南京大学
NANJING UNIVERSITY

面向对象分析方法

张天
软件工程组
ztluck@nju.edu.cn
2025年秋季

案例：

- 需要开发一款手机银行APP，使得用户通过该软件，可以对自己的银行帐户进行操作。
- 使用面向对象分析方法对该系统进行分析
  - 用户场景
    - 访问控制、转账、代缴费、查询（余额、账单）、理财推荐、公益捐赠
    - 申请贷款、信用卡还款、发红包、挂失、扫二维码、预约、绑定收集、短信通知

内容组织

- 面向对象概念
- 面向对象分析

回顾面向对象程序设计过程

- 定义类
- 系统
- ？？？

面向对象的基本概念

- “面向对象”是一种认识客观世界的世界观，这种世界观将客观世界看成是有许多不同种类的对象构成的，每个对象有自己的内部状态和运动规律，不同对象之间的相互联系、相互作用就构成了完整的客观世界。
- “面向对象”是从结构组织的角度去模拟客观世界的一种方法，这种方法的基本着眼点是构成客观世界的那些成分----对象。
- 用“面向对象”的观点去认识客观世界，用“面向对象”的方法去模拟客观世界，这就构成了“面向对象”的完整含义。

面向对象的基本概念

- 抽象 （Abstract）
- 对象（Object）
- 类（class）
- 封装（encapsulation）
- 继承（inheritance）
- 多态性（polymorphism）
- 信息/实现的隐藏（information/implementation hiding）
- 状态保持（state retention）
- 对象标识（object identity）
- 消息（message）
- 一般性（generality）

面向对象的基本概念

- 抽象：从许多事物中舍弃个别的、非本质的特征，抽取共同的、本质性的特征。
- 抽象是形成概念的必须手段。第一，关注本质：尽管问题域中的事物是很复杂的，但是分析员并不需要了解和描述它们的一切，只需要分析研究其中与系统目标有关的事物及其本质性特征。第二，通过舍弃个体事物在细节上的差异，抽取其共同特征而得到一批事物的抽象概念。
- 抽象原则包括过程抽象和数据抽象两个方面。
  - 过程抽象是指，任何一个完成确定功能的操作序列，其使用者都可以把它看作一个单一的实体，尽管实际上它可能是由一系列更低级的操作完成的。
  - 数据抽象是根据施加于数据之上的操作来定义数据类型，并限定数据的值只能由这些操作来修改和观察。数据抽象是OOA的核心原则。它强调把数据（属性）和操作（服务）结合为一个不可分的系统单位（即对象），对象的外部只需要知道它做什么，而不必知道它如何做。

面向对象的基本概念

- 对象(Object)
  - 在现实世界中是客观世界中个体或事物的抽象表示，在面向对象程序中表达成计算机可理解、可操纵、具有一定属性和行为的对象；在计算机世界中是一个可标识的存储区域
  - 属性表示对象的性质，属性值规定了对象所有可能的状态。
  - 操作是指该对象可以展现的外部服务。

面向对象的基本概念

- 类(Class)
  - 类是某些对象的共同特性的表示，它描述了这些对象内部是如何构造的。相同类的对象在它们的操作和它们的信息结构两个方面都有相同的定义。
    - 类是一组具有相同数据结构和相同操作的对象的集合
    - 类的定义包括一组数据属性和在数据上的一组合法操作。
- 分类：就是把具有相同属性和服务的对象划分为一类，用类作为这些对象的抽象描述。分类原则实际上是抽象原则运用于对象描述时的一种表现形式。

面向对象的基本概念

- 类与对象
  - 类与对象之间的关系，就是类与其对应实例之间的关系。
  - 类定义可以视为一个具有类似特性与共同行为的对象的模板，可用来产生对象。
  - 类是对象的类型（模版），对象是类的实例。

在面向对象系统中，每个对象都属于一个类。属于某个特定类的对象称为该类的实例。因此，常常把对象和实例当作同义词。实例是从某类创建的一个对象。

面向对象的基本概念

- 继承：子类隐式使用超类（或父类）的属性和操作。
  - 如果类B继承类A，那么类A中描述的操作和信息结构将成为类B的一部分。
  - 借助继承，可以表示类之间的类似性，并且在其他类能继承的一个类中描述这些相似性。因此，就能够复用公共的描述。继承常常被提倡为软件工业界中关于复用的一个核心思想。继承还有利于软件维护。
  - 通过抽取和共享公共特性就能够通用化一些类，并且把它们放在继承层次的更高位置。同样，如果希望增加新类，可以寻找这样一个类，它已经提供了适用于该新类的某些操作和信息结构。然后，让新类继承这个类，只需增加该新类所独有的那些内容。然后，使这个类专用化。

