

软件协同设计课程之

6.2 问题域设计



2025年1月



1 软件总体设计

- 设计方法

 - SA/SD

 - OOA/OOD

- 最富创造性地阶段：

 - 构造问题的解

 - 形成产品的结构 Architecture

- 总体设计/高层设计/概念设计

 - 系统由部件组成

 - 部件之间通信（参数传递/接口）

1 软件总体设计内容

- 硬件、OS、网络
- 软件体系结构：总体架构
- 问题域的设计：功能模块结构图/主题划分的类图
- 数据管理：（文件、数据库？）、类设计
- 软件运行驱动设计
- 用户界面、界面类的设计
- 编程语言
- 可复用组件的设计
- 质量与安全设计

2 传统的结构化设计



■ 结构化概要设计

- 对软件产品进行模块化分解，产生具有期望功能的模块结构
- 基于DFD图进行设计

■ 结构化详细设计

- 模块的数据结构与算法



3 面向对象的设计



- 问题域类图设计（概要设计）

- 对需求进行分析理出问题域 **的类**

- 基于对业务逻辑分析，设计整体类图

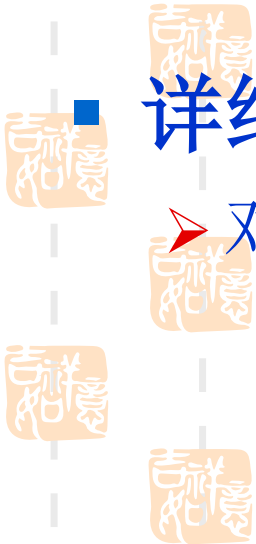
- 生产信息系统主题划分与类设计，见下页

- 详细设计

- 对类的设计进一步完善

- 类的属性

- 类的方法：数据结构与算法



4. 如何确定类



- ① 标识候选的对象
- ② 筛选对象，确定最终对象的类
- ③ 检查和调整
- ④ 标识类



4. 如何确定类

(1) 标识候选的对象

需求陈述中的**名词或名词短语**是可能的候选对象，它们以不同的形式展示出来。

□ 外部实体

如其他系统、设备、人员，它们生产或消费计算机系统所使用的信息

□ 物件、构造物

如成绩单、报表、报告、显示、信函、信号，它们是问题信息域的一部分

□ 发生的事情或事件

如性能改变或完成一组机器人移动动作，它们出现在系统运行的环境中

□ 角色

如管理者、工程师、销售员，它们由与系统交互的人扮演

□ 组织单位

如部门、小组、小队，它们与一个应用有关

□ 场所

如制造场所、装载码头，它们建立问题和系统所有功能的环境

□ 潜在的类

- 是否有存储、转换、分析与处理
- 是否有外部系统



4. 如何确定类

(2) 筛选对象，确定最终对象的类

可以用以下选择特征来确定最终的对象：

□ 保留的信息

仅当必须记住这些候选对象的信息，系统才能运作时，则该对象可定为候选对象

□ 需要的服务

候选对象必须拥有一组可标识的操作，它们可以按某种方式修改对象属性的值

□ 多个属性

- 在分析阶段，关注点应该是“较大的”信息。
- 仅具有单个属性的对象，最好把它表示为另一对象的属性

4. 如何确定类

(3) 检查和调整

□ 类的属性和操作不适合该类的全部对象

- 重新分类

□ 属性和操作相同的类

- 可能合并，要根据实际情况

□ 对同一事物的重复描述

- 学生证类与学生类



4. 如何确定类

(4) 标识类

这里介绍一种类—责任—协作者（class—responsibility—collaborator，简称CRC）技术。CRC实际上是一组表示类的索引卡片，每张卡片分成三部分，它们分别描述类名、类的责任和类的协作者

类名：

类的类型：（如：设备，角色，场所，……）

类的特征：（如：确切的事物，原子的，并发的，持久的……）

责任：

协作者：

CRC卡

4. 如何确定类

■ 类的责任：即类的操作，大体可分为三类

□ 以某种方式操纵数据的操作

➤ 如：增加、删除、重新格式化、选择等

□ 完成某种计算的操作

□ 为控制事件的发生而监控对象的操作



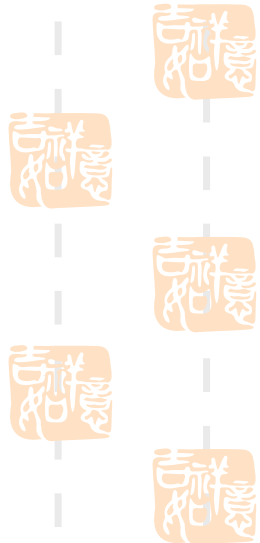
4. 如何确定类

■ 协作者

- 一个类也可和其他类协作来完成某个责任。

■ 标识协作者

- 两个类具有整体与部分关系
- 一个类必须从另一个类获取信息
- 一个类依赖于（depends-upon）另一个类，则它们间往往有协作关系




4. 如何确定类

(5) 复审CRC卡：基于用例

在填好所有CRC卡后，应对它进行复审。复审应由客户和软件分析员参加，复审方法如下：

- 1) 参加复审的人，每人拿CRC卡片的一个子集。注意，有协作关系的卡片要分开，即，没有一个人持有两张有协作关系的卡片
- 2) 将所有用例/场景分类
- 3) 复审负责人仔细阅读用例，当读到一个命名的对象时，将令牌（token）传送给持有对应类的卡片的人员

