学 院：南京大学软件学院

团 队：第5组

成 员：陈书玉 陈俐俐 程芸菲 陈悦

完成日期：2016/10/26

酒店预定系统

详细设计文档

更新历史 4

1.引言 5

1.1编制目的 5

1.2词汇表 5

1.3参考资料 5

2.产品概述 5

3.体系结构设计概述 6

4.结构视角 6

4.1业务逻辑层的分解 6

4.1.1 userbl模块 6

4.1.1.1 userbl模块概述 6

4.1.1.2 userbl整体结构 6

4.1.1.3 userbl模块内部类的接口规范 8

4.1.1.4 userbl业务逻辑层的动态模型 11

4.1.1.5 userbl业务逻辑层的设计原理 13

4.1.2 webstrategybl模块 13

4.1.2.1 webstrategybl模块概述 13

4.1.2.2 webstrategybl整体结构 14

4.1.2.3 webstrategybl模块内部类的接口规范 15

4.1.2.4 webstrategybl业务逻辑层的动态模型 18

4.1.2.5 webstrategybl业务逻辑层的设计原理 20

4.1.3 HotelStrategybl模块 20

4.1.3.1 HotelStrategybl模块概述 20

4.1.3.2 HotelStrategybl整体结构 20

4.1.3.3 HotelStrategybl模块内部类的接口规范 22

4.1.3.4 HotelStrategybl业务逻辑层的动态模型 24

4.1.3.5 HotelStrategybl业务逻辑层的设计原理 27

4.1.4 personnelbl模块 27

4.1.4.1 personnelbl模块概述 27

4.1.4.2 personnelbl整体结构 27

4.1.4.3 personnelbl模块内部类的接口规范 29

4.1.4.4 personnelbl业务逻辑层的动态模型 33

4.1.4.5 personnelbl业务逻辑层的设计原理 35

4.1.5 searchhotelbl模块 35

4.1.5.1 searchhotelbl模块概述 35

4.1.5.2 searchhotelbl整体结构 36

4.1.5.3 searchhotelbl模块内部类的接口规范 37

4.1.5.4 searchhotelbl业务逻辑层的动态模型 38

4.1.5.5 searchhotelbl业务逻辑层的设计原理 40

4.1.6 Checkinbl模块 40

4.1.6.1 Checkinbl模块概述 40

4.1.6.2 Checkinbl整体结构 40

4.1.6.3Checkinbl模块内部类的接口规范 42

4.1.6.4Checkinbl业务逻辑层的动态模型 45

4.1.6.5Checkinbl业务逻辑层的设计原理 47

4.1.7AvailableRoombl模块 47

4.1.7.1AvailableRoombl模块概述 47

4.1.7.2AvailableRoombl整体结构 47

4.1.7.3 AvailableRoombl模块内部类的接口规范 48

4.1.7.4 AvailableRoombl业务逻辑层的动态模型 51

4.1.7.5AvailableRoombl业务逻辑层的设计原理 53

4.1.8Creditbl模块 53

4.1.8.1 Creditbl模块概述 53

4.1.8.2 Creditbl整体结构 54

4.1.8.3 Creditbl模块内部类的接口规范 55

4.1.8.4Creditbl业务逻辑层的动态模型 56

4.1.8.5Creditbl业务逻辑层的设计原理 58

4.1.9 hotelbl模块 58

4.1.9.1 hotelbl模块概述 58

4.1.9.2 hotelbl整体结构 58

4.1.9.3 hotelbl模块内部类的接口规范 60

4.1.9.4 hotelbl业务逻辑层的动态模型 62

4.1.9.5 hotelbl业务逻辑层的设计原理 65

4.1.10 Order模块 65

4.1.10.1 Orderbl模块概述 65

4.1.10.2 Orderbl整体结构 65

4.1.10.3 Orderbl模块内部类的接口规范 67

4.1.10.4 Orderbl业务逻辑层的动态模型 73

4.1.10.5 Orderbll业务逻辑层的设计原理 77

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **修改日期** | **修改原因** | **版本号** |
| 陈悦,陈书玉 | 2016/10/17 | 创建模板 | V0.1 |
| 全体成员 | 2016/10/22 | 完成草稿 | V0.2 |
| 陈悦 | 2016/10/22 | 整理 | V0.3 |
| 全体成员 | 2016/10/23 | 修改草稿 | V0.4 |
| 陈悦 | 2016/10/23 | 修改后整理 | V0.5 |
| 陈书玉 | 2016/10/24 | 进行目录和标号的统一 | V0.6 |
| 陈悦 | 2016/10/26 | 将controller接口的后置条件改为无 | V0.7 |

# 1.引言

## 1.1编制目的

本报告详细完成对酒店预订系统的详细设计，达到指导后续软件构造的目的，同时实现和测试人员及用户的沟通。

本报告面向开发人员、测试人员及最终用户而编写，是了解系统的导航。

## 1.2词汇表

|  |  |
| --- | --- |
| **缩写或单词** | **解释** |
| \_ui | 表示某展示层 |
| \_bl | 表示某逻辑层 |
| \_data | 表示某数据层 |
| RMI | 表示远程方法调用 |
| \_Str\_ | 表示某具体策略 |