面向对象的基本概念

- 封装：把对象的属性和服务结合为一个不可分的系统单位，并尽可能隐蔽对象的内部细节。
  - 隐藏对象实现细节称为封装。
  - 类包括接口和实现两个部分。接口可以被其他对象看到和使用。实现对使用者隐藏的。
  - 封装为对象提供两种保护：
    - 对象内部状态不被使用者打断。
    - 使用者代码不能修改对象实现
  - 把对象的全部属性和全部服务结合在一起，形成一个不可分割的独立单位（对象）。
  - 尽可能隐蔽对象的内部细节

面向对象的基本概念

- 多态：**隐藏一个接口不同实现的能力。**
  - 使用者可以激活一个对象的操作，而且事先不知道其类型。
  - 多态允许使用者通过超类去操纵对象
- 子类覆盖（overriding）父类的方法，它和重载（overloading）的区别在于重载是在同一个类中定义，利用参数的不同来进行动态绑定（dynamic binding）。

面向对象的基本概念

- 信息/实现的隐藏，将某些属性或方法限制在封装内部使用，限制外部的可见性。
- 状态保持，对象能够保持状态，可以用于后续的处理。
- 对象标识，每个对象可以作为软件实体被标识和处理，每个对象都有一个对象标识符（object identifier OID）。
- 消息，对象间发送请求的载体。
- 一般性，类的定义是参数化的或模版化的，提高了定义的通用性。

面向对象的基本概念

- 聚合：又称组装，其原则是：把一个复杂的事物看成若干比较简单的事物的组装体，从而简化对复杂事物的描述。
- 关联：是人类思考问题时经常运用的思想方法：通过一个事物联想到另外的事物。能使人发生联想的原因是事物之间确实存在着某些联系。
- 消息通信：这一原则要求对象之间只能通过消息进行通信，而不允许在对象之外直接地存取对象内部的属性。通过消息进行通信是由于封装原则而引起的。在OOA中要求用消息连接表示出对象之间的动态联系。
- 粒度控制：一般来讲，人在面对一个复杂的问题域时，不可能在同一时刻既能纵观全局，又能洞察秋毫。因此需要控制自己的视野：考虑全局时，注意其大的组成部分，暂时不详察每一部分的具体的细节；考虑某部分的细节时则暂时撇开其余的部分。这就是粒度控制原则。
- 行为分析：现实世界中事物的行为是复杂的。由大量的事物所构成的问题域中各种行为往往相互依赖、相互交织。

面向对象分析

- 面向对象分析方法（Object-Oriented Analysis，OOA），是在一个系统的开发过程中进行了系统业务调查以后，按照面向对象的思想来分析问题、表达问题，表达分析的结果。
- 分析过程是在软件工程的环境中建立基本系统行为的过程，方法是要构造待开发软件系统的形式模型，捕捉系统最基本的需求, 重点是模型建立的机制。
- OOA与SA有较大的区别：
  - SA需要对系统业务现状和方法的分析；
  - OOA所强调的是在系统调查资料的基础上，针对OO方法所需要的素材进行的归类分析和整理；

分析方法回顾

- 功能分解：功能＝功能＋｛ 子功能｝＋功能接口
  - 直接地反映用户的需求， 容易入手
  - 不能直接地映射问题域， 难以检验
  - 不适应需求频繁变化
  - 局部的错误和修改会引起系统波动
- 结构化分析方法（数据流方法）：数据流法＝ 数据流＋数据处理（加工）＋数据存储＋端 点＋处理说明＋数据字典
  - 数据流和加工的数量太多，引起分析文档的膨胀
- 信息建模法：系统＝实体＋属性＋关系
- 区别：认识问题的基础不同，构造系统的基础和工具不同
- 共性：系统对现实世界（问题域）的不同映射

