OOA

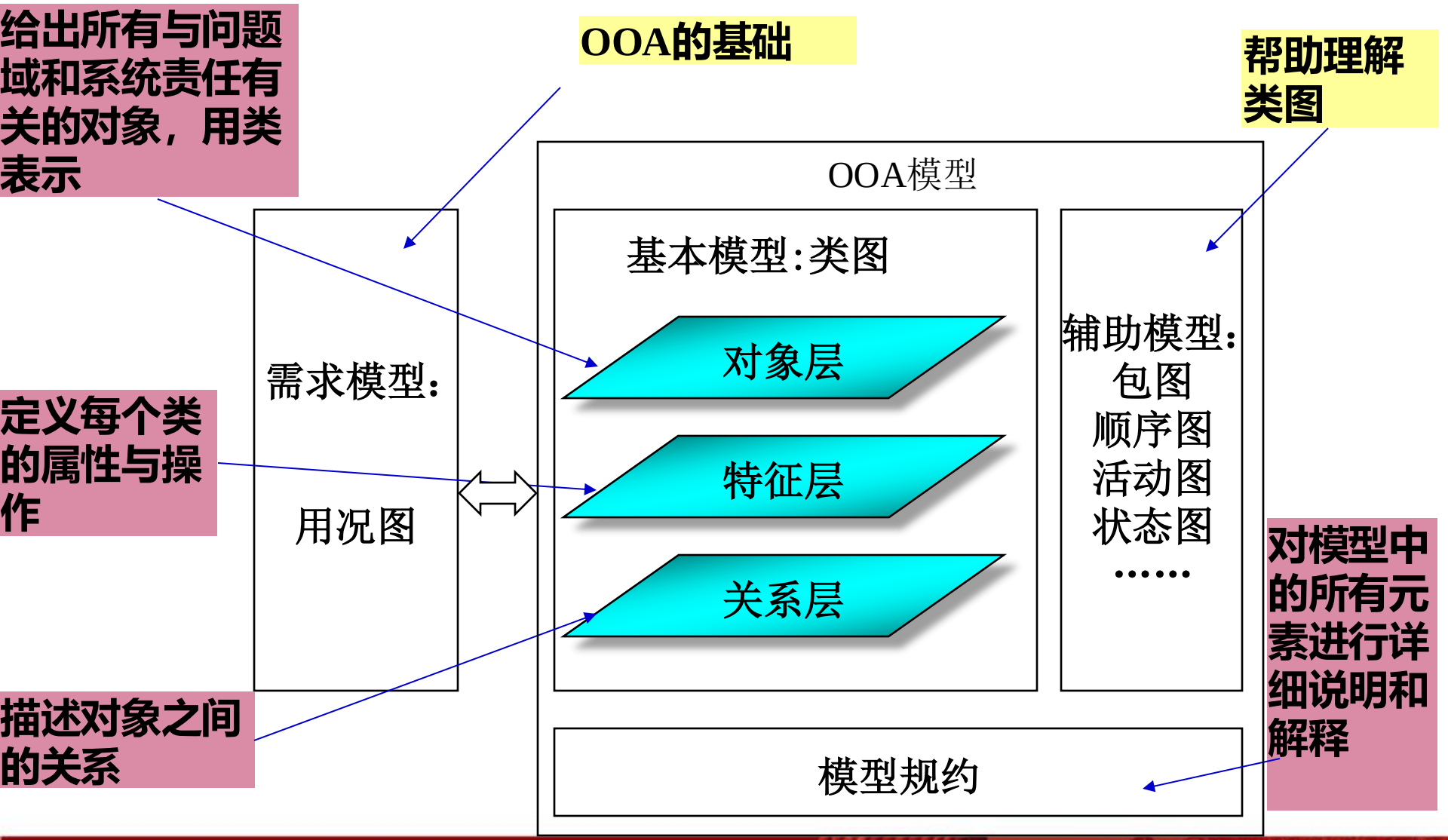
面向对象的分析(Object-oriented analysis, OOA)

1、OOA的基本任务:

运用面向对象方法，对问题域（被开发系统的应用领域）和系统责任（所开发系统应具备的职能）进行分析和理解，对其中的事物和它们之间的关系产生正确的认识，找出描述问题域和系统责任所需的类和对象，定义这些类和对象的属性和操作，以及它们之间所形成的各种关系。最终目的是产生一个符合用户需求，并能够直接反映问题域和系统责任的OOA模型及其规约。

- - 面向对象的分析与设计，邵维忠等编著，清华大学出版社，2013.1.

2、OOA模型


给出所有与问题域和系统责任有关的对象，用类表示
OOA的基础
帮助理解类图
OOA模型
基本模型:类图
对象层
特征层
关系层
需求模型:
用况图
辅助模型:
包图
顺序图
活动图
状态图
……
定义每个类的属性与操作
描述对象之间的关系
对模型中的所有元素进行详细说明和解释
模型规约

3、OOA过程

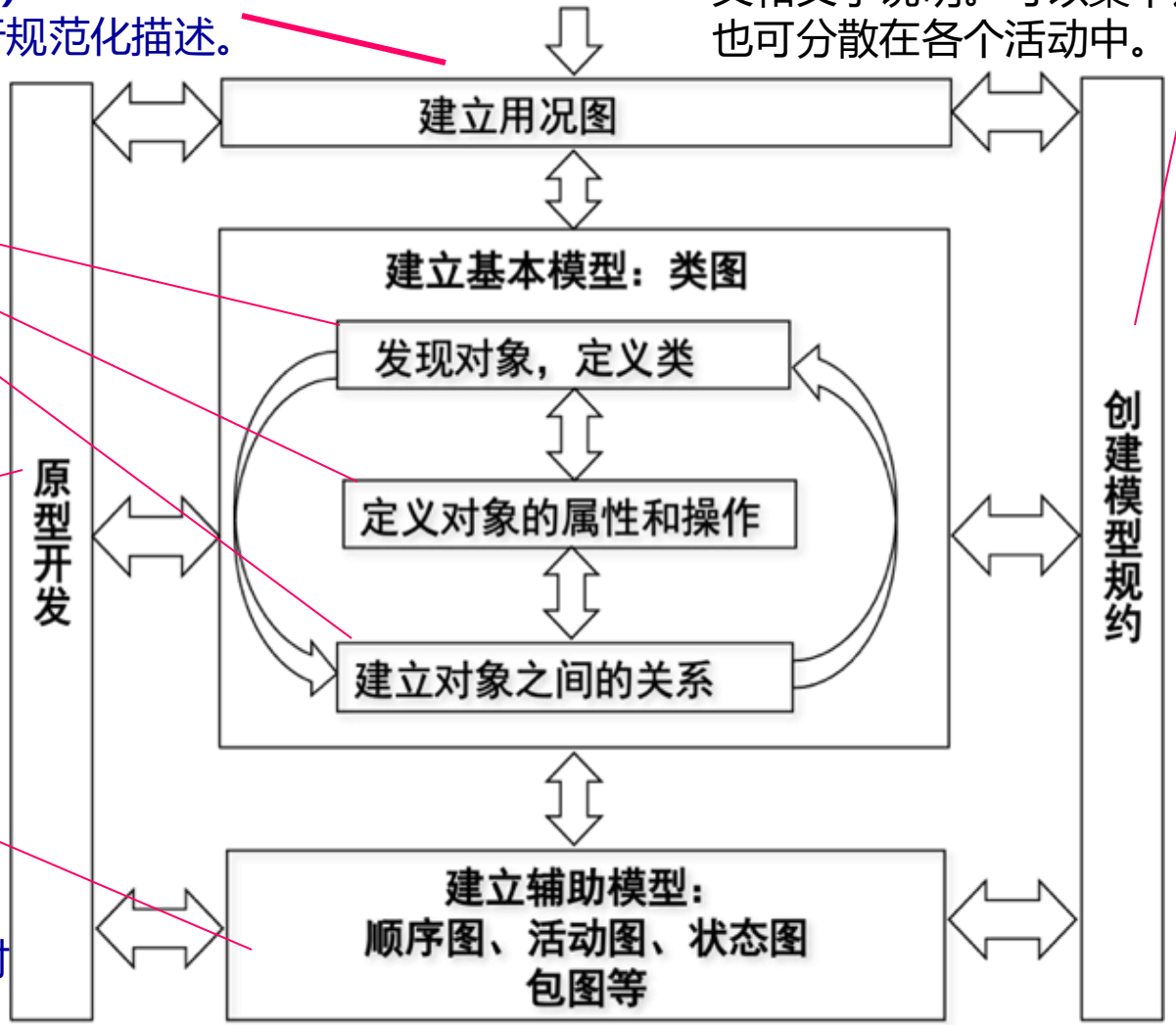
建立模型规约
对模型中的成分进行规范的定义和文字说明。可以集中进行，也可分散在各个活动中。

定义use case（辅助模型，可选）
用use case对用户需求进行规范化描述。

建立类图（基本模型）
* 发现对象、定义对象类
* 识别对象的内部特征
* 识别对象的外部关系

原型开发
结合其他活动反复进行

建立交互图（辅助模型，可选）
对照use case，描述一组对象进行协作时的交互情况和消息的时序关系。

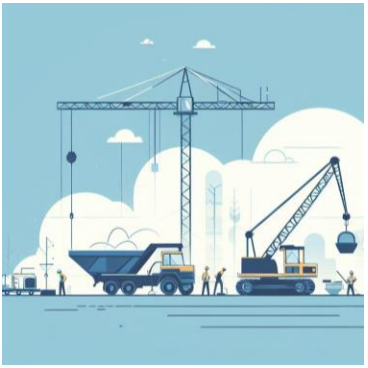


OOA--类图构建

1. 发现对象，定义类
2. 定义对象的属性
3. 定义对象的操作
4. 建立对象之间的关系
5. 创建类的模型规约

发现对象，定义类

- **（1）考虑系统边界**
  - 确定系统边界，就是划出被开发的系统和该系统打交道的人或外部事物之间的明确界限，并确定它们之间的接口。
  - 在系统边界之内，是系统本身所包含的对象。在系统边界以外，是系统外部的参与者。主要是人、设备和外系统三种外部活动者。
  - 考虑系统边界，主要帮助分析人员考虑并确定三种系统参与者是否需要将其作为系统内的对象而抽象为类。
  - 考虑系统边界，可以启发分析人员发现一些与系统边界以外的参与者进行交互，并且处理系统对外接口的对象。