## 1.3参考资料

1. 酒店预定系统用例文档
2. 酒店预定系统需求规格说明文档V1.1
3. 酒店预定系统体系设计文档V1.2

4）2.丁二玉，刘钦.计算与软件工程（卷二）[M]机械工业出版2012：134—182

# 2.产品概述

参考《酒店预订系统用例文档》和《酒店预订系统需求规格说明文档》中对产品的概括描述。

# 3.体系结构设计概述

参考《酒店预订系统体系结构文档》中对产品的概括描述。

# 4.结构视角

## 4.1业务逻辑层的分解

### 4.1.1 userbl模块

#### 4.1.1.1 userbl模块概述

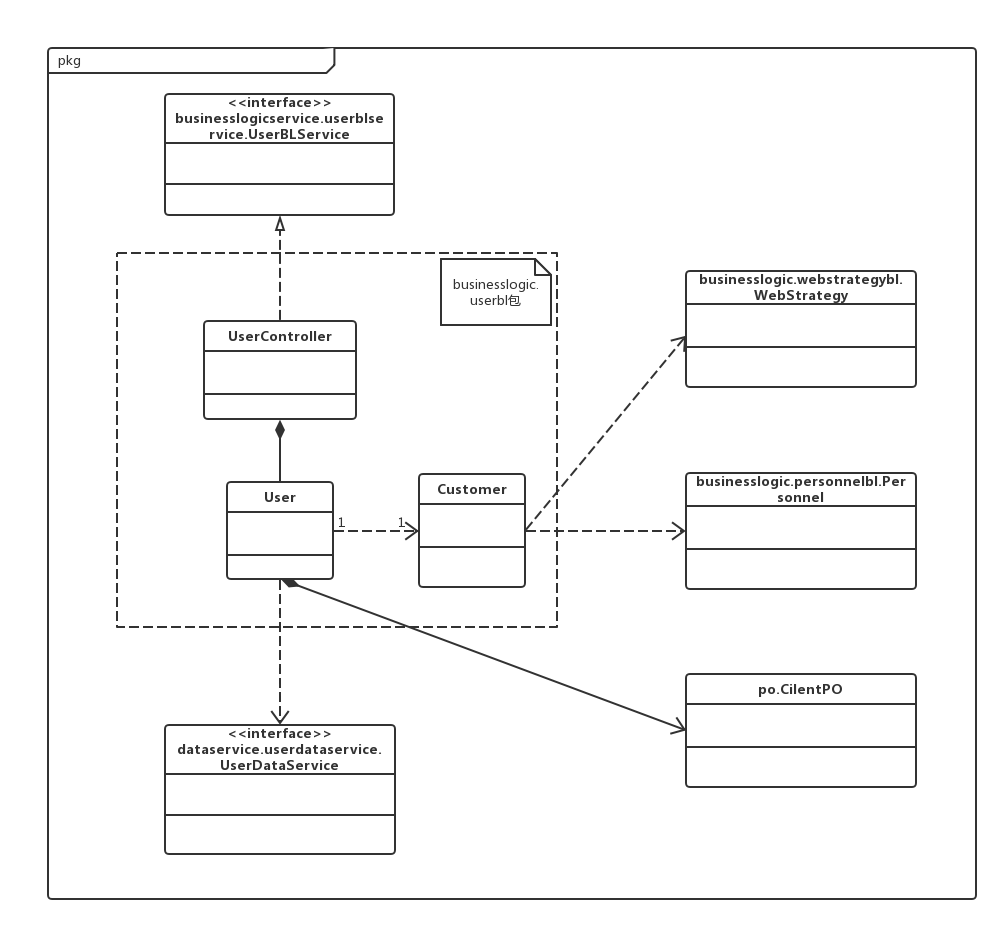
userbl 模块承担的需求见需求规格说明书文档功能需求及相关非功能需求。

userbl 模块的职责及接口参见软件体系结构描述文档。

#### 4.1.1.2 userbl整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。在展示层和业务逻辑层之间添加businesslogicservice.userblservice.UserBLService接口。在业务逻辑层和数据层之间添加dataservice.userdataservice.UserDataService接口。为了隔离业务逻辑职责和逻辑控制职责，添加UserController，这样UserController会将对用户的业务逻辑处理委托给User对象。CustomerPO是作为用户信息的持久性对象被添加到设计模型中去的。Customer的添加是让User将维护个人信息的任务委托给Customer去完成。Customer保有顾客个人详细信息的数据和维护个人信息的职责。Customer依赖businesslogic.webstrategybl.WebStrategy和businesslogic.personnelbl.Personnel来获取和更改顾客个人详细信息。

userbl模块的设计如图4．1.1.2-1所示。



4．1.1.2-1 userbl模块类的设计

userbl模块各个类的职责如表4．1.1.2-1所示

表4．1.1.2-1 userbl模块各个类的职责

|  |  |
| --- | --- |
| **模块** | **职责** |
| UserController | 负责实现对应于登陆界面所需要的服务以及维护个人信息的服务。 |
| User | 系统用户的领域模型对象，拥有用户数据的姓名，密码，可以解决登录和修改密码的问题 |
| Customer | 顾客的领域模型对象，拥有顾客详细信息，可以提供维护个人信息的服务 |

#### 4.1.1.3 userbl模块内部类的接口规范

UserController和User的接口规范如表4.1.1.3-1和表4.1.1.3-2所示

表4.1.1.3-1 UserController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| UserController.login | 语法 | Public ResultMessage login(String name ,String password) |
| 前置条件 | 已经创建一个User领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| UserController.getDetailInfo | 语法 | Public DetailInfoVO getDetailInfo(String name) |
| 前置条件 | 已经创建一个User领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| UserController. checkTel | 语法 | Public ResultMessage checkTel（String tel） |
| 前置条件 | 已经创建一个User领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| UserController. Confirm | 语法 | Public ResultMessage confirmUserInfo（DetailInfoVO vo） |
| 前置条件 | 已经创建一个User领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| UserController.checkOldPassword | 语法 | Public ResultMessage checkOldPassword（String name，String password） |
| 前置条件 | 已经创建一个User领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| UserController.checkNewPassword | 语法 | Public ResultMessage checkNewPassword（String password） |
| 前置条件 | 已经创建一个User领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| UserController.confirmPassword | 语法 | Public ResultMessage confirmPassword（PasswordVO passwordVO） |
| 前置条件 | 已经创建一个User领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| UserController.getBasicInfo | 语法 | Public BasicInfoVO getBasicInfo(String name) |
| 前置条件 | 已经创建一个User领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| UserController.getUserID | 语法 | Public String getUserID (String name) |
| 前置条件 | 已经创建一个User领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| UserController.getGrade | 语法 | Public int getGrade（String name） |
| 前置条件 | 已经创建一个User领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | |
| User. login | 用户登录 | |
| User. getDetailInfo | 获得顾客详细个人信息 | |
| User. checkTel | 检查输入的电话号码格式 | |
| User. Confirm | 确认修改的个人信息 | |
| User. checkOldPassword | 检查旧密码是否正确 | |
| User. checkNewPassword | 检查新密码的格式是否正确 | |
| User. confirmPassword | 确认修改的密码，若密码一致，更新个人信息。否则提示密码不一致 | |
| User. getBasicInfo | 获得用户基本信息 | |
| User. getUserID | 获得用户编号 | |
| User. getGrade | 获得顾客会员等级 | |