- 
- 4) 收到令牌的类卡片持有者要描述卡片上记录的责任，复审小组将确定该类的一个或多个责任是否满足用例的需求。

当某个责任需要协作时，将令牌传给协作者，并重复4)

- 5) 如果卡片上的责任和协作不能适应用例，则需对卡片进行修改，这可能导致定义新的类，或在现有的卡片上刻画新的或修正的责任及协作者

这种做法持续至所有的用例都完成为止

举例



- 项目介绍

- 简化的需求\需求分析0.doc

- 简化的需求\需求分析1.doc

- 简化的需求\需求分析2.doc

(介绍)

- 简化的需求\需求分析3.doc



■ 类之间的关系

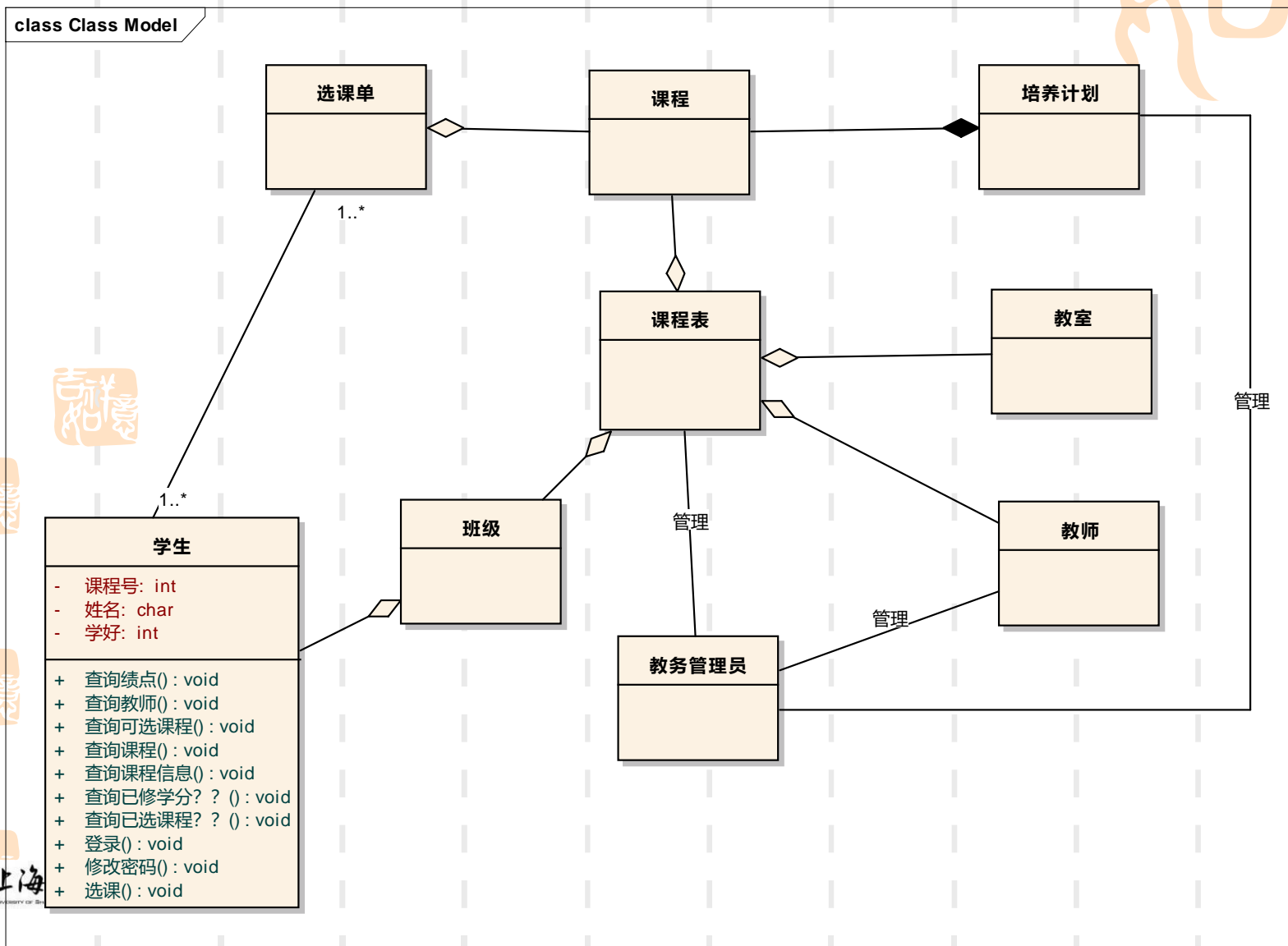
- 依赖关系 (using a, 一个类中的方法使用另一个类)
- 关联 association: 类之间的关系
 - 一个类的属性类型是另一个类
- 聚集和组合 (has a)
- 泛化关系 (is a)



类之间的关系图形

关 系	功能	符号
关联	类实例间连接的描述	—————
依赖	二个模型元素之间的一种关系	----->
泛化	更特殊描述与更一般描述之间的一种关系，用于继承和多态性类型声明	—————▷
实现	规约 (specification) 与它的实现之间的关系	-----▷

教务管理系统：选课管理类图



总结



- 面向对象设计：包括若干个侧面
- 问题域设计：基于业务与需求描述，构建问题域类图
- CRC卡片与类的审核