面向对象分析的任务

- 基本任务：运用面向对象方法，对问题域和系统责任进行分析和理解，找出描述问题域及系统责任所需的对象， 定义对象的属性、操作以及它们之间的关系。
- 目标：建立一个符合问题域、满足用户需求的需求分析规格说明（OOA模型）
- 面向对象分析是抽取和整理用户需求并建立问题域精确模型的过程。识别问题域的对象并分析它们相互之间的关系，最终建立简洁、精确、可理解的正确模型是分析阶段的关键。
- 开发人员首先要理解用户的需求，找出描述问题域和系统责任所需的对象和类，将用例行为映射到对象上，进一步分析它们的内部构成和外部关系，从而建立面向对象分析模型。在此基础上，开发人员和用户一起检查模型，保证模型的正确、一致、完整和可行。
- 面向对象的分析过程是一个循环渐进过程，需要多次循环迭代完成。

面向对象分析的优点

- 加强了对问题域和系统责任的理解；
- 改进与分析有关的各类人员之间的交流；
- 对需求的变化具有较强的适应性；
- 支持软件复用的要求；
- 贯穿软件生命周期全过程的一致性。
- 考虑实用性；
- 有利于用户参与。

OOA方法

- Booch方法
- Rumbaugh方法(OMT)
- Jacobson方法(OOSE)
- 。。。。
- ＝》基于UML的OOA方法

面向对象分析过程

- 标识对象和类
  - 对象是对数据及其处理方式的抽象，它反映了系统保存和处理现实世界中某些事物的信息的能力。类是多个对象的共同属性和方法集合的描述，它包括如何在一个类中建立一个新对象的描述。
- 标识结构
  - 结构是指问题域的复杂性和连接关系。类成员结构反映了泛化-特化关系，整体-部分结构反映整体和局部之间的关系。
- 定义主题
  - 主题是指事物的总体概貌和总体分析模型。
- 定义属性
  - 属性就是数据元素，可用来描述对象或分类结构的实例，可在图中给出，并在对象的存储中指定。
- 定义服务
  - 方法是在收到消息后必须进行的一些处理方法：方法要在图中定义，并在对象的存储中指定。对于每个对象和结构来说，那些用来增加、修改、删除和选择一个方法本身都是隐含的（虽然它们是要在对象的存储中定义的，但并不在图上给出），而有些则是显示的。

OOA建模与表达

- OOA通过分析活动构造分析模型
  - 5层次模型
    - 对象－类层
    - 属性层
    - 服务层
    - 结构层
    - 主题层

对象—类层

- 对象—类层，表示待开发系统的基本构造块。这一层是整个OOA模型的基础。问题在于如何建立“现实世界中事物”的抽象表示，也就是**如何建立基本块**
- 信息建模，就是指从现实世界中捕捉并抽象出应用论域的基本结构的过程。这是OOA过程中最基本和最关键的活动之一。
- 应用论域是非常重要的。同一概念在不同论域中抽象出来的基本构造块是不同的 。
- 图符
- 模板类或抽象类
- 属性和服务
- 实例连接：应用论域的某些限制条件或事务规则。例如，定金取消后，相应的订户也应该被取消。

属性层和服务层

- 属性层
  - 对象的属性和实例连接共同组成了OOA模型的属性层。
  - OOA在定义属性的同时，要识别实例连接。实例连接是一个实例与另一个实例的映射关系。
- 服务层
  - OOA在定义服务的同时要识别消息连接。当一个对象需要向另一对象发送消息时，它们之间就存在消息连接。
  - 对象的服务，加上对象实例之间的消息通信，共同组成了OOA模型的**服务层**。
  - 对象实例都分别执行一定的工作或功能，相互之间也通信，即所谓的协同。消息连接用有向箭头表示。

结构层

- 该层负责捕捉特定应用论域中的结构关系。
- 结构层描绘出了系统的整体结构。
- 结构层的另一种类型称为一般—特殊结构或泛化—特化结构（泛化—特化结构表明了类的继承性 。
- 在这种方法中定义了两种对象类之间的结构，一种称为分类结构，一种称为组装结构。分类结构就是所谓的一般与特殊的关系。组装结构则反映了对象之间的整体与部分的关系。

主题层

- OOA模型的结构庞大而复杂，可以将对象归类到各个主题中，把有关的对象用一个主题边框框起来。

OOA模型

- 功能（用例）模型：描述系统整体视图，表达系统的详细需求，由用例图和场景描述组成
- 对象（结构）模型:表示静态的、结构化的系统“数据”性质，对用例模型进行分析,把系统分解成互相协作的分析类,通过类图/对象图描述对象/对象的属性/对象间的关系,是系统的静态模型
- 动态（行为）模型:描述系统的动态行为,系统的动态结构和对象之间的交互，表示瞬时的、行为化的系统的“控制”特性。

- 谢谢！