表4.1.1.3-2 User的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| User.Login | 语法 | Public ResultMessage login(String name ,String password) |
| 前置条件 | 账号、密码符合语法规则 |
| 后置条件 | 无 |
| User. getDetailInfo | 语法 | Public DetailInfoVO getDetailInfo(String name) |
| 前置条件 | 用户已登录 |
| 后置条件 | 无 |
| User. checkTel | 语法 | Public ResultMessage checkTel（String tel） |
| 前置条件 | 无 |
| 后置条件 | 无 |
| User. ConfirmUserInfo | 语法 | Public ResultMessage confirmUserInfo（DetailInfoVO vo） |
| 前置条件 | 输入信息格式正确 |
| 后置条件 | 系统更新个人信息 |
| User. checkOldPassword | 语法 | Public ResultMessage checkOldPassword（String name，String password） |
| 前置条件 | 用户应登录 |
| 后置条件 | 无 |
| User. checkNewPassword | 语法 | Public ResultMessage checkNewPassword（String password） |
| 前置条件 | 用户已确认旧密码 |
| 后置条件 | 无 |
| User. confirmPassword | 语法 | Public ResultMessage confirmPassword（PasswordVO passwordVO） |
| 前置条件 | 新密码格式正确 |
| 后置条件 | 若密码一致，更新个人信息数据。 |
| User. getBasicInfo | 语法 | Public BasicInfoVO getBasicInfo(String name) |
| 前置条件 | 用户已登录 |
| 后置条件 | 无 |
| User. getUserID | 语法 | Public String getUserID (String name) |
| 前置条件 | 用户已登录 |
| 后置条件 | 无 |
| User. getGrade | 语法 | Public int getGrade（String name） |
| 前置条件 | 用户已登录 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | |
| UserDataService. getUserPassword | 提供VO返回用户密码信息 | |
| UserDataService. setPassword | 根据提供的VO修改用户信息 | |
| DatabaseFactory.getUserDataService | 得到User数据库服务的引用 | |
| Customer. getDetailInfo | 得到顾客详细信息 | |
| Customer. getGrade | 得到顾客会员等级信息 | |
| Customer.modifyDetailInfo | 修改顾客详细信息 | |

表4.1.1.3-3 Customer的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Customer. getGrade | 语法 | Public int getGrade（String name） |
| 前置条件 | 用户已经登录 |
| 后置条件 | 无 |
| Customer. getDetailInfo | 语法 | Public DetailInfoVO getDetailInfo(String name) |
| 前置条件 | 用户已经登录 |
| 后置条件 | 无 |
| Customer.modifyDetailInfo | 语法 | Public ResultMessage modifyDetailInfo(DetailInfoVO detailInfoVO) |
| 前置条件 | 用户已经登录 |
| 后置条件 | 更新顾客个人详细信息 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | |
| WebStrategy.getGradeRule | 根据webstr 返回顾客会员等级信息 | |
| Personnel.getPersonDetail | 根据顾客编号返回顾客详细信息 | |
| Personnel.modifyPerson | 修改顾客个人信息 | |

#### 4.1.1.4 userbl业务逻辑层的动态模型

图4.1.1.4-1表明了酒店预订系统中，当用户修改密码后，用户业务逻辑处理的相关对象之间的协作。

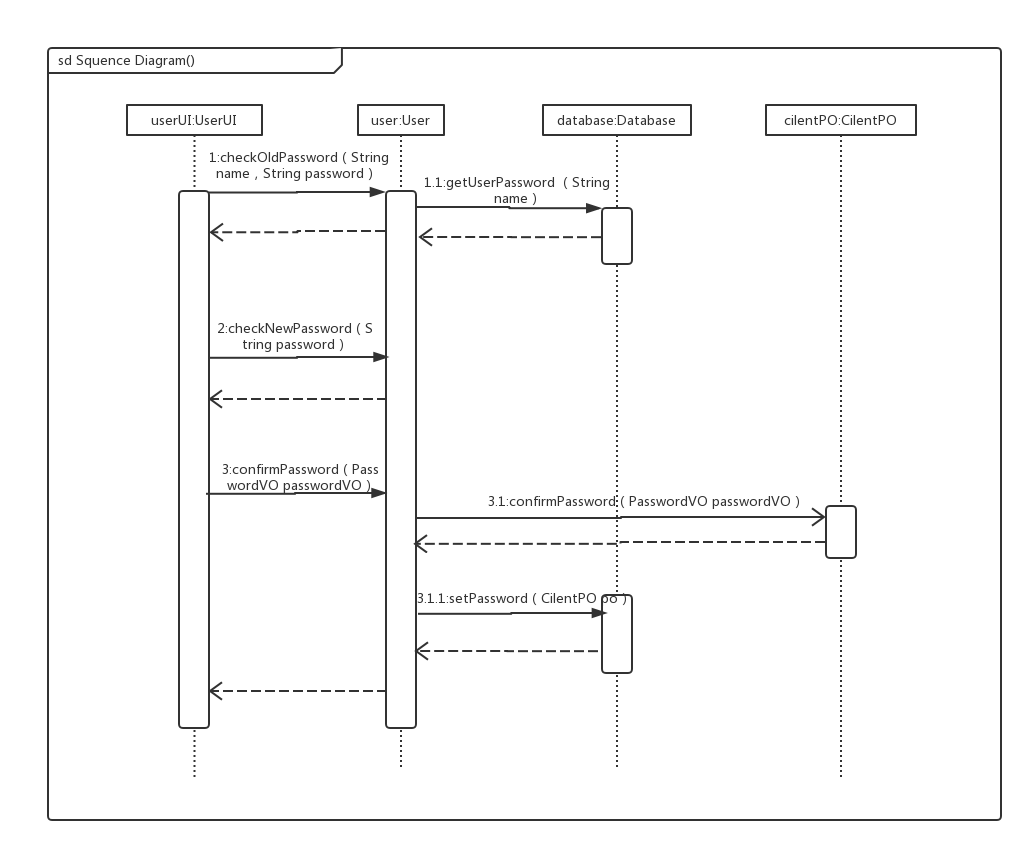


图4.1.1.4-1 用户修改密码的顺序图

图4.1.1.4-2表明了酒店预订系统中，当顾客维护个人信息时，用户业务逻辑处理的相关对象之间的协作。

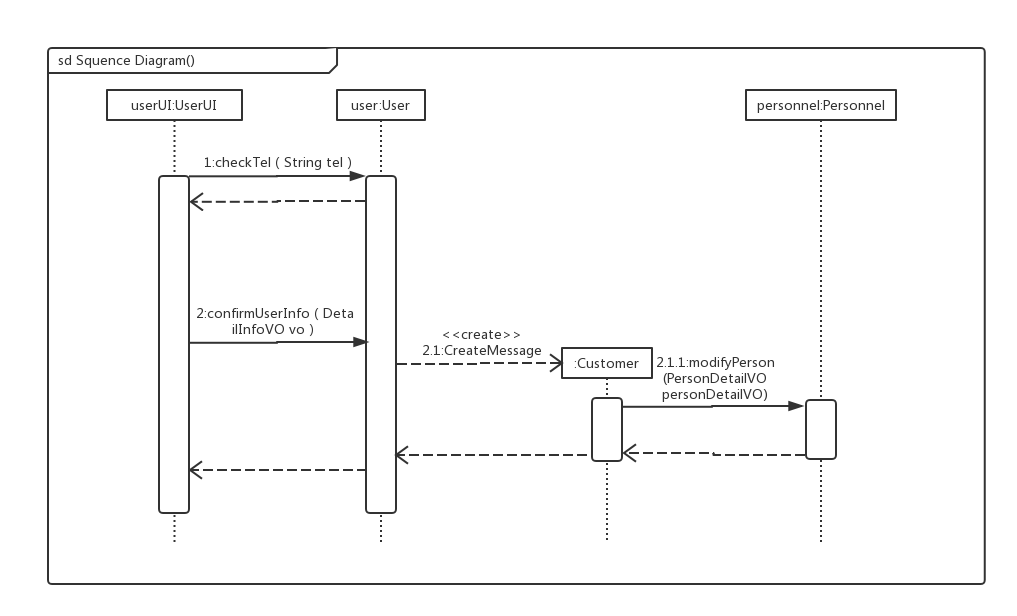


图4.1.1.4-2 维护个人信息的顺序图

如图4.1.1.4-3所示的状态图描述了User对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。随着login方法被UI调用，进入User状态，之后通过修改密码进入Password状态，也可通过维护个人信息进入CustomerInfo状态。

