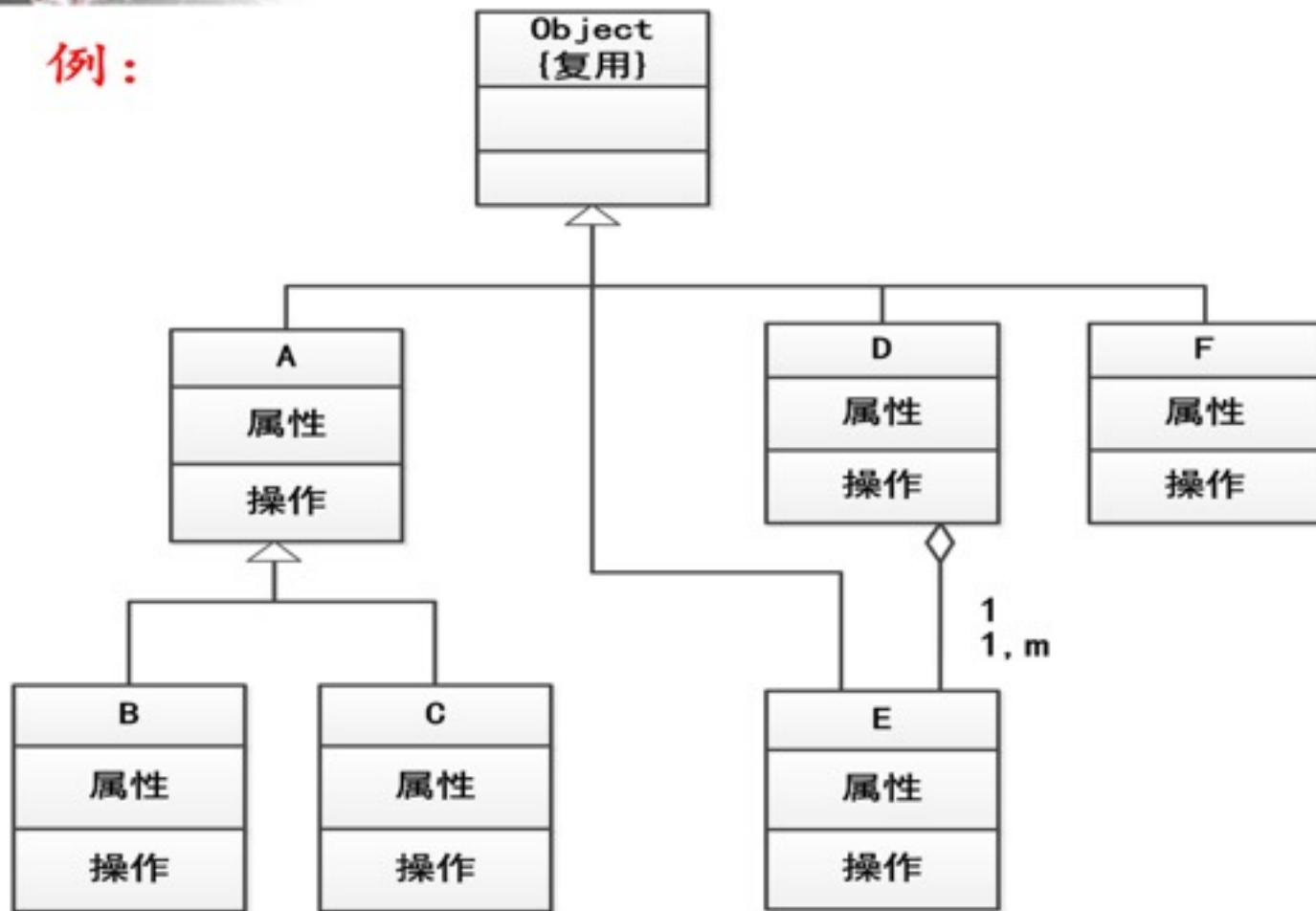


如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节



例：



上图表示“Object”是模型中新增的一般类，其它类都是模型中原有的类。这样的表示法表明，“object”是由语言提供的可复用类，实现时不需对它进行任何编程就可让A、D、E、F等类直接引用它作为一般类（B和C则通过A间接地继承。）



北京大学

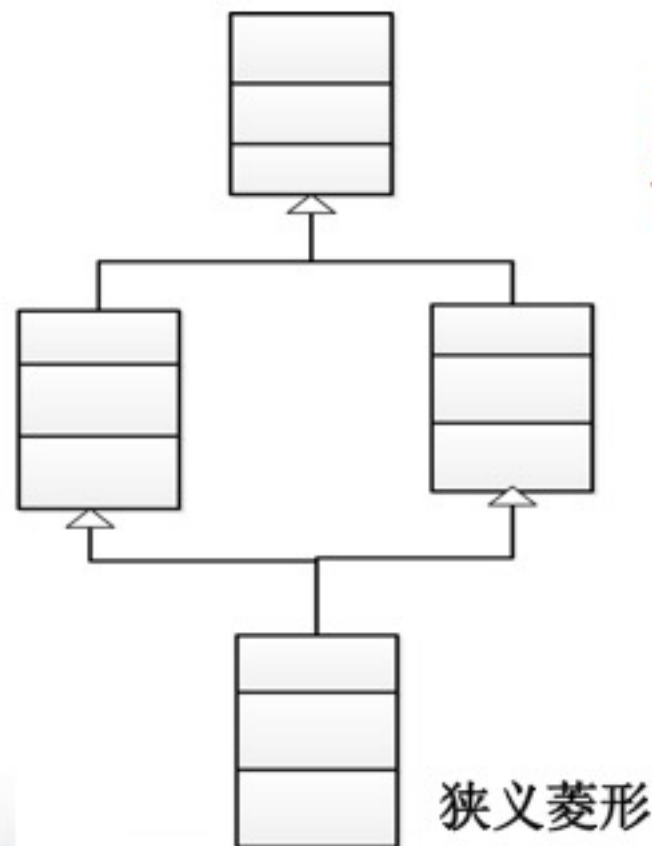
如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节

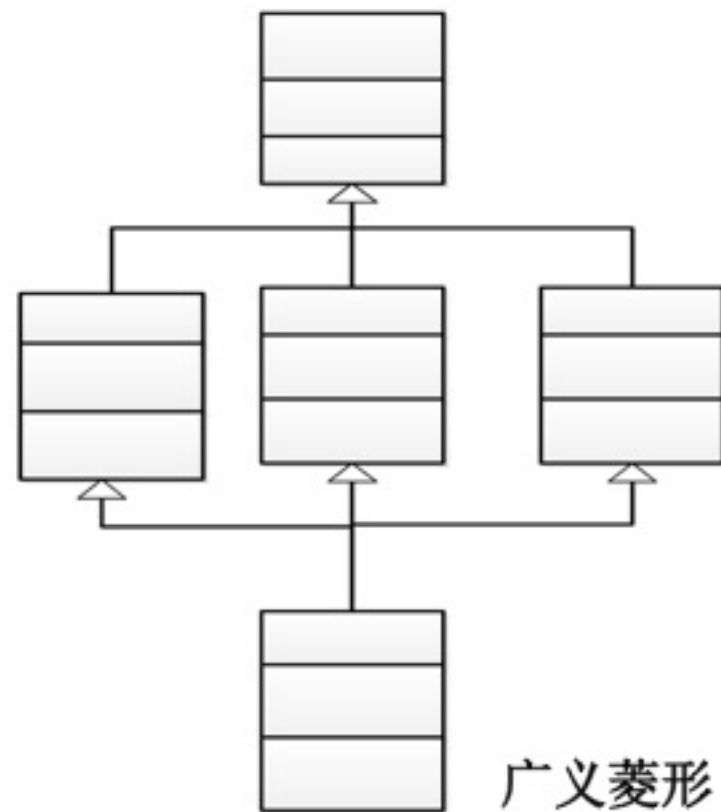


(3) 按编程语言调整继承和多态

起因：OOA强调如实地反映问题域，OOD考虑实现问题，所用语言不支持多继承和多态



多继承
模式



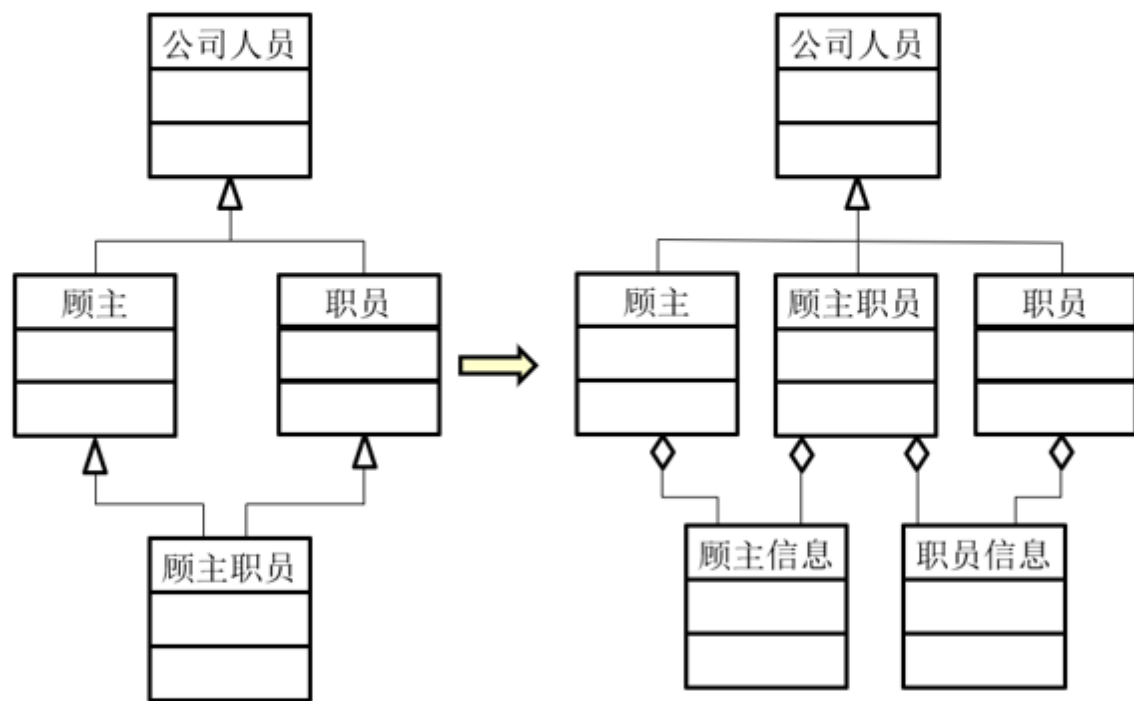
北京大学

如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节

①对继承的调整

方法1：采用聚合，将多继承转换为单继承



- 保持原来多继承结构中的每个类
- 把形成多继承的每一组特殊信息从有关的类中剥离出来，定义为部分对象类
- 再通过聚合使各个特殊类能拥有这些信息


问题：道理何在？



北京大学

如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节



一般—特殊结构和整体—部分结构在某些情况下可以互相变通的道理如下：

尽管继承和聚合反映了现实世界中两种不同的关系，**但是从最终效果来看却存在共性——都是使一个类的对象能够拥有另一个（一些）类的属性和操作。**

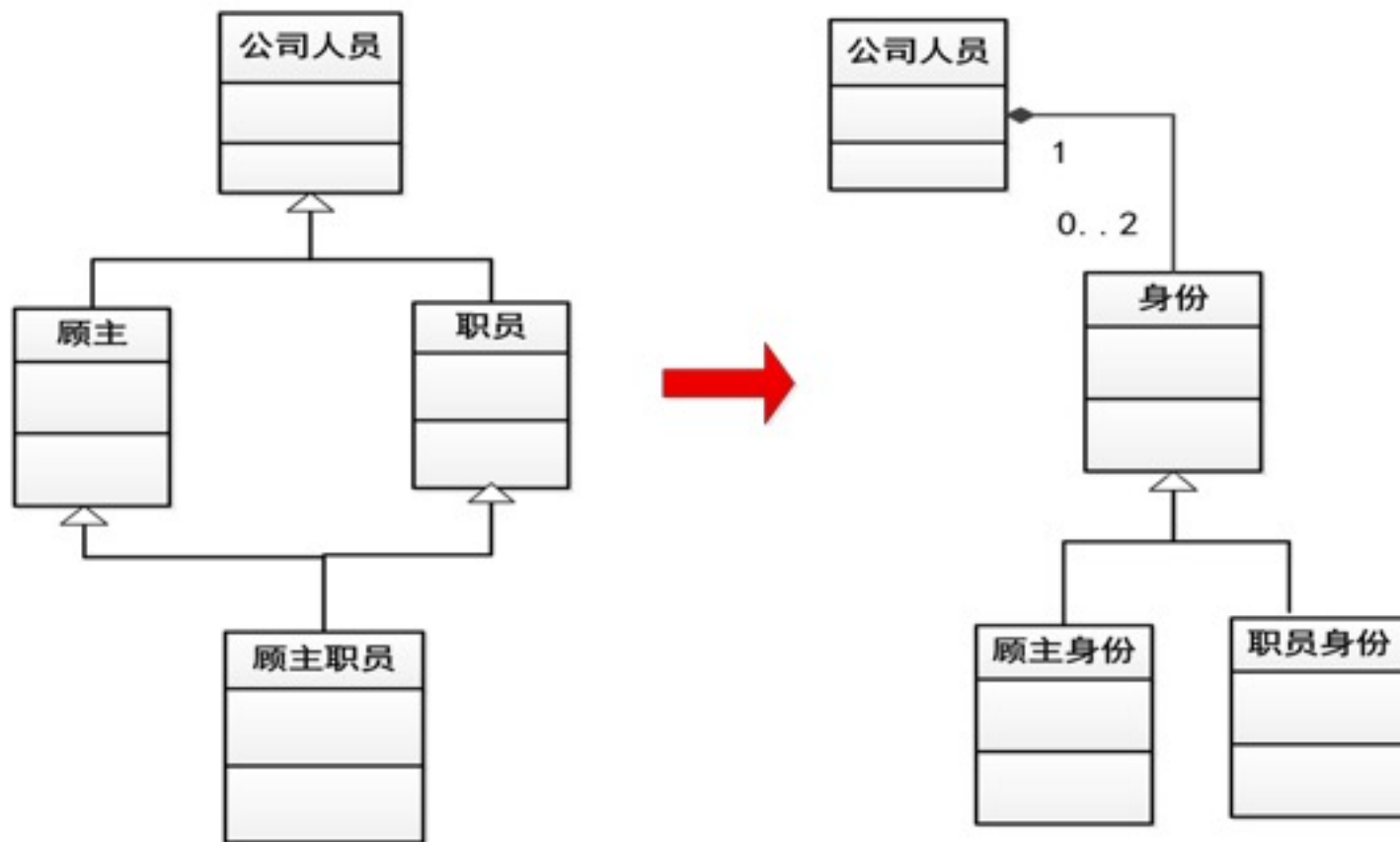


如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节



方法2：重新定义对象类



问题：与方法1有何区别？



北京大学

如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节



②取消继承

若编程语言不支持继承，有两种方法应对。

方法一：把继承结构展平

顾主

顾主职员

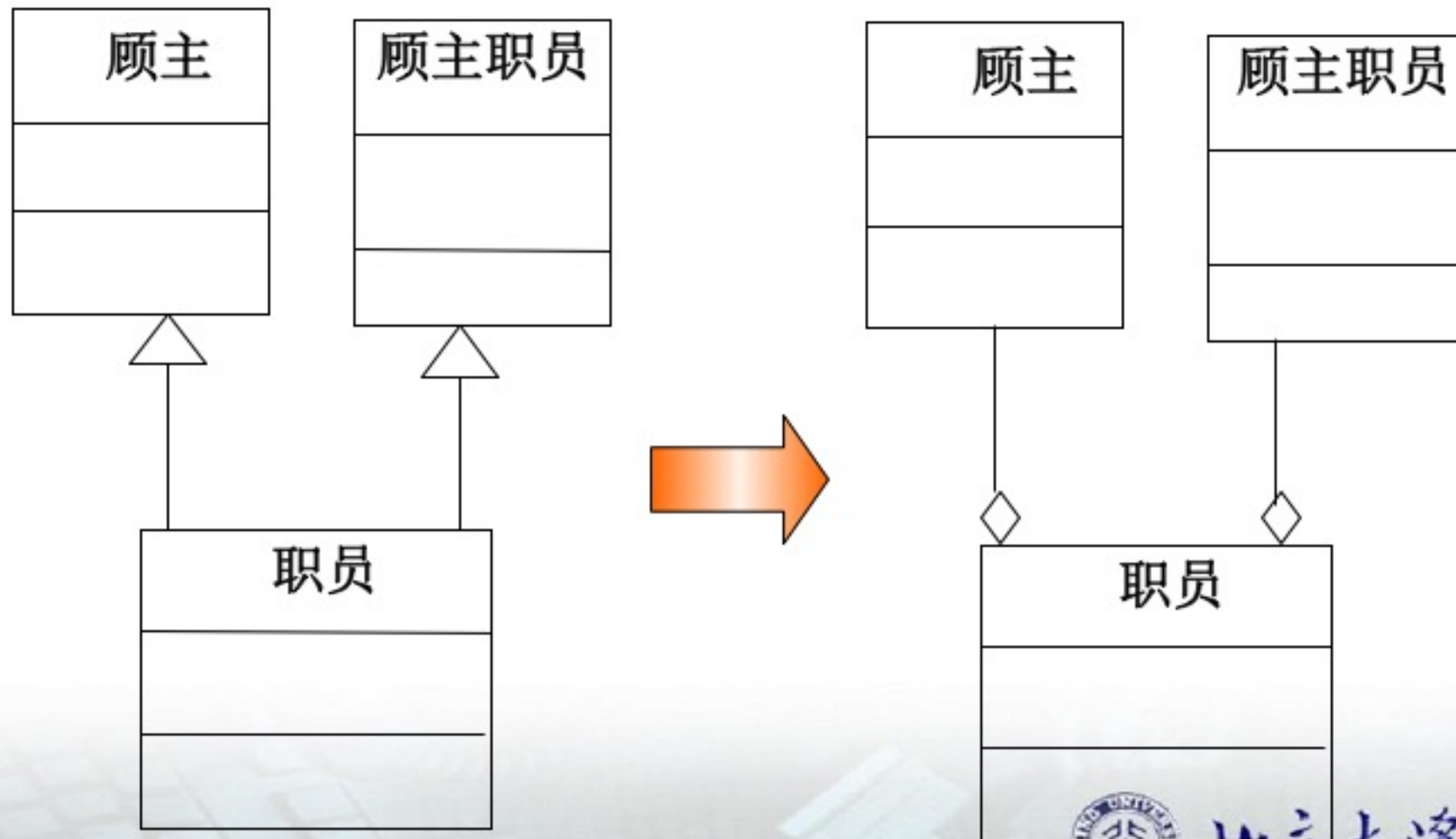
职员



如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节

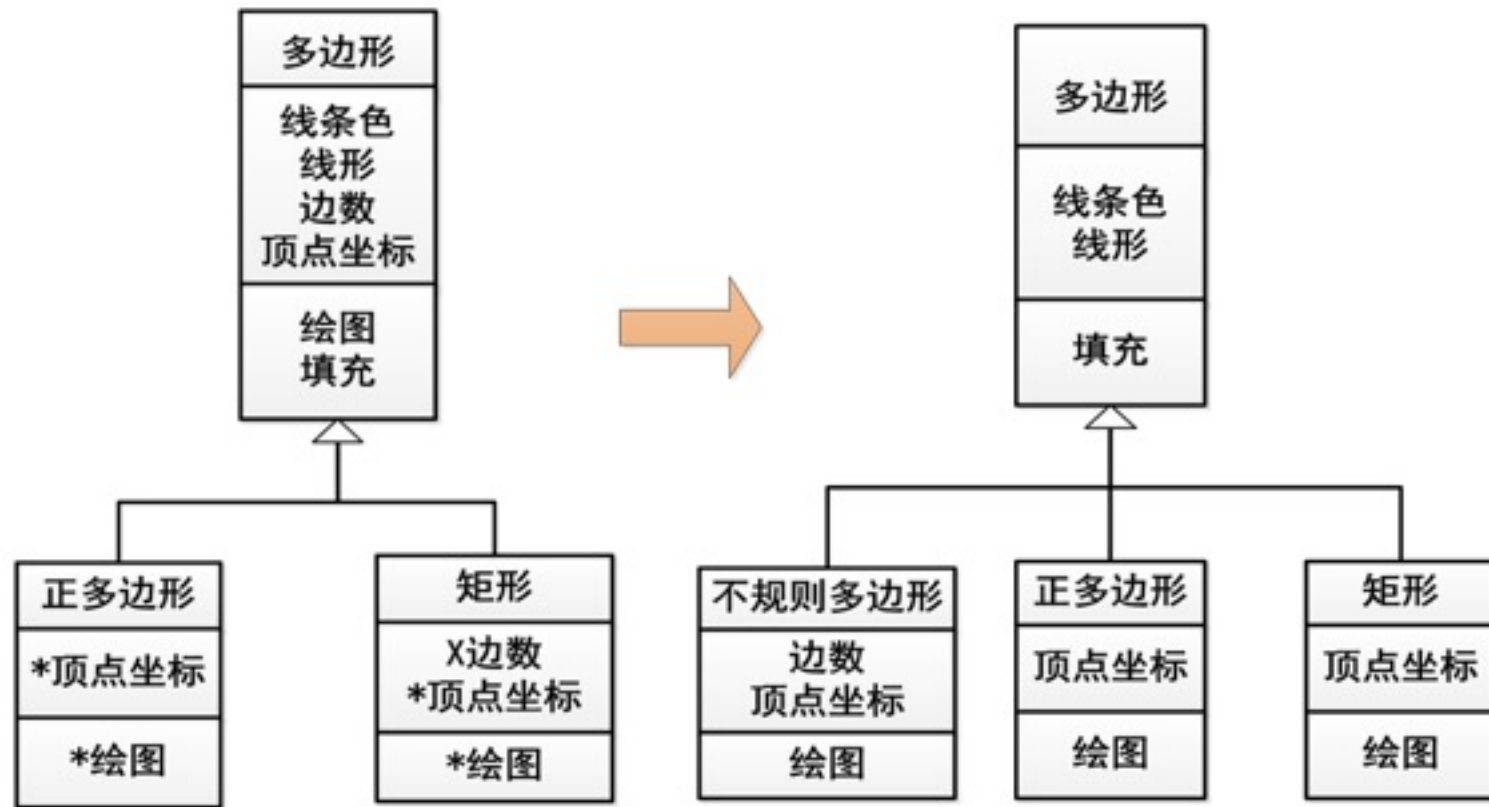
方法2：采用聚合的方法



如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节

③对多态性的调整



多态: 在继承结构中,具有相同的属性和操作,在不同的类中可以具有不同的类型和行为.



北京大学

如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节

注意:多态和重载的区别:

重载是指相同的操作名在同一个类中可以被定义多次,按参数的个数、种类或次序等的不同对它们进行区分。

多态是指在继承结构中,具有相同的属性和操作,在不同的类中可以具有不同的类型和行为。



如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- **提高性能**
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节

(4) 提高性能

a、影响性能的因素：

数据传输时间:主要是在网络环境下不同处理机之间为完成某项系统功能进行必要的数据传输所专用的时间。影响数据传输时间的主要因素有：系统分布方案、网络拓扑结构、从发送者到接收点经由的每一段线路的传送速率和数据拥挤程度。

数据存取时间:是指完成一项系统功能时，在外部存储设备上读取或保存数据所用的时间。影响因素有：存储设备的物理性能（各种设备都有明确的性能指标说明）、访问外存的次数和每次访问的数据量。

数据处理时间：指计算机主机为完成一项功能，进行处理、计算所花费的时间。



如何进行问题域部分的设计

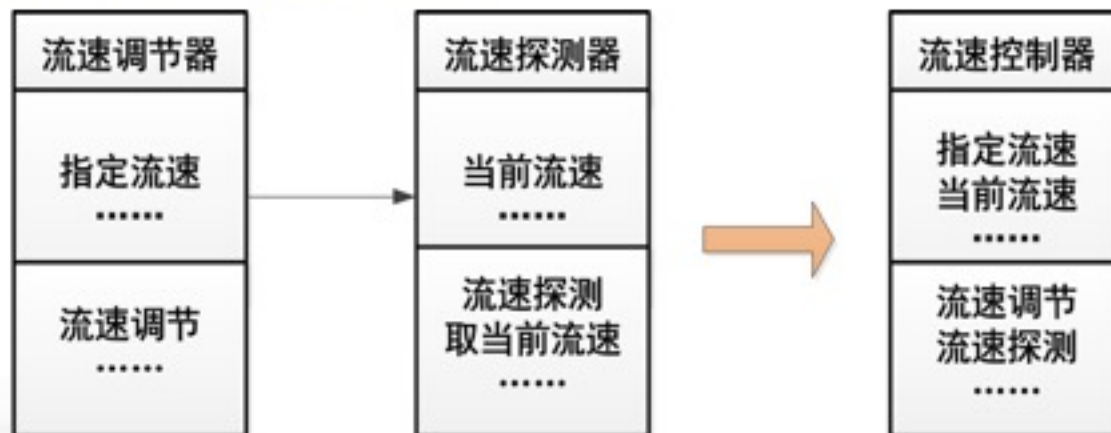
- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- **提高性能**
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节

b、改进性能的设计策略

设计人员为改进性能所做的设计决策，就是对OOD模型采取某些提高性能的策略，只有这些策略仍不能满足性能要求时，才考虑改变计算机、网络、DBMS等基本配置。

***调整对象分布：不同处理机间的数据传输成为性能瓶颈**

***合并通讯频繁的类**



合并前

合并后

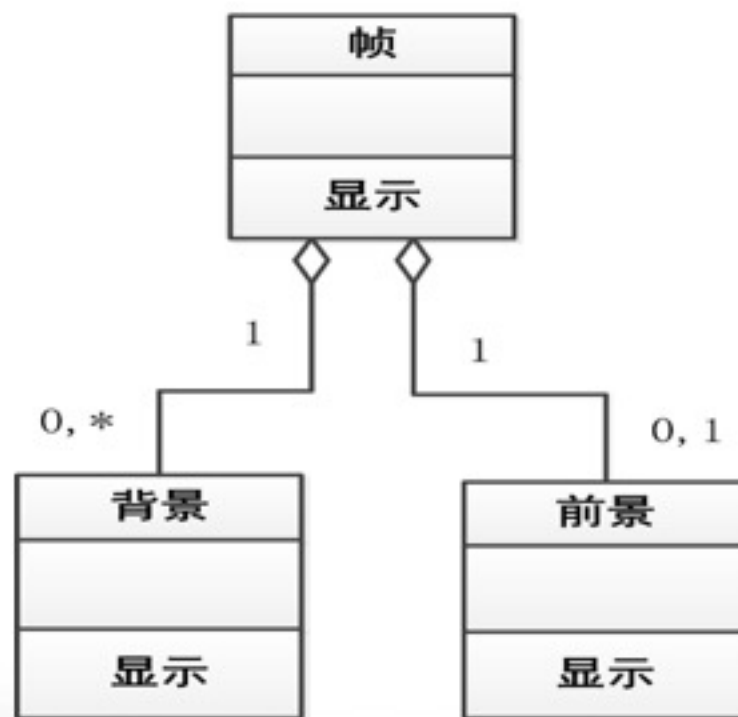


北京大学

如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- **提高性能**
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节

- *增加属性以减少重复计算
- *降低算法的计算复杂性
- *用聚合关系描述复杂类



如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节



(5) 为实现对象永久存储所做的修改

有些类的对象实例需要被永久存储。如果选用文件系统或关系数据库管理系统实现对象的永久存储，则需要对这些类做某些修改，如为实现对象的存储和恢复操作而增加属性和操作。这些修改在数据管理部分设计中介绍。



如何进行问题域部分的设计

- 为复用设计与编程的类而增加结构
- 增加一般类以建立共同协议
- 按编程语言调整继承和多态
- 提高性能
- 为实现对象永久存储所做的修改
- 为编程方便增加底层细节

(6) 为编程方便增加底层成分 ——细化对象的分类

例：将几何图形分成多边形、椭圆、扇形等特殊类