  - 人员：作为系统以外的参与者与系统进行直接交互的各类人员，如系统的操作员、直接使用系统的用户等。
  - 设备：作为系统以外的参与者与系统相连并交换信息的设备。
  - 外系统：与系统相连并交换信息的其他系统。

发现对象，定义类

- **（2）考虑问题域和系统责任**
  - ①考虑问题域

    侧重于客观存在的事物与系统中对象和类的映射。

    可以启发分析人员发现对象的因素包括：人员、组织、物品、设备、抽象事物、事件、文件及结构等。

发现对象，定义类

- **（2）考虑问题域和系统责任**
  - ①考虑问题域
    - **人员**：（a)需要由系统保存和管理其信息的人员，如户籍管理系统中的每个居民；（b)应该在系统中完成某些功能，提供某些服务的人员，如户籍管理员。符合上述情况之一者，应考虑用相应的人员对象来描述。
    - **组织**：在系统中发挥一定作用的组织结构。如计算机学院。
    - **物品**：需要由系统管理的各种物品。如经营的商品等。
    - **设备**：在系统中动态地运行、由系统进行监控或供系统使用的各种设备、仪表、机器及运输工具等。
    - **抽象事物**：指没有具体的物理形态，却对用户的业务具有实际意义的逻辑上的事物。例如法律条文、生产计划等。
    - **事件**：指那些需要由系统长期记忆的事件。例如：商品订货系统的贸易成交事件，保险业务系统的投保事件等。
    - **文件**：泛指在人类日常的管理和业务活动中使用的各种各样的表格、档案、证件和票据等文件。
    - **结构**：当发现一个对象时，从现实世界中与这些事物有分类关系和构成关系等结构出发，发现其他对象。

发现对象，定义类

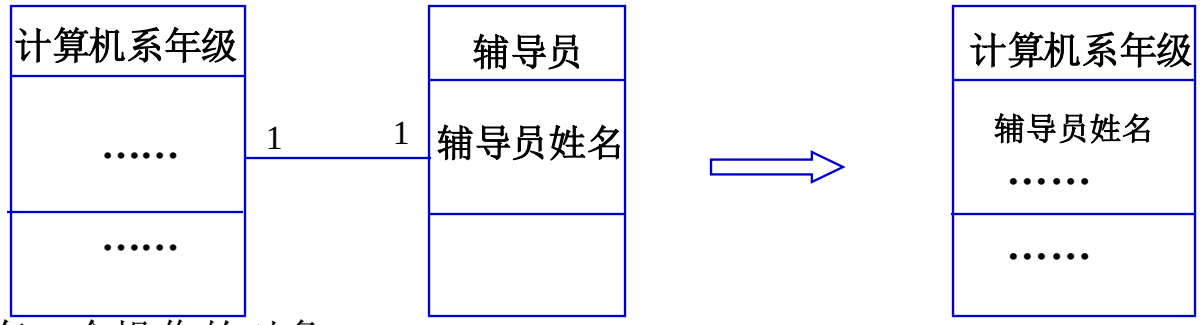
- **（2）考虑问题域和系统责任**
  - ②考虑系统责任
    - 根据用户需求描述，确定系统责任内的每一项功能是由哪个类，或由哪几个类来完成。
    - 检查每一项功能需求是否有相应的对象提供，发现新的对象。
    - 可以通过用况图中动作的模拟执行顺序来帮助系统分析人员逐一落实完成每项功能的对象或类，也可以通过分析完成一项复杂功能的多个对象之间的交互顺序来查找遗漏的对象或类。
    - 也可以将原来在问题域中发现的对象根据系统责任进一步确定是否应该保留。

发现对象，定义类

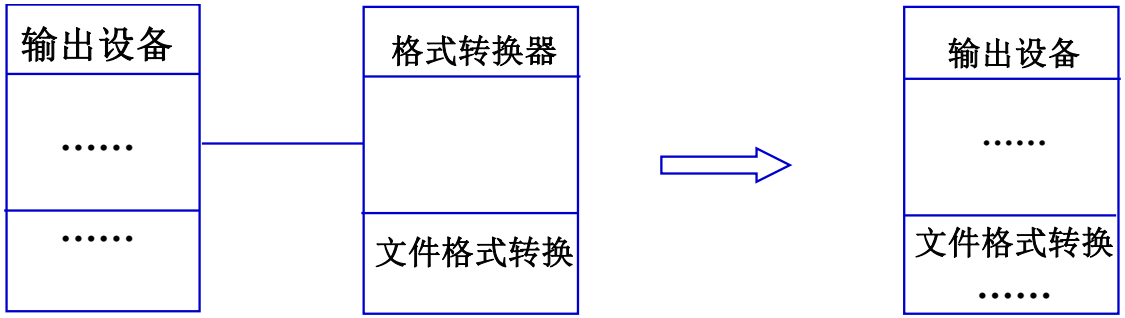
- **（3）审查与筛选**
  - ①舍弃无用的对象
    - 通过属性判断：是否通过属性记录了一些对参与者或对系统的其他对象有用的信息？（这个对象所对应的事物，是否有些信息需要在系统中进行保存和处理？）
    - 通过操作判断：是否通过操作提供了某些有用的功能？（这个对象所对应的事物，是否有某些行为需要在系统中模拟，并在系统中发挥一份作用？）
    - 二者都不是——无用

发现对象，定义类

- **（3）审查与筛选**
  - ②对象的精简
    - 只有一个属性的对象



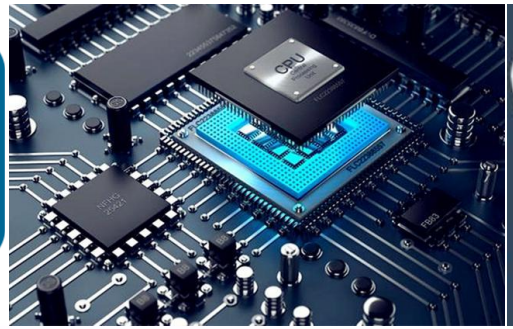
    - 只有一个操作的对象



发现对象，定义类

- **（3）审查与筛选**
  - ③与实现条件有关的对象，推迟到OOD考虑
    - 例如：与
      - 图形用户界面（GUI）系统、
      - 数据管理系统、
      - 硬件
      - 操作系统

      有关的对象
    - OOA模型应独立于实现环境



OOD

发现对象，定义类

- **（4）识别主动对象**
  - ①考虑问题域和系统责任哪些对象需呈现主动行为？
  - ②考虑系统边界以外的参与者与系统中哪些对象直接进行交互？如果一个交互是由系统外的参与者发起的，第一个处理该交互的对象是主动对象
  - ③考虑系统的功能需求，控制线程的起点在哪个对象？

在分析阶段不能完全确定

发现对象，定义类

- **（5）对象分类，建立类图中的类**
  - ①对象分类
    - 使用问题域和系统责任知识，为每组具有相同属性和操作的对象定义一个类
  - ②异常情况的检查和调整
    - **类的属性或操作不适合全部对象实例**
      - 例：“汽车”类的“乘客限量”属性➔问题：分类不够详细——进一步划分特殊类
    - **属性及操作相同的类**
      - 经过抽象，差别很大的事物可能只保留相同的特征
      - 例如“吸尘器”和“电子琴”作为商品销售➔考虑能否合并为一个类
    - **属性及操作相似的类**➔考虑能否提升出一个一般类或部分类
      - 例如：轿车和货车➔提取增加一般类“汽车”
      - 例如：机床和抽风机➔提取部分类“电动机”
    - **同一事物的重复描述**
      - 例如：“职员”和“工作证”➔取消其中一个

发现对象，定义类

- **（5）对象分类，建立类图中的类**
  - ③类的命名
    - 使用名词或名词性短语，避免无意义的符号和数字
    - 反映个体而不是群体（如书而非书籍）
    - 适合该类及其特殊类的全部对象实例
    - 使用问题域通用、规范的词汇
    - 在中国：可用中、英文双重命名
  - ④建立类图的对象层
    - 用类符号表示每个对象类，填写类的名称；
    - 对于已经确认的主动对象，注意标注为主动类
    - 填写类规约中关于每个类的详细信息；
    - 发现的属性与操作，可以随时加到类符号中。

OOA--类图构建

1. 发现对象，定义类
2. 定义对象的属性
3. 定义对象的操作
4. 建立对象之间的关系
5. 创建类的模型规约

定义对象的属性

- **（1）识别属性的策略**
  - 按常识这个对象应该有哪些属性？
    - 例如人的姓名、职业、地址等
  - 在当前的问题域中，对象应该有哪些属性？
    - 例如商品的条形码
  - 根据系统责任，这个对象应具有哪些属性？
    - 例如持卡人的使用地点
  - 建立这个对象是为了保存和管理哪些信息？
  - 对象为了实现操作的功能，需要增设哪些属性？
    - 例如传感器对象，为了实现其定时采集信号的功能，需要一个“时间间隔”属性；为了实现其报警功能，需要一个“临界值”属性

定义对象的属性

- **（1）识别属性的策略**
  - 对象是否需要通过专设的属性描述其状态？
    - 例如设备对象，在关闭、待命、运行、故障等不同状态将呈现不同的行为，需要为其设置一个“状态”属性
  - 用什么属性表示聚合和关联？
    - 对于关联
      - 应该在关联一端的类中定义一个属性，来指出另一端的哪个对象与本端的对象发生关联，其数据类型是指向另一端对象的指针或对象标识。
    - 对于聚合关系
      - 紧密、固定的聚合：在整体对象中定义一个属性，它是一个嵌套在整体对象中的部分对象，所以它的数据类型就是部分对象类。
      - 松散的聚合：既可以在整体对象中定义一个属性指向另一端的哪个对象是它的组成部分，也可以在部分对象中定义一个属性指向另一端的哪个对象是它的整体。其方向取决于聚合关系的多重性。

定义对象的属性

- **（2）审查和筛选**
  - 是否体现了系统中有用的信息
    - 例：书的重量
  - 是否描述对象本身的特征
    - 例：课程—电话号码
  - 是否破坏了对象特征的“原子性”
  - 是否可通过继承得到
  - 是否可以从其它属性直接导出
  - 与实现条件有关的问题都推迟到OOD考虑
    - 规范化问题（OOA中定义的对象属性可以是任何数据类型，数据类型的规范化工作在OOD中考虑）
    - 对象标识问题
    - 性能问题（如为了提高操作的执行速度，增加一些属性来保持操作的阶段性执行结果）

定义对象的属性

- **（3）属性的命名：原则与类的命名相同**
  - 使用名词或名词性短语
  - 使用规范的、问题域通用的词汇
  - 避免使用无意义的字符和数字
  - 语言文字的选择要和类的命名要一致
  - 定位原则：一个类的属性必须适合这个类和它的全部特殊类的所有对象，并在此前提下充分运用继承

定义对象的属性

- **（4）属性的详细说明**

  要在类规约中对属性进行详细说明，其中包括：属性的解释、数据类型和具体限制等。

  – 属性的文字解释：例如“课程”对象的“学时”属性，其解释为“课堂讲授学时数，每学时为50分钟”

  – 属性的数据类型：常用的数据类型；表示整体-部分结构或关联的属性类型可以是类或某一类对象的指针

  – 属性所体现的关系：用于表示整体-部分关系或关联的属性，应该特别指明并加以解释

    - 例如对“课程”对象的“主讲教师”属性，可说明为：“表示课程与教师对象间的关联，指出该课程由哪个教师主讲

  – 实现要求及其它：如属性的取值范围、精度要求、初始值、度量单位、数据完整性及安全性要求、存取限制条件等

    - 如果一个属性实现时应作为类属性处理，也要在这里明确指出

OOA--类图构建

1. 发现对象，定义类
2. 定义对象的属性
3. 定义对象的操作
4. 建立对象之间的关系
5. 创建类的模型规约

定义对象的操作

- **（1）识别操作的策略**

  为了明确OOA应该定义对象的哪些操作，首先区分对象行为的不同类型：

  - ①系统行为
    - 例：创建、删除、复制、转存
  - ②对象自身的行为——算法简单的操作
    - 例：读、写属性值
  - ③对象自身的行为——算法复杂的操作计算或监控

  仅用于操纵类属性的操作叫做类范围的操作，其余的操作叫做实例范围的操作。

定义对象的操作

- **（1）识别操作的策略**
  - ①考虑系统责任
    - 要逐项审查用户需求中提出的每一项功能要求，看它应由哪些对象来提供，从而在该对象中设立相应的操作
  - ②考虑问题域
    - 对象在问题域对应的事物有哪些行为？
  - ③分析对象状态
    - 对象状态的转换，是由哪些操作引起的？
  - ④追踪操作的执行路线
    - 模拟操作的执行，并在整个系统中跟踪

定义对象的操作

- **（2）审查与调整**
  - 审查

    对象的每个操作是否真正有用，即是否直接提供系统责任所要求的某项功能？或响应其它操作的请求间接地完成这种功能的某些局部操作？
  - 调整

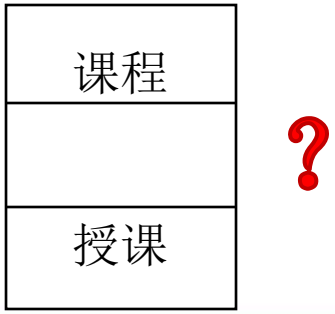
    取消无用的操作

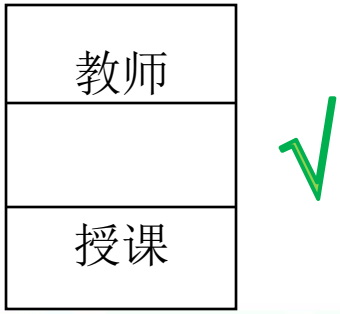
  - 审查

    是不是高内聚的，即一个操作只完成一项单一的、完整的功能
  - 调整：
    - ①拆分（一个操作中包括了多项可独立定义的功能）
    - ②合并（一个独立的功能分割到多个对象操作中完成）

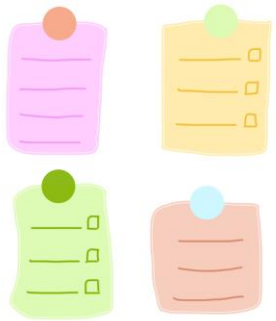
定义对象的操作

- **（3）操作的命名和定位**
  - 命名：动词或动宾结构
  - 定位：与实际事物一致
    - 例：售货员——售货，商品——售出
  - 注：在“一般--特殊”结构中的位置
    - 通用的操作放在一般类，专用的操作放在特殊类，一个类中的操作应适合这个类及其所有特殊类的每一个对象实例。





定义对象的操作



- **（4）操作的详细说明**

  在类规约中，要对操作进行详细说明，包括操作的解释、操作的特征标记、主动性、多态性、操作要发送的消息和约束条件等。

  – ①操作的文字解释：解释该操作的作用和功能。
  – ②操作名、输入输出参数、参数类型：给出操作的入口消息格式。
  – ③消息发送：指出在这个操作执行时，需要请求哪些其他的对象操作。内容包括接收消息的对象类名以及执行这个消息的操作名。
  – ④约束条件：操作的执行的前置条件、后置条件以及执行时间的要求等事项说明。
  – ⑤操作流程：对于功能比较复杂的操作，要给出一个操作的流程图或活动图，表明该操作是怎样执行的。

案例分析：超级市场销售管理系统

超级市场业务管理系统的子系统，只负责前台的销售管理

- **功能需求：**
  - 为顾客选购的商品计价、收费、打印清单。
  - 记录每一种商品的编号、单价及现有数量。
  - 帮助供货员发现哪些商品将要脱销，以及时补充货源。
  - 随时按上级系统的要求报告当前的款货数量、增减商品种类或修改商品定价。
  - 交接班时结算货款数目，报告上级系统。

案例分析——发现对象类

- **1、通过建立系统的use case 图，在系统边界以外与系统进行交互的活动者有收款员、供货员和它的上级系统。这样，可以启发我们发现一些对象：**
  - **收款机：**该对象直接与收款员这种活动者进行交互，模拟收款员的登陆、售货和结算等行为。由于某些行为是收款员的意愿主动发生的，所以将“收款机”定义为主动对象。
  - **供货员：**此类对象用来与实际的供货员进行交互（提醒他们及时补充货物）并模拟他们的行为（在增加货物时修改系统中的商品数量），这些行为是从系统内部引发的，所以它们是被动对象。
  - **上级系统接口：**用来处理与上级系统的交互。它的某些行为（如查账、更改商品的种类和价格）是由上级系统（而不是从本系统内部）引发的，所以它也应该是主动对象。

案例分析——发现对象类

- **2、考虑系统问题域内部的事物和系统责任可以发现下述对象：**
  - **商品：**这是系统中最明显的对象。每一个对象实例表示一种商品，记录该商品的名称、价格、数量等信息，并通过相应的服务动态地保持这些信息的准确性。
  - **特价商品：**这是一类特殊的商品，该类商品在指定的时间内按特殊价格销售，它有自己特殊的属性。

案例分析——审查和筛选对象类

- 计量商品：这是另一类特殊的商品，它的包装是不标准的，或没有包装，需要在收款时按照它们的重量、长度或容积等单位进行计量，并按计量结果计算其价格。
- 商品一览表：考虑系统责任，为了在收款时能根据输入的商品编号快速地找到相应商品的信息，需要设计一个“商品一览表”对象，它保持一个商品目录表，并提供对商品项的检索及增删等功能。
- 销售事件：顾客购买一组商品，只要通过一次计价收款完成的，就称为一个销售事件。每个这样的事件都需要保留一段时间，以便汇成账目并在必要时复查。所以，设立“销售事件”对象。
- 账册：记录一个收款员在一个班次内经手的所有销售事件的款、货账目，负责向上级系统报账，并在换班时进行账目交接。它的一个对象实例只针对一个收款员的一个班次，不是总账（总账在上级系统中）

案例分析——审查和筛选对象类

按照常识，在一个超级市场中收款员和管理人员都应该是一些值得考虑的对象，假如，我们把“收款员”和“经理”列为候选的对象类，现在考虑对象类的筛选，看看是否有必要保留这两个类？

案例分析——审查和筛选对象类

按照常识，在一个超级市场中收款员和管理人员都应该是一些值得考虑的对象，假如，我们把“收款员”和“经理”列为候选的对象类，现在考虑对象类的筛选，看看是否有必要保留这两个类：

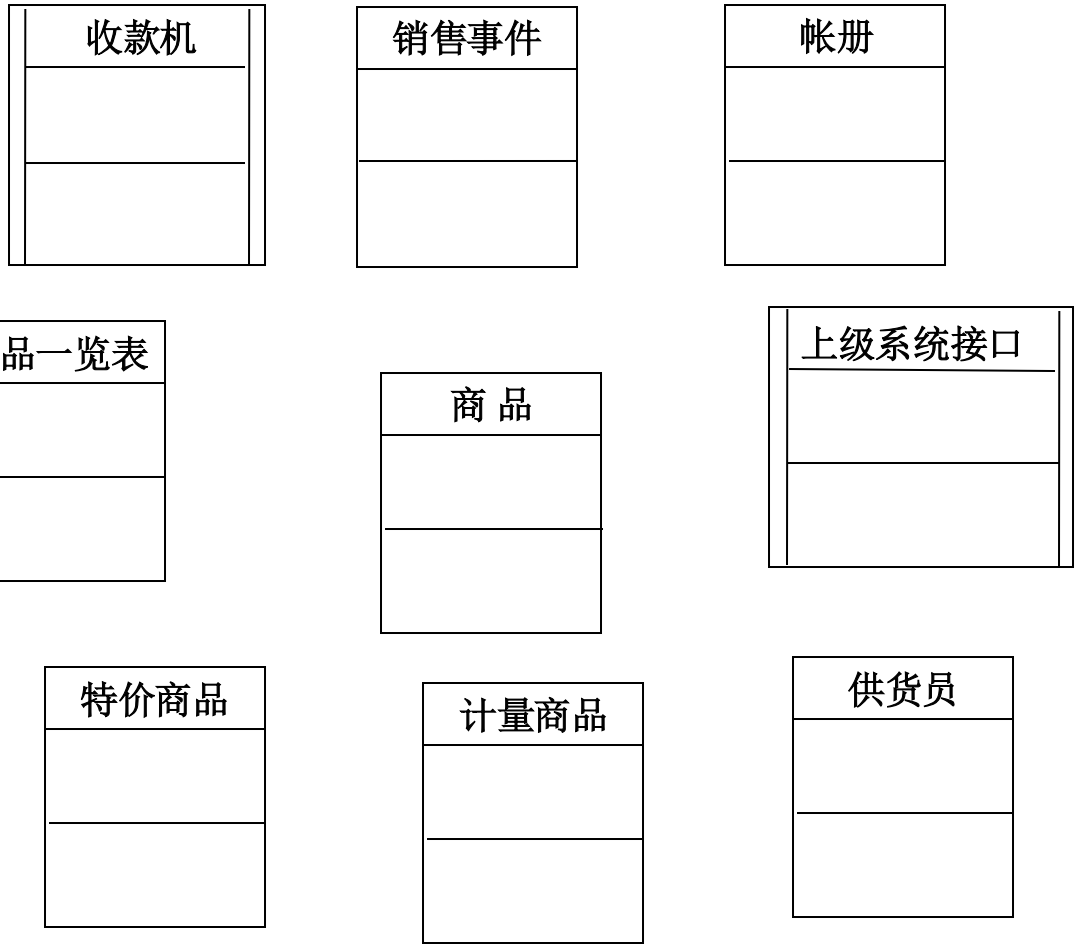
- 1、本系统的功能需求没有要求对各类人员的信息进行计算机管理，所以各类人员对象是否有必要存在只是看这些人员的行为和个人信息是否对系统功能的履行起到一定的作用；
- 2、需求中没有包括对经理的工作进行计算机处理或提供辅助的支持，经理本人的信息对于完成需求中规定的业务处理功能也没有用处，所以不必设立“经理”对象；
- 3、收款员与系统的功能需求有密切关系，他们是与系统对话的活动者，系统应提供相应的对象处理这种对话，但在上述发现的对象中，“收款机”对象就是进行这种处理的，如果愿意，也可以把“收款机”对象改名为“收款员”对象，但没有必要设立两类对象。