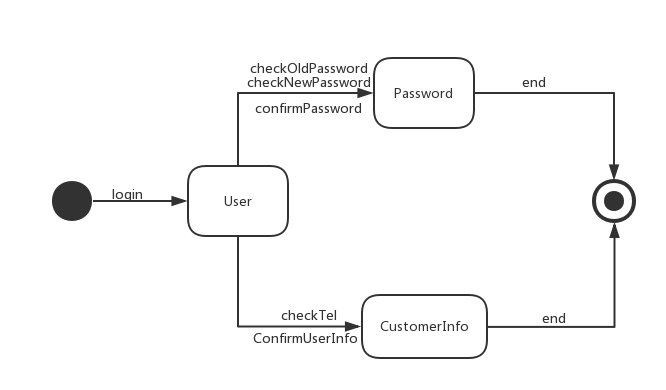


图4.1.1.4-3User对象状态图

#### 4.1.1.5 userbl业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。

### 4.1.2 webstrategybl模块

#### 4.1.2.1 webstrategybl模块概述

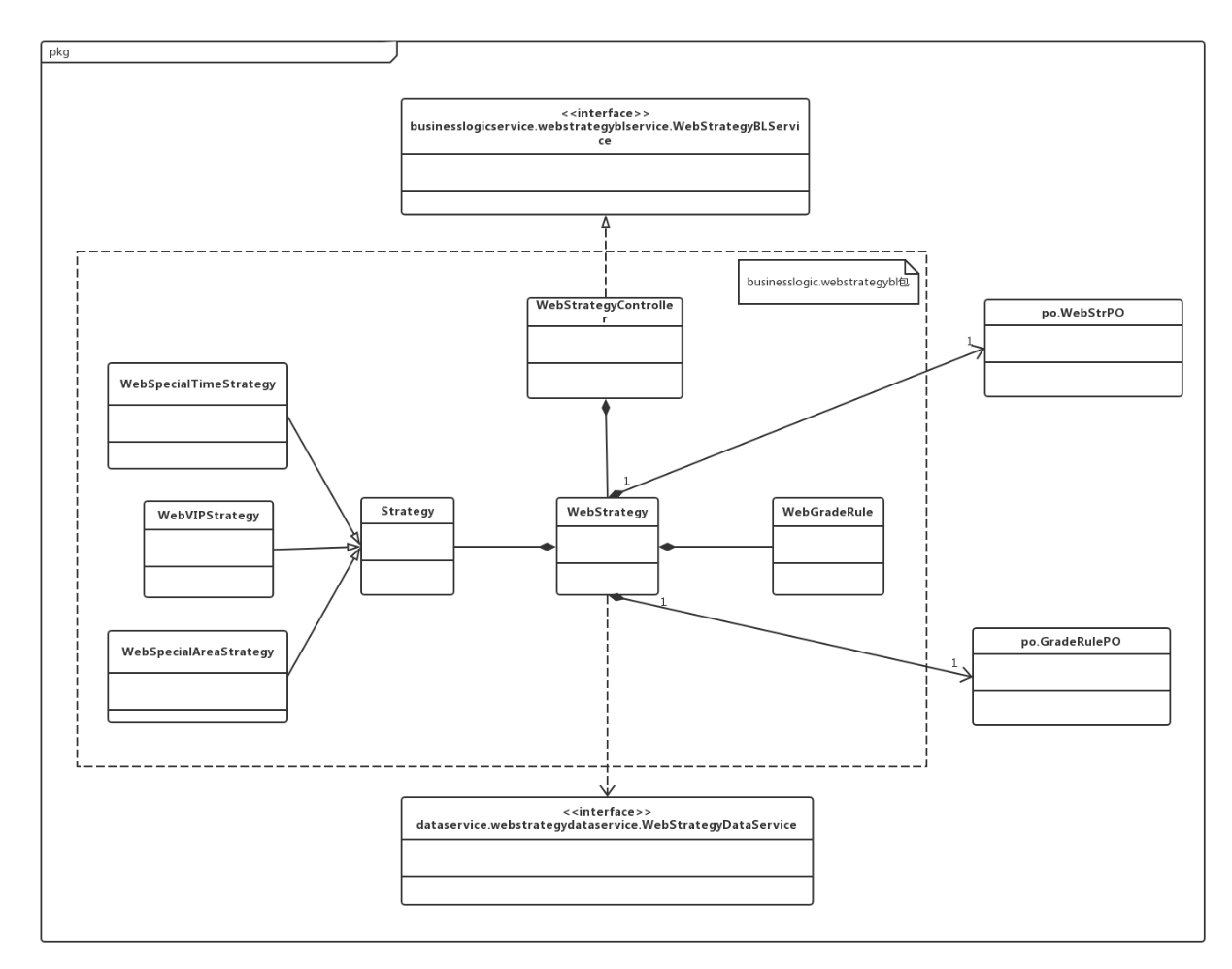
webstrategybl 模块承担的需求见需求规格说明书文档功能需求及相关非功能需求。

webstrategybl 模块的职责及接口参见软件体系结构描述文档。

#### 4.1.2.2 webstrategybl整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。在展示层和业务逻辑层之间添加businesslogicservice.webstrategyblservice.WebStrategyBLService接口。在业务逻辑层和数据层之间添加dataservice.webstrategydataservice.WebStrategyDataService接口。为了隔离业务逻辑职责和逻辑控制职责，添加WebStrategyController，这样WebStrategyController会将对用户的业务逻辑处理委托给WebStrategy对象。GradeRulePO是作为会员等级规则信息的持久性对象被添加到设计模型中去的。WebStrPO是作为网站营销策略信息的持久性对象被添加到设计模型中去的。Strategy的添加是WebSpecialTimeStrategy，WebVIPStrategy和WebSpecialAreaStrategy的父类。Strategy封装了折扣数据和修改策略的职责。WebGradeRule保有会员等级规则信息的数据和修改会员等级规则的职责。

webstrategybl模块的设计如图4．1.2.2-1所示。



4．1.2.2-1 webstrategybl模块类的设计

webstrategybl模块各个类的职责如表4．1.2.2-1所示

表4．1.2.2-1 webstrategybl模块各个类的职责

|  |  |
| --- | --- |
| **模块** | **职责** |
| WebStrategyController | 负责实现修改网站营销策略的服务。 |
| WebStrategy | 网站策略的领域模型对象，拥有策略和会员等级规则信息，可以解决修改网站营销策略的问题 |
| Strategy | 策略的领域模型对象，拥有折扣信息，可以提供修改策略的服务 |
| WebSpecialTimeStrategy | 网站特惠时期折扣策略的领域模型对象，拥有退回时间信息和折扣值，可以提供修改策略的服务 |
| WebVIPStrategy | 网站VIP会员折扣策略的领域模型对象，拥有会员等级和折扣值，可以提供修改策略的服务 |
| WebSpecialAreaStrategy | 网站特定商圈折扣策略的领域模型对象，拥有商圈和折扣值，可以提供修改策略的服务 |
| WebGradeRule | 网站会员等级规则的领域模型对象，拥有会员等级和信用值等信息，可以提供修改规则的服务 |

#### 4.1.2.3 webstrategybl模块内部类的接口规范

WebStrategyController和WebStrategy的接口规范如表4.1.2.3-1和表4.1.2.3-2所示

表4.1.2.3-1 WebStrategyController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| WebStrategyController. getWebBestStrategy | 语法 | Public WebBestStrVO getWebBestStrategy (String credit,TradingArea area,Date time) |
| 前置条件 | 已经创建一个WebStrategy领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| WebStrategyController. getGradeRule | 语法 | Public GradeRuleVO getGradeRule（） |
| 前置条件 | 已经创建一个WebStrategy领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| WebStrategyController. getWebStrategy | 语法 | Public WebStrVO getWebStrategy(WebStrategyType type) |
| 前置条件 | 已经创建一个WebStrategy领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| WebStrategyController. comfirmGradeRule | 语法 | Public ResultMessage comfirmGradeRule（GradeRuleVO vo） |
| 前置条件 | 已经创建一个WebStrategy领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| WebStrategyController. comfirmWebStrategy | 语法 | Public ResultMessage comfirmWebStrategy（WebStrVO vo） |
| 前置条件 | 已经创建一个WebStrategy领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | |
| WebStrategy. getWebBestStrategy | 获得最佳网站营销策略 | |
| WebStrategy. getGradeRule | 获得会员等级制定规则信息 | |
| WebStrategy. getWebStrategy | 获得网站策略信息 | |
| WebStrategy. comfirmGradeRule | 更新会员等级制定规则信息 | |
| WebStrategy. comfirmWebStrategy | 系统更新网站策略信息 | |

表4.1.2.3-2 WebStrategy的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| WebStrategy. getWebBestStrategy | 语法 | Public WebBestStrVO getWebBestStrategy (String credit,TradingArea area,Date time) |
| 前置条件 | 无 |
| 后置条件 | 无 |
| WebStrategy. getGradeRule | 语法 | Public GradeRuleVO getGradeRule（） |
| 前置条件 | 无 |
| 后置条件 | 无 |
| WebStrategy. getWebStrategy | 语法 | Public WebStrVO getWebStrategy(WebStrategyType type) |
| 前置条件 | 无 |
| 后置条件 | 无 |
| WebStrategy. comfirmGradeRule | 语法 | Public ResultMessage comfirmGradeRule（GradeRuleVO vo） |
| 前置条件 | 输入信息格式正确 |
| 后置条件 | 系统更新会员等级规则的数据 |
| WebStrategy. comfirmWebStrategy | 语法 | Public ResultMessage comfirmWebStrategy（WebStrVO vo） |
| 前置条件 | 输入信息格式正确 |
| 后置条件 | 系统更新网站策略的数据 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | |
| WebGradeRule.getGradeRule | