**《软件工程》**

**实验报告六：面向对象的系统建模D**

**姓 名： 向申赤 学 号： 202210120510**

**院 系： 计算机与信息学院 专 业： 计算机科学与技术**

**实 验 室： J1-306 实验日期： 2024.11.18**

**总评成绩： 审阅教师： 杨青**

# 实验目的

1. 掌握 UML 建模工具 Rational Rose 软件的安装和基本操作；
2. 掌握面向对象设计模型，包括类和类间的关系建模
3. 能够采用B-C-E分析类，构建类图静态模型
4. 能够基于B-C-E分析类，使用顺序图（Sequence）构建行为模型，对用例的事件流建模

# 实验环境

Rational Rose Enterprise Edition, StarUML.

# 实验要求

1. 阅读下面材料，根据实验五中“医院预约挂号系统”编写的系统用例图，对预约挂号用例和支付挂号费用例，使用Rose完成以下任务：
2. 根据系统用例图([word版本参考](https://star.jmhui.com.cn/u/cms/www/202211/21072322srku.docx)）或[pdf版本参考](https://star.jmhui.com.cn/u/cms/www/202304/25103359u5a8.pdf)，绘制边界类、控制类和实体类三种分析类，并绘制B-C-E包图
3. 对预约挂号用例和支付挂号费用例中的控制类绘制相应的顺序图，转换为协作图；
4. 绘制“参与类类图”VOPC图。以一个Word文件的形式提交（Rose绘制的模型图贴到文档的适当位置）
5. 参考[《第5章 面向对象方法5（use Case Analysys）.ppt》](https://star.jmhui.com.cn/u/cms/www/202110/28175221uwxd.pptx)或[《第5章 面向对象方法5（use Case Analysys）.pdf](https://star.jmhui.com.cn/u/cms/www/202304/23081816t8hz.pdf)旅店预订案例
6. 文件以“学号-姓名-软件工程实验六.doc”的方式命名，提交到长江雨课堂“软件工程实验六”

# 实验内容

“医院预约挂号系统”问题陈述：

为了规范和推动医院预约挂号服务，卫生部2019年8月在其官方网站发布了《关于在公立医院施行预约诊疗服务工作的意见(征求意见稿)》，要求在推动医院开展预约挂号工作的同时，提高对预约挂号服务工作的认识、加强对预约挂号服务工作的管理、并认真做好相关组织工作。

某IT公司瞄准此次契机，决定着手开发一个通用的“医院预约挂号系统”，以满足各级公立医院的预约挂号需求。

系统的基本流程如下：未注册用户可以通过该系统查询医院、相关科室、各科室的医生等各类信息，但不能使用其它与预约相关的业务。需要进行预约挂号的用户必须通过该网站利用身份证号进行实名注册，并提供手机号，注册信息由系统管理员进行审核，审核通过后，用户才可使用手机号和验证码登录该系统。为了方便用户登录，系统还支持第三方（如微信公众号）登录。微信公众号登录必须首先绑定手机号用户才能登录。预约挂号时，用户首先选择需要预约的医院，之后选择要预约的科室和时间（指定某个日期的上午或下午）；此时，系统应自动显示该时间段内该科室所有出诊的医生。需要注意的是，每个医生每次出诊所能看病的人数有一定的限制，当某个医生的预约人数满员后即不可预约。用户可以选择一个可预约的医生进行预约，一个用户每个时间段最多只能预约5位医生。预约成功后，用户可以打印预约单。用户还可以通过第三方的支付系统（I期只支持淘宝的支付宝，后续支持各类信用卡）网上支付挂号费，也可以暂不交费。年龄超过60岁的患者免费。已交费的用户还可打印挂号单，并在看病当天拿着预约单和挂号单直接去医院相应的科室分诊台进行分诊，分诊台的护士核查预约单和挂号单无误后盖章确认，即允许用户看病。未交费的用户需要拿着预约单到医院的挂号处交费，挂号处核查预约单，并打印出挂号单，盖章确认后交给分诊台护士后进行分诊。

在看病的前一天，用户可随时取消预约记录，系统不收取任何费用，已缴的费用会自动退回到用户的帐号。看病当天的预约记录只能在医院挂号处现场取消，也不收取费用。但是，对于那些在网上预约成功，却不去看病也不按时取消的用户，系统会进行警告：已收取的费用不再退回，每出现一次用户的信用等级下降1级；当用户信用等级将为0时，不再允许使用该系统。用户的初始信用等级是在审核用户注册信息时设定的。

此外，有关医生的出诊信息可以由系统管理员手动维护，也可通过定制一些规则后由系统提前若干天（具体多少天可以由系统管理员设置）生成某日的出诊信息。当用户预约成功、缴费成功、退费成功、预约时间到达前2小时提醒、预约未使用未取消时系统均要求使用短信通知用户。

支持登录用户代患者挂号功能，以解决对于无法使用APP的用户如婴幼儿或老年人也能够使用该预约挂号系统。

1. 根据系统用例图([word版本参考](https://star.jmhui.com.cn/u/cms/www/202211/21072322srku.docx)）或[pdf版本参考](https://star.jmhui.com.cn/u/cms/www/202304/25103359u5a8.pdf)，绘制边界类、控制类和实体类三种分析类，并绘制B-C-E包图。

在面向对象分析中，边界类、控制类和实体类是用于建模系统的三种类型的类。边界类代表系统与外部世界之间的接口，例如用户界面、输入/输出设备和通信协议。控制类封装了管理系统行为的逻辑和规则，负责决策并协调其他类之间的交互。实体类则表示被系统操作的数据和对象，如客户、订单和产品。B-C-E包图是这些类的图形表示，其中边界类位于外围，控制类位于中间，而实体类位于核心，展示了它们如何相互作用与协作以实现系统目标。

图示

描述已自动生成

图示

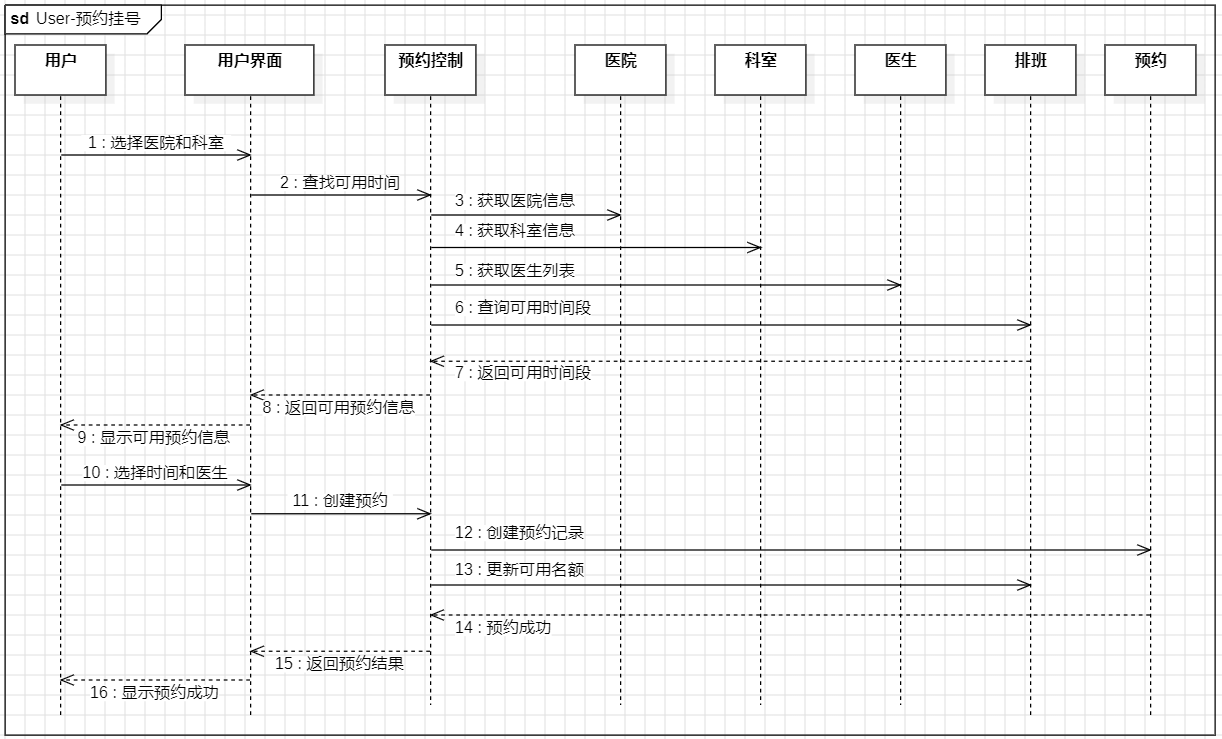
描述已自动生成

1. 对预约挂号用例和支付挂号费用例中的控制类绘制相应的顺序图，转换为协作图；

顺序图（Sequence Diagram）是一种用于描述对象之间交互的UML图表，主要关注消息的传递顺序和时间顺序。它通过垂直的生命线表示参与者（对象或角色），并通过水平的箭头表示它们之间的消息传递。顺序图能够清晰地展示系统在特定场景下的动态行为，帮助开发者理解对象之间的协作关系和交互流程，通常用于需求分析和系统设计阶段。

协作图（Collaboration Diagram），也称为通信图，是一种UML图表，用于描述对象之间的交互关系和消息传递。与顺序图不同，协作图强调对象之间的结构和关系，通过连接线表示对象之间的连接，并用标注的箭头指示消息的传递顺序。协作图能够直观展示系统中对象的协作方式，适用于分析和设计阶段，帮助开发者理解系统的功能和对象间的相互作用。

预约挂号：



图示

描述已自动生成

支付挂号费：

表格, 日历

描述已自动生成

图示

描述已自动生成

1. 绘制“参与类类图”VOPC图。

参与类类图（Collaboration Class Diagram）是一种UML图表，用于描述对象之间的协作关系和消息传递，强调对象之间的结构和交互方式。它展示了系统中类或对象的关系，帮助开发者理解系统的动态行为和结构。VOPC图（View, Operation, Property, Class）是对参与类类图的一种扩展，进一步细分了系统的视图、操作、属性和类，提供了更全面的系统设计视角。两者在目的上相似，都是用于分析和设计阶段。

图示, 示意图

描述已自动生成

# 实验结论

本次实验通过对医院预约挂号系统的建模分析，成功完成了面向对象的系统建模。主要包括以下几个方面：

1. 采用B-C-E分析方法，将系统合理划分为边界类、控制类和实体类三层架构，通过包图清晰展示了系统的整体结构和模块关系。这种分层设计有利于系统的维护和扩展。
2. 通过顺序图和协作图对预约挂号和支付挂号费两个核心用例进行了动态行为建模，详细描述了用户、界面、控制层和数据层之间的消息传递和交互过程，体现了系统的业务流程。
3. 使用VOPC图展示了系统的静态结构，描述了各个类之间的关联、继承等关系，体现了面向对象设计中的封装、继承、多态等特性。

实验采用StarUML工具，遵循软件工程规范，通过B-C-E分析法将系统划分为清晰的三层架构：边界层处理用户交互界面，控制层负责核心业务逻辑，实体层管理基础数据对象。通过包图清晰展示了系统的整体结构和模块关系，这种分层设计不仅提高了系统的内聚性，降低了模块间耦合度，更为后期的系统维护和功能扩展提供了良好的架构基础。

在系统行为建模方面，重点分析了预约挂号和支付挂号费两个核心业务用例。通过顺序图详细描述了用户操作、界面响应、业务处理和数据存取的完整流程，并使用协作图展示了系统各组件间的交互关系。这些动态模型不仅清晰展示了用户、界面、控制层和数据层之间的消息传递和时序关系，也完整覆盖了正常流程和异常处理场景，充分体现了系统的业务流程特征，为后续的详细设计和代码实现提供了清晰的指导。在静态结构设计中，通过VOPC图全面展示了系统的类结构，合理设计了类的属性和方法，正确建立了类之间的关联、继承等关系，充分体现了面向对象设计中的封装、继承、多态等特性，确保了系统结构的合理性和可扩展性。

通过本次实验，我不仅掌握了面向对象建模工具的使用方法，更加深了对面向对象分析与设计方法的理解，也提高了系统分析与设计能力。同时也发现了一些值得改进的方面：界面类设计可以更加细化以提升用户体验，业务规则约束可以更加完善以确保系统稳定性，类之间的关系还可以进一步优化以提高系统效率。这些问题的发现为后续的优化提供了明确的方向。总的来说，本次实验达到了预期的建模目标，积累了宝贵的系统分析与设计经验，为后续的系统实现和今后进行更复杂的软件系统开发奠定了坚实的基础。

# 仓库地址

<https://github.com/xiangshenchi/soft-engineering/tree/main/SEproject6>

<https://gitee.com/help_xsz/soft-engineering/tree/main/SEproject6>