筛选原则：系统中任何对象都是为了提供某些信息或履行某些功能，如果没有这些用途，则这种对象就没有必要在系统中存在。

案例分析——审查和筛选对象类

- **OOA模型对象层：**

为什么没有“收款员”和“经理”这两类对象？



案例分析——分析每个类的属性和操作

- **（1）收款机：**
  - 属性：应指明当前是哪个收款员在本台收款机上工作（本班收款员），她或他在本次工作的开始和结束时间（开始时间和结束时间）。
  - 操作：
    - 登录：本班收款员开始工作，它是一个主动操作；
    - 售货“循环地为每个顾客计价收款；
    - 结账：在收款员下班或交班时结算本班的账目。
- **（2）商品：**
  - 属性：商品的编号、名称、单价、架上数量及下限
  - 操作：
    - 售出：从架上数量减去已售出商品的数目，若剩余的数目低于下限则向“供货员”对象发消息；
    - 补充：当供货员补充了一些商品时，把补充数量与该种商品原先的架上数量相加；
    - 价格更新：由“上级系统接口”对象请求修改商品的价格。

案例分析——分析每个类的属性和操作

- **（3）特价商品：**
  - 继承了“商品”的所有属性和操作，同时具有自己的特殊属性（开始日期和结束日期），指明实行特价的时间范围；
- **（4）计量商品：**
  - 继承了商品的所有属性和操作。
  - 属性：“单价”属性的语义发生了变化，指的是一个计量商品的价格。补充的特殊属性有“计量单位”和“计价方式”。
  - 操作：对于从“商品”类中继承的操作要重新定义，因为算法发生了变化。
- **（5）商品一览表：**
  - 属性：是一个汇总店内所有商品的“商品目录”，它是一个数组，每一个元素包括一件商品的编号和指向“商品”对象的指针（或对象标识）
  - 操作：
    - 检索：通过编号查找相应的商品对象；
    - 种类增删：增添或删除“商品目录”中的商品项；

案例分析——分析每个类的属性和操作

- **（6）供货员：**
  - 属性：一个缺货记录表，每当某种商品的架上数量低于其下限时，就在这个登记表种记录下来。
  - 操作：
    - 缺货登记：将告缺的商品名称及编号记到登记表中；
    - 供货：在供货员补充了货物之后，向“商品”对象发服务请求更改其数量，并删除缺货登记表中相应的条目。
- **（7）销售事件：**
  - 属性：
    - 收款人：记录由哪个收款员在哪台收款机上处理这个销售事件；
    - 购物清单：记录顾客选购的每件商品的编号、名称、数量及价格；
    - 应收款：累计所有被选购商品的价格总和；
    - 售出日期及时间
  - 操作：
    - 销售计价：逐条记录商品清单，并累计应收款数；
    - 入账：将本次销售事件的信息计入账册；

案例分析——分析每个类的属性和操作

- **（8）账册：**
  - 属性：记录一个收款员在一个班次中的每个销售事件，并累计其销售收入。实际上，用一组指针（或对象标识）指向每个销售事件（销售时间表），就可得到它们的明细信息。因此，“账册”与“销售事件”有聚合关系。前班结余、本班结余、上交款，以及该账册开始使用和结账日期和时间等。
  - 操作：
    - 接班：记录从上一班收款员那里接收了多少未上交的货款（因为收款机上总需要保留一些零钱）；
    - 记账：记录每一个“销售事件”对象，并累计其收入金额；
    - 报账：向上级系统报账；
    - 交班：向上级系统报账，记录上交的款数和移交给下一班收款员的款数；

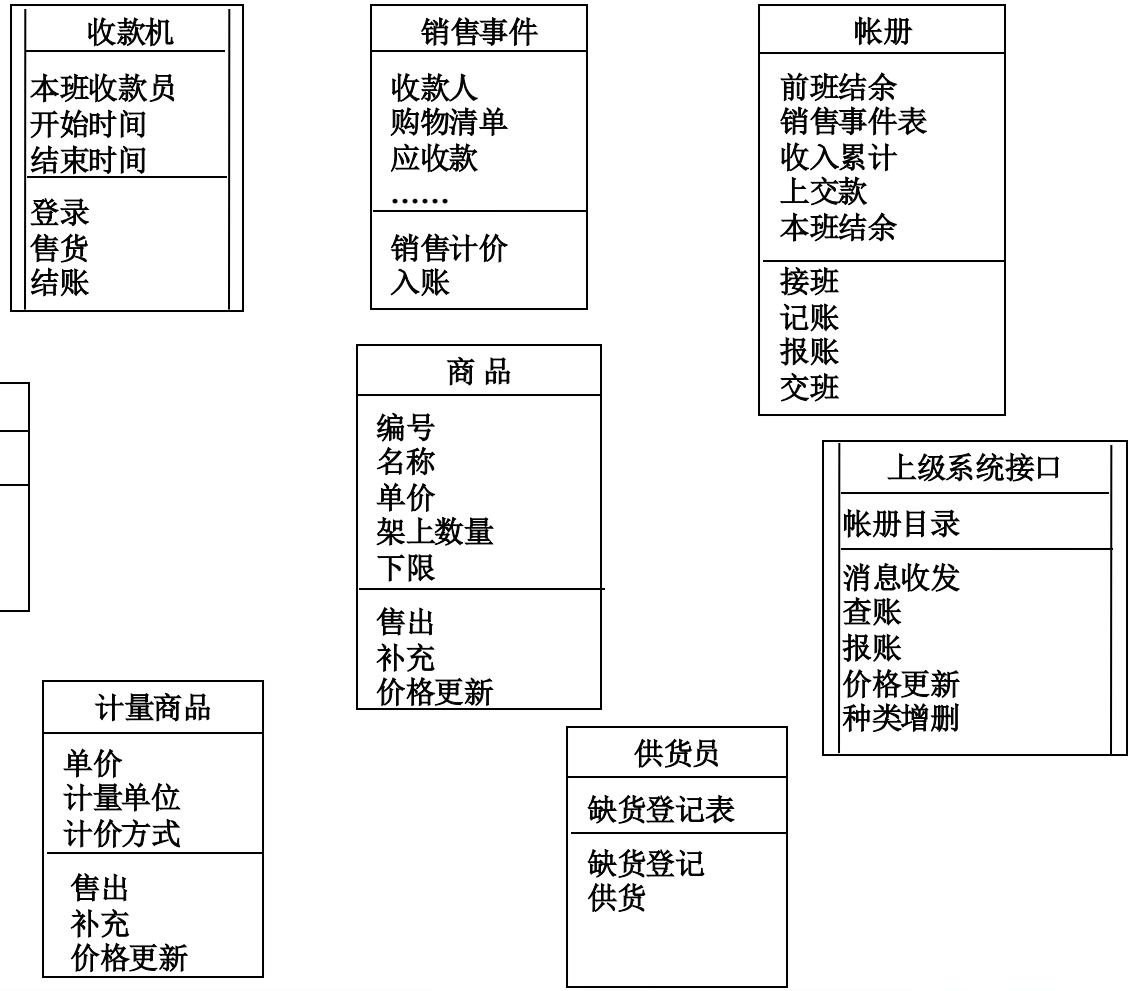
案例分析——分析每个类的属性和操作

- **（9）上级系统接口：**
  - 属性：设立一个“账册目录”属性，记录店内所有正在使用和已经结算的账册的指针信息，以便及时找到它们；
  - 操作：
    - 消息收发：负责与上级系统通信，并通过请求其他对象类的服务完成上级系统要求处理的事项，该操作是主动操作；
    - 查账：按上级系统的要求查阅账目并报告结果；
    - 报账：从本系统向上级系统报告账目；
    - 价格更新：按照上级系统传来的信息，更新指定商品的单价；
    - 种类增删：按照上级系统的信息增添或删除商品对象及其在商品一览表中的条目。

- **根据以上对属性和操作的分析，得到该系统OOA模型的特征层，如下图所示：**

案例分析——分析每个类的属性和操作

- OOA模型的特征层



OOA--类图构建

1. 发现对象，定义类
2. 定义对象的属性
3. 定义对象的操作
4. 建立对象之间的关系
5. 创建类的模型规约

建立泛化（继承）关系

- **（1）识别泛化（继承）关系的策略**
  - ①学习当前领域的分类学知识
    - 因为问题域现行的分类方法往往比较正确地反映事物的特征、类别以及各种概念的一般性和特殊性。按照问题域已有的分类方法，可以找出一些与它对应的继承关系。
  - ②按常识考虑事物的分类
    - 如果问题域没有可供参考的现行分类方法，可以按照自己的常识，从各种不同的角度考虑事物的分类，从而发现继承关系。
    - 例如对于“人员”可以从以下几种角度去分类：
      - 青年人、成年人与老年人；
      - 男人与女人；
      - 黄种人、白种人和黑种人；
      - 在职人员与离退休人员；
      - 正式职工与临时工；

从不同的角度考虑问题域中事物的分类，可以形成一些建议一般-特殊关系的初步设想，从而启发自己发现一些确实需要的一般-特殊关系。

建立泛化（继承）关系

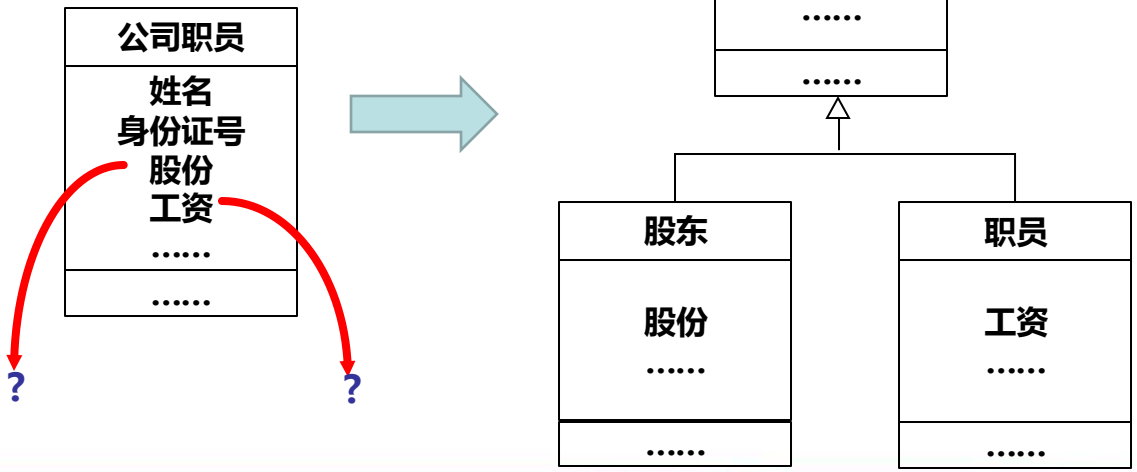
- **（1）识别泛化（继承）关系的策略**
  - ③使用继承的定义

    使用两种思路去发现继承关系：
    - （a）一种思路是把每个类看作是一个对象集合，分析这些集合之间的包含关系，如果一个类是另一个类的子集（例如“职员”是“人员”的子集，“轿车”是“汽车”的子集），则它们应组织到同一个一般-特殊关系中
    - （b）看一个类是不是具有另一个类的全部特征，这又包括以下两种情况：
      - 一种是建立这些类时已经计划让某个类继承另一个类的全部属性与操作，现在应建立继承关系来落实；
      - 另一种是起初只是孤立地建立每个类，现在发现一个类中定义的属性与操作全部在另一个类中重新出现，此时应考虑建立继承关系，把后者作为前者的特殊类，以简化其定义。

建立泛化（继承）关系

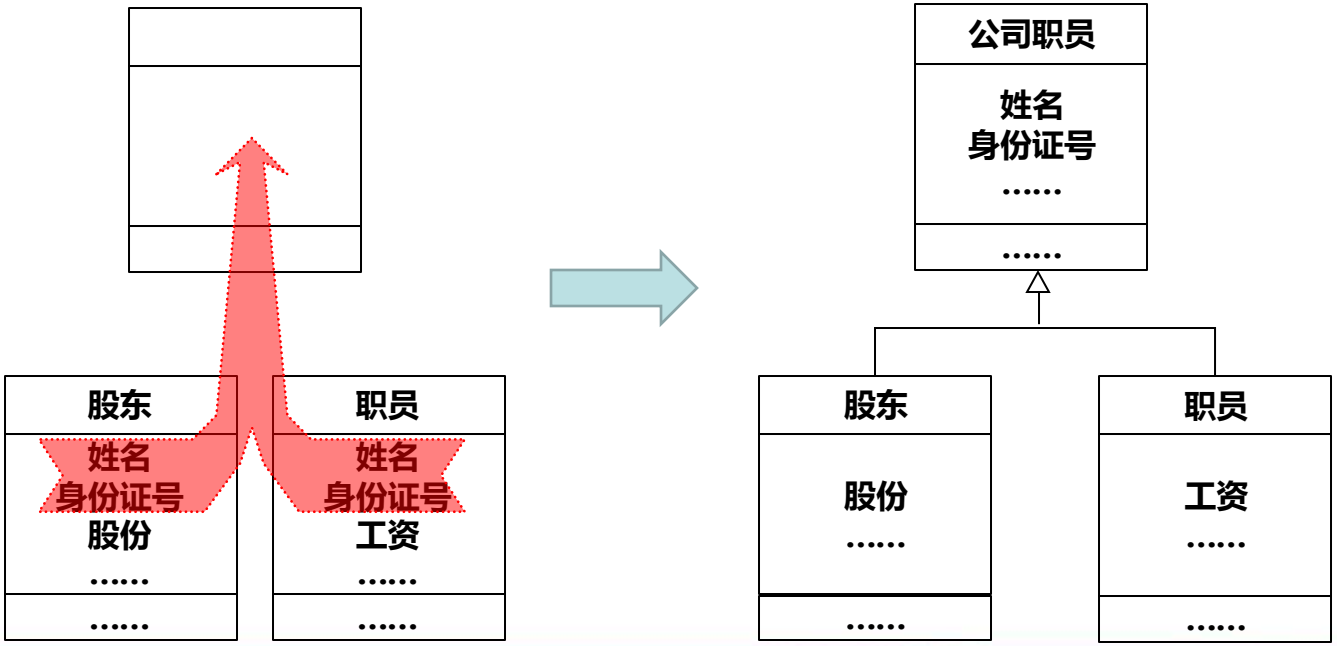
- **（1）识别泛化（继承）关系的策略**
  - ④考察属性与操作的适用范围

    对系统中的每个类，从以下两方面考虑它们的属性与操作：
    - 看一个类的属性与操作是否适合这个类的全部对象。如果某些属性或操作只适合该类的一部分对象，说明应从这个类中划分出一些特殊类，建立继承关系。



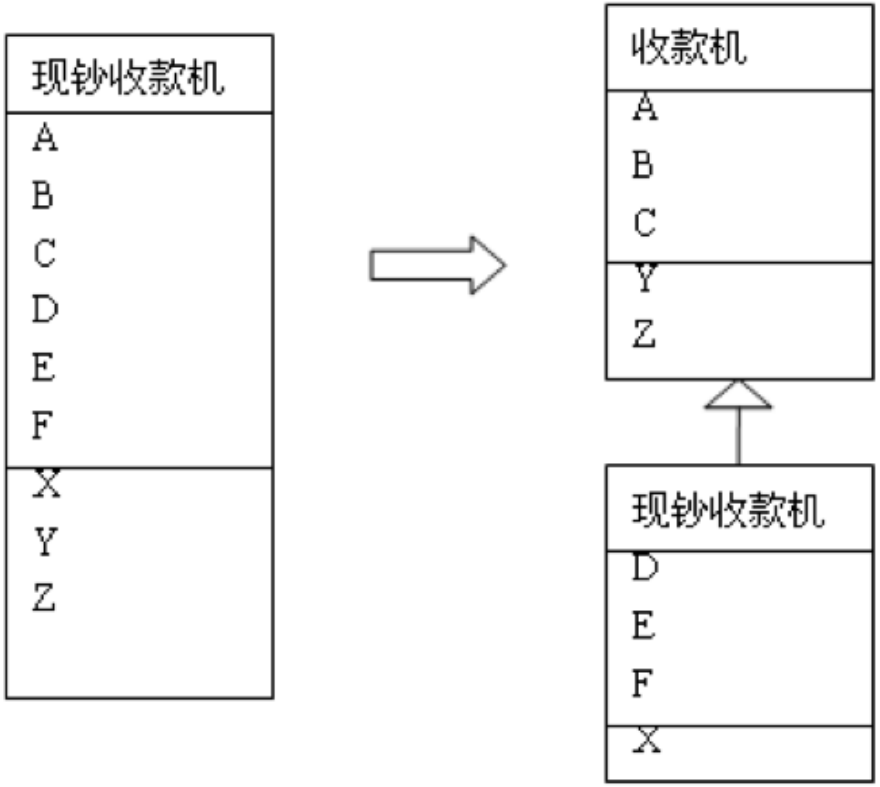
建立泛化（继承）关系

- **（1）识别泛化（继承）关系的策略**
  - ④考察属性与操作的适用范围
    - 检查是否有两个（或更多）的类含有一些共同的属性与操作。如果有则考虑，若把这些共同的属性与操作提取出来，能否构成一个在概念上包含原先那些类的一般类，形成一个继承关系。



建立泛化（继承）关系

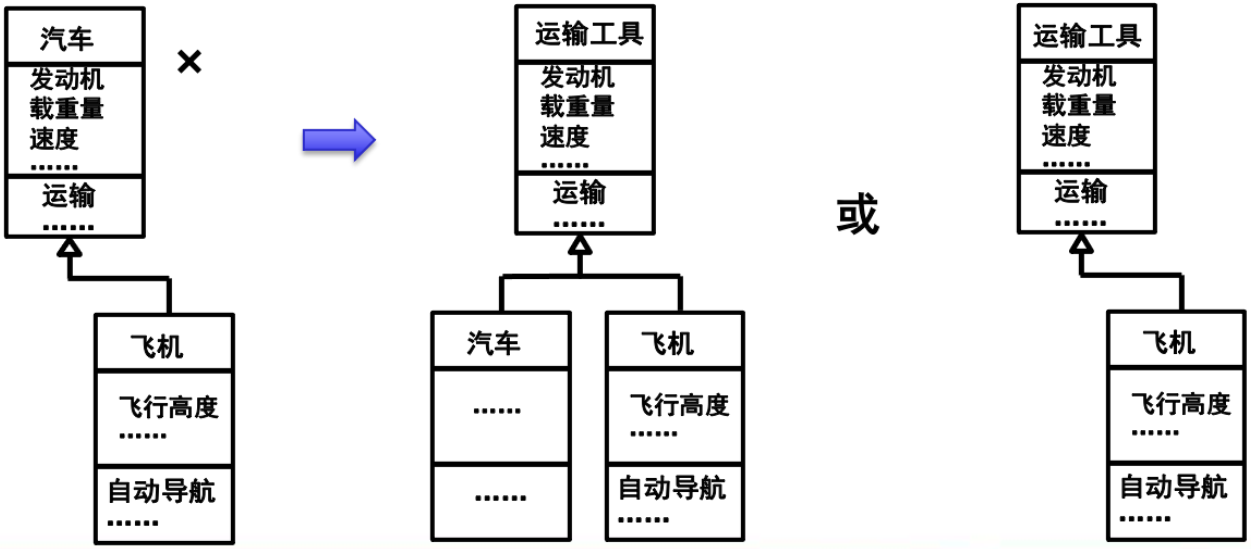
- **（1）识别泛化（继承）关系的策略**
  - ⑤考虑领域范围内的复用



建立泛化（继承）关系

- **（2）审查与调整**
  - ①问题域是否需要这样的分类？（例：书—线装书）
  - ②系统责任是否需要这样的分类？（例：职员—本市职员）
  - ③是否符合分类学的常识？（用“is a kind of”去套）
  - ④是否构成了继承关系？（确实继承了一些属性或操作）

例如：航标船



建立泛化（继承）关系

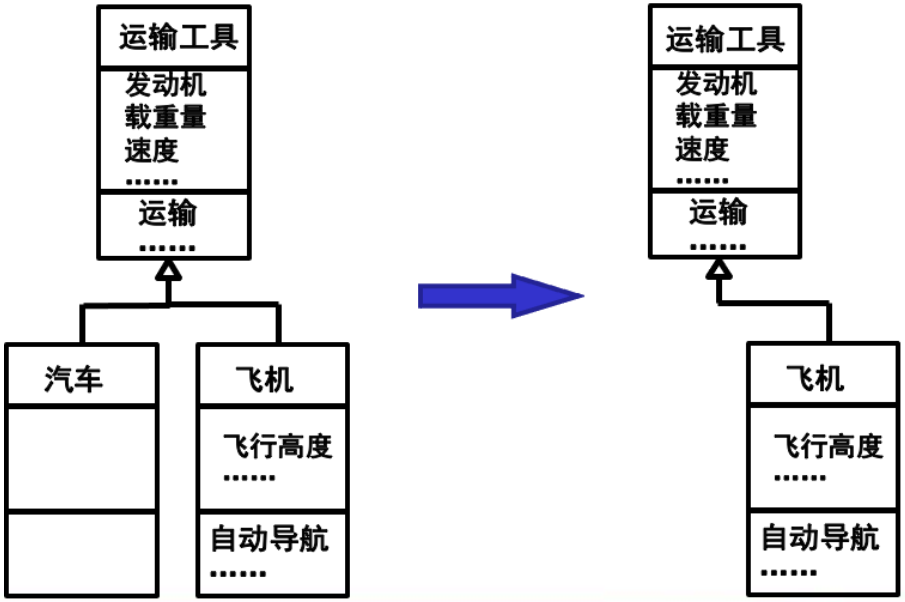
- **（3）继承关系的简化**

  从一般类划分出太多的特殊类，使系统中类的设置太多，增加了系统的复杂性；

  建立过深的继承层次，增加了系统的理解难度和处理开销

  对继承关系的运用要适度

  – ①取消没有特殊性的特殊类



建立泛化（继承）关系

- （3）继承关系的简化
  - ②增加属性简化继承关系（某些特殊类之间的差别可以由一般类的某个属性值来体现，而且除此之外没有太多的不同）

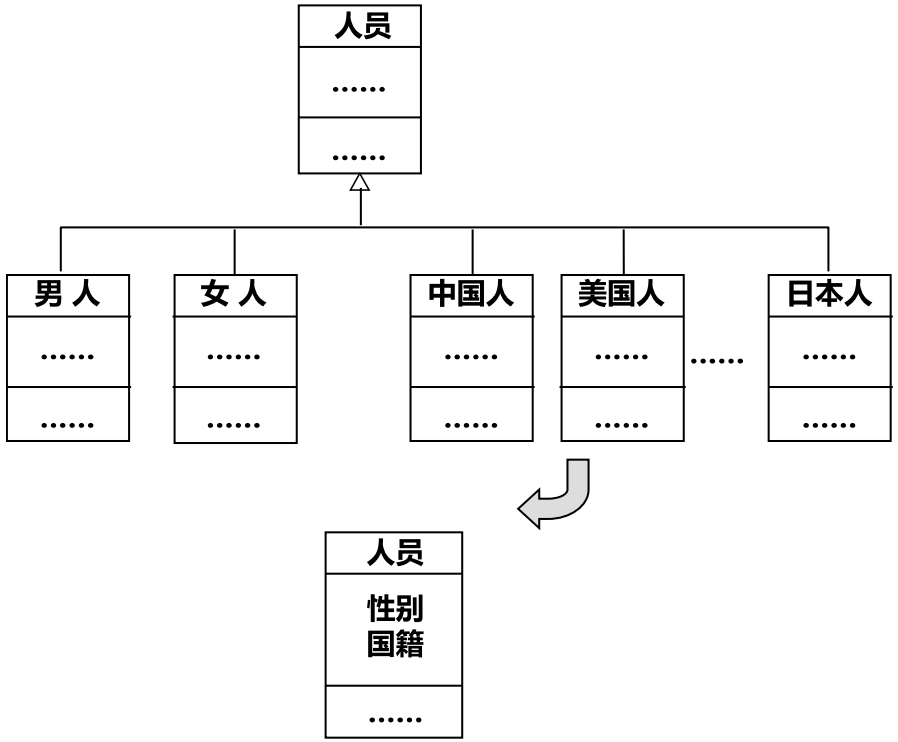


图 通过增加属性简化继承关系

建立泛化（继承）关系

- **（3）继承关系的简化**
  - ③取消用途单一的一般类，减少继承层次

    一般类存在的理由：
    - 有两个或两个以上的特殊类
    - 需要用它创建对象实例
    - 有助于软件复用

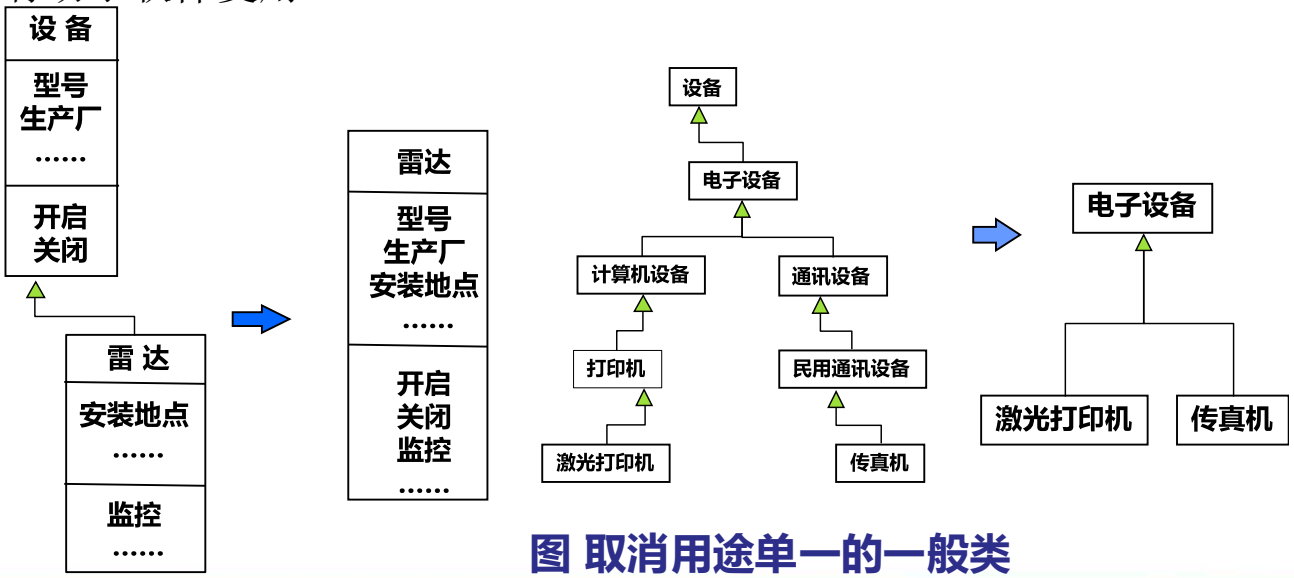


图 取消用途单一的一般类

建立泛化（继承）关系

- **（4）调整对象层和特征层**

  定义继承的活动，将使分析员对系统中的对象和类及其特征有更深入的认识，在很多情况下，随着继承的逐步建立，需要对类图的对象层和特征层进行某些修改，包括增加、删除、合并或分开某些类，以及增、删某些属性与操作或把它们移到其他类。

建立关联关系

- **（1）识别关联的策略**
  - ①从问题域角度识别类之间的静态联系

    判定这些静态联系是否提供了一些与系统责任有关的信息，如果是，则将这些静态联系标注为类之间的关联。
  - ②判断已识别出的关联是否是聚合关系。可以从以下几个角度进行判断：

    （a）判断具有关联关系的两个类是否物理上存在整体和部分的关系，如果是，则标注为聚合关系。例如汽车和轮胎，人和大脑。

    （b）判断具有关联关系的两个类是否存在组织机构和下级部门的关系，如果是，则标注为聚合关系。例如大学和院系。

    （c）判断具有关联关系的两个类是否存在团体和成员的关系，如果是，则标注为聚合关系。例如计算机学会与会员、学校和老师。

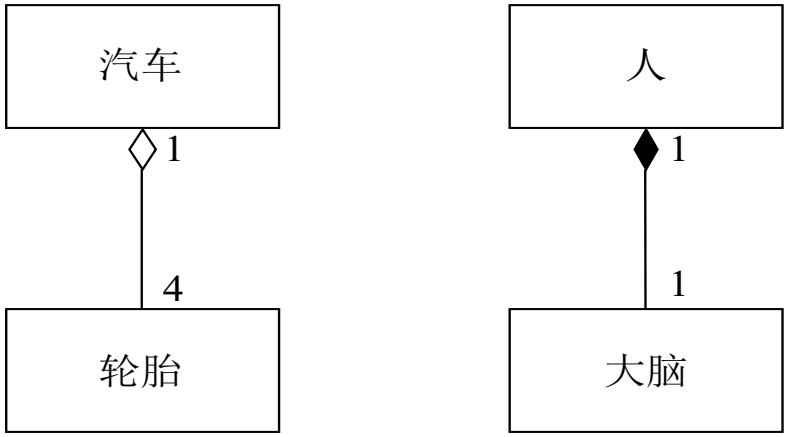
    （d）判断具有关联关系的两个类是否存在空间上的包容关系，如果是，则标注为聚合关系。例如，工厂和设备、教室和讲台。

    （e）判断具有关联关系的两个类是否是抽象事物的整体与部分关系，如果是，则标注为聚合关系。例如，法律与法律条文、项目管理计划和进度计划。

    （f）判断具有关联关系的两个类是否具有材料上的组成关系，如果是，则标注为聚合关系。例如房屋由混凝土、钢筋组成。

建立聚合关系

如果判断出关联是聚合关系，还需要判断该聚合关系的整体对象和部分对象的生命周期是否相同，如果相同，则该聚合关系是组合关系。需将指向整体对象的菱形画为实心菱形。



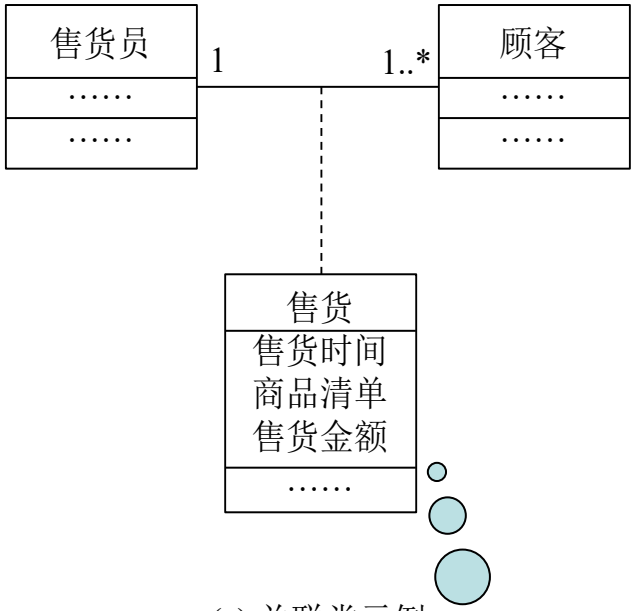
(a) 普通的聚合　　(b) 组合

建立关联关系

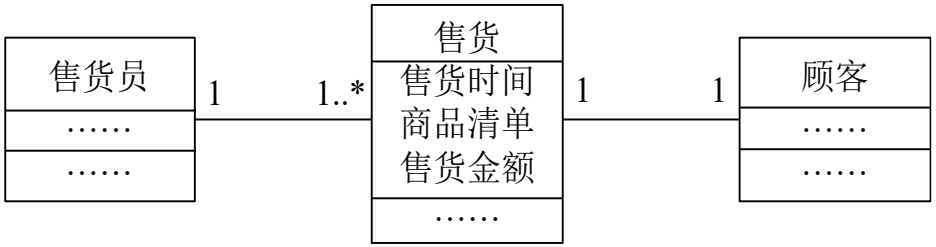
- **（1）识别关联的策略**
  - ③认识关联的属性与操作
  - 对于考虑中的每一种关联，进一步分析它是否应该带有某些属性和操作。就是说，是否含有一些仅凭一个简单的关联不能充分表达的信息。
  - 例：

    在售货员和顾客的“售货”的关联中，是否需要给出售货时间、商品清单、售货金额等属性信息？
  - 如果有，则可先在关联线上附加一个关联类符号来容纳这些属性和操作，或在两个类之间插入一个类来描述这些属性与操作。

建立关联关系



(a) 关联类示例



(b) 把关联类表示为普通类示例

有某些信息需要描述（如售货时间、商品清单、售货金额）

把带有属性和操作的关联表示为关联类（如售货）

建立关联关系

- **（1）识别关联的策略**
  - ④分析并表示关联的多重性和关联的性质

    对于每个关联，从连接线的每一端看本端的一个对象与另一端的几个对象发生连接，把结果标注在连接线的另一端。

    若需要，使用关联角色和限定符，以详细描述关联的性质。

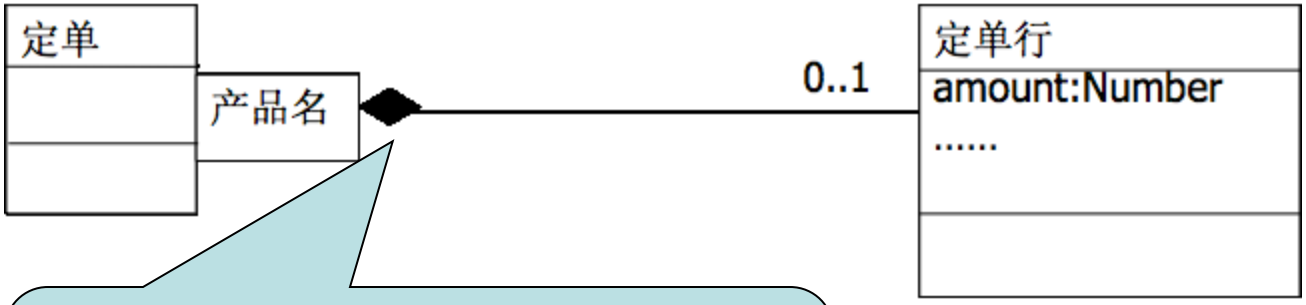
限定符的值用于确定该关联的另一端类的对象。即给定类的一个对象,并指定限定符内的属性值，能选定另一端类的一个对象或一组对象。

建立关联关系

- **（1）识别关联的策略**
  - 限定符举例：

    通常产品定单由定单行和一些其他描述信息组成。上图使用带有限定符的组合描述定单、定单行以及它们之间的关系。

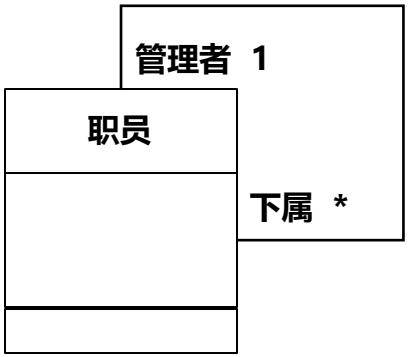
    对于一份定单，并指定了产品名，在另一端可能有或没有一个定单与其对应。如果没有这个限定符，给定一份定单，对应的定单行可能有许多。

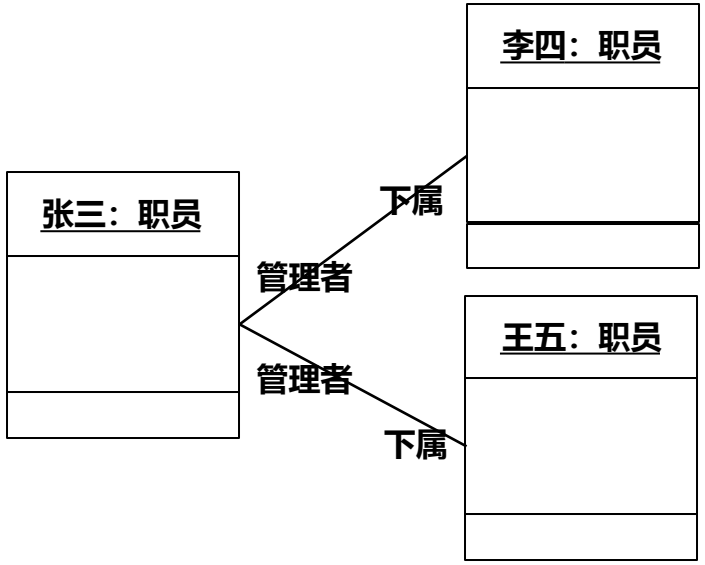


把限定符画成一个与关联的一端和该端的类图符相连的小矩形,并把受限的类（如“定单行”） 的属性放在小矩形中

建立关联关系

- **（1）识别关联的策略**
  - 关联角色：一个类参与一个关联的角色标识。





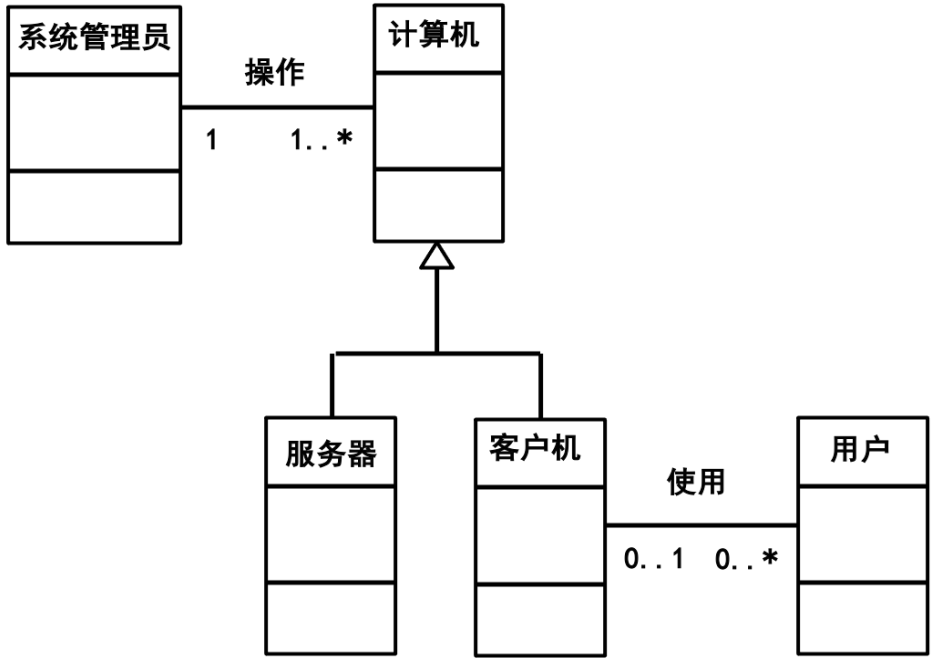
建立关联关系

- **（2）命名与定位命名：**

  当关联线的语义很清晰时，则关联的命名可缺省。否则，关联可用动词或动宾结构命名。

  – 定位问题：当连接线的某一端是一个继承结构时，要考虑连接线画到结构中的哪个类符号上。

  – 原则：如果这个关联适应结构中的每个类的对象，则画到一般类上，如果只适应其中某些特殊类，则画到相应的特殊类上。



建立关联关系

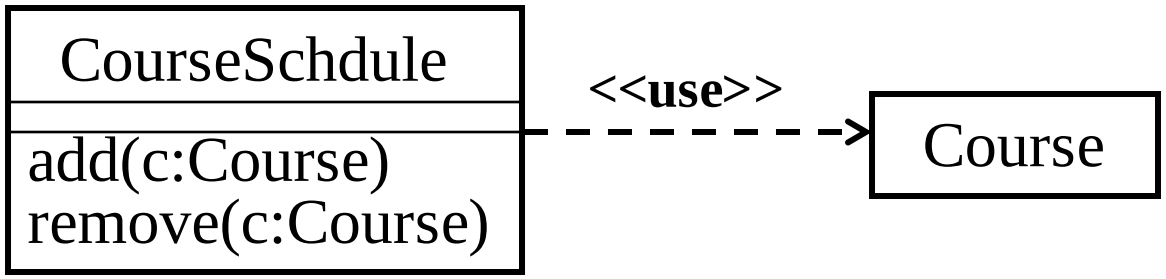
- **（3）对象层、特征层的修改以及关联的说明**
  - 在建立关联的过程中可能增加一些新的类，要把这些新增加的类补充到类图的对象层中，并建立它们的类规约。
  - 对于每一个关联，要给出其有关性质的说明，至少要说明它所表示的实际意义。

建立依赖关系

依赖是一种使用关系，用于描述一个类目（如类**Window**）使用另一类目（如类**Event**）的信息和服务。

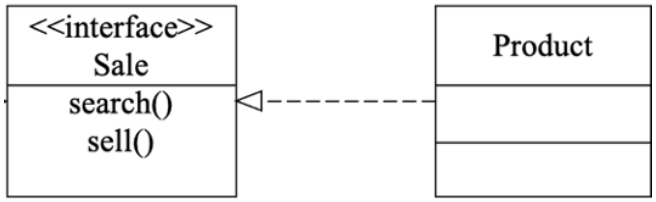
- 在大多数情况里，使用依赖来描述一个类使用另一个的操作；
- 如果被使用的类发生变化，那么另一个类的操作也会受到影响；

建议：在初步建立类之间的关系时，可以暂时使用依赖。在最终的类图中，若能用其他关系明确地指明类之间关系的含义，就不要使用依赖。

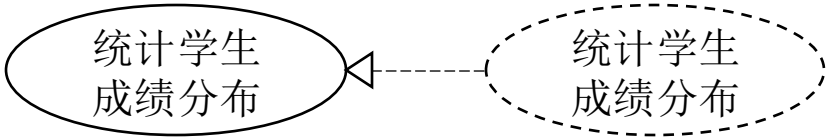


建立实现关系

在实际应用中，一般在2个地方会使用实现关系：
①接口与实现它们的类和构件之间；
②用况与实现它们的协作之间。
所以在类图中如果已有接口和实现它们的类，可以建立实现关系。



Sale接口与实现它的Product类之间建立实现关系



统计学生成绩分布用况与实现它的统计学生成绩分布协作之间建立实现关系

OOA--类图构建

1. 发现对象，定义类
2. 定义对象的属性
3. 定义对象的操作
4. 建立对象之间的关系
5. 创建类的模型规约

创建类的模型规约

类图中每个类以及类之间关系应进行详细描述。
面向对象分析设计文档对类的各个方面的描述进行了规约，如下所示：

2.2.2 类图文档

给出面向对象分析后得到的类图，并对类图进行文字描述。
文字描述由以下部分组成：类图综述、类描述、关联描述、泛化描述、依赖描述和其他与类图有关的说明。

2.2.1 类图综述

从总体上阐述整个类图的目的、结构、功能及组织。

2.2.2 类描述

对类图中的每一个类进行详细描述，包括类的整体说明、属性说明、操作说明、关联说明、泛化说明、依赖说明及其他说明。可以为每个类单独设置一节，具体内容可参照下面的格式。

创建类的模型规约

2.2.2.1 类1

a）类的整体说明

对整个类及其对象的情况加以说明，内容包括：

1）类名：应是中文名或英文名；

2）解释：对类的责任的文字描述；

3）一般类：描述该类是从哪些类泛化而来的；

4）主动性：有无主动性；

5）引用情况：若此类为其他类图所定义，则要标明它所属于的类图；若此类被其他类图引用，则表明所引用的类图；

6）其他：是否有特别的数据完整性或安全性要求等。

b）属性说明

逐个地说明类的属性。每个属性的详细说明包括以下内容：

1）属性名：中文属性名或英文属性名；

2）多重性：该属性的多重性；

3）解释：该属性的作用；

4）数据类型；

5）聚合关系：如果这个属性的作用是为了表明聚合关系，则在这里说明这种关系；

6）组合关系：如果这个属性的作用是为了表明组合关系，则在这里说明这种关系；

7）关联关系：如果这个属性是为了实现该类的对象和其他对象之间的链而设置的，则在这里明确地说明这一点；

属性名	多重性	解释	数据类型	聚合关系	组合关系	关联关系

创建类的模型规约

c）操作说明
逐个地说明类中的每个操作。每个操作的详细说明包括以下内容：

1）操作名：中文操作名或英文操作名；
2）主动性：有无主动性；
3）多态性：有无多态性；
4）解释：该操作的作用；
5）约束条件及其他：若该服务的执行有前置条件、后置条件或执行时间的要求等其他需要说明的事项，则再次说明。

操作名	主动性	多态性	解释	约束条件及其他

d）关联
描述该类所涉及的所有的关联。每个与该类相关的关联可有关联名
e）泛化
描述该类所涉及的所有的泛化。每个与该类相关的泛化可有泛化名
f）依赖
描述该类所涉及的所有的依赖。每个与该类相关的依赖可有依赖名

2.2.2.2 类2
2.2.2.3 类3
……

创建类的模型规约

2.2.3 关联描述

类图中的每一个关联都有如下的描述：

a）关联名称：中文关联名或英文关联名；

b）关联的类型：一般二元关联，聚合，组合，多元关联等；

c）关联所连接的类：按照一定顺序列举出关联所连接的类；

d）关联端点：对每一个关联端点的描述如下：

1) 导航性：是否有导航性；

2) 排序：是否排序；

3) 聚合：是否有聚合，如果有，则要指明是聚合还是组合；

4) 多重性；

关联名称	关联类型	关联所连接的类	关联端点			
			导航性	排序	聚合	多重性

2.2.4 泛化描述

类图中的每一个泛化都有如下的描述：

a）泛化名称；

b）泛化关系中的父类；

c） 泛化关系中的子类；

泛化名称	父类	子类

创建类的模型规约

2.2.5 依赖描述

类图中的每一个依赖都有如下的描述：

a）依赖名称；

b）依赖所涉及的类的名称；

c）依赖的类型：依赖一般是<<use>>类型，如果想表示使用关系之外的依赖类型，可参照课程中介绍的其他类型，标出所表达的依赖类型。

依赖名称	依赖涉及的类名称	依赖类型

案例分析：超级市场销售管理系统（续）

- **超级市场业务管理系统的子系统，只负责前台的销售管理**
- **功能需求：**
  - 为顾客选购的商品计价、收费、打印清单。
  - 记录每一种商品的编号、单价及现有数量。
  - 帮助供货员发现哪些商品将要脱销，以及时补充货源。
  - 随时按上级系统的要求报告当前的款货数量、增减商品种类或修改商品定价。
  - 交接班时结算货款数目，报告上级系统。
- **前面已发现的对象：**
  - 收款机、供货员、上级系统接口、商品、特价商品、计量商品、商品一览表、销售事件、帐册

案例分析：定义对象类之间的关系

- **1、继承：**
  - 一般类“商品”和它的两个特殊类“特价商品”及“计价商品”构成继承关系，并且特殊类中的某些属性和操作是多态的。
- **2、聚合：**
  - “商品一览表”和“商品”构成聚合关系，通过前者的“商品目录”属性体现这种关系；
  - “账册”和“销售事件”构成聚合关系，通过前者的“销售事件表”属性体现这种关系。
- **3、关联：**
  - 这个例子中没有其他关联关系。

案例分析：定义对象类之间的关系

- **4、依赖：**

  依赖是一种使用关系，说明一个事物（如类windows)使用另一个事物（如类event)的信息和服务。

  注：由于UML在类图上不能体现对象在行为上的关系，如果开发者想在类图上体现对象行为上的关系，就只能借用“依赖”这个概念及其表示法来弥补消息的空缺。

  如果一类对象的行为依赖另一类对象的行为，即一类对象的操作在执行时需要依靠和使用另一类对象的操作所提供的功能。

  ——《面向对象的系统分析（第2版）》，邵维忠，杨芙清著，北京：清华大学出版社，2006.12.)

  从收款员、供货员和上级系统这三类活动者的相关对象开始执行路线追踪，以发现对象行为之间的依赖关系：

  – （1）供货员
    - “供货员”对象在执行“供货”操作时，向“商品”对象发消息，请求其“补充”服务。

案例分析：定义对象类之间的关系

- （2）“收款机”
  - “收款机”对象在执行“售货”操作时向“商品一览表”对象发消息，请求其“检索”服务以找到相应的“商品”对象；
  - “收款机”对象向“商品”对象发消息以获知该种商品的属性信息并请求其“售出”服务；
  - 若“商品”对象发现该种商品数量低于规定的下限，则向“供货员”对象发消息，请求其“缺货登记”服务；
  - “收款机”对象向“销售事件”对象发消息，请求“销售计价”和“入账”服务；
  - 在执行“入账”操作时，“销售事件”对象向“账册”对象发消息，以请求其“记账”服务；
  - “收款机”对象在执行自己的“登陆”操作时，向“账册”对象发消息，请求“接班”服务；
  - “收款机”对象在执行“结账”操作时，也要向“账册”对象发消息，请求其“报账”和“交班”服务；
  - “账册”对象向“上级系统接口”对象发消息，请求其“报账”服务。

案例分析：定义对象类之间的关系

- （3）上级系统接口
  - “上级系统接口”对象在上级系统要求查账时向“账册”对象”发消息，请求它的”报账“服务；
  - 在上级系统要求进行价格更新或商品种类增删时，分别向“商品”和“商品一览表”对象发消息，请求相应的服务。

例如：超级市场销售管理系统（关系层、完整的类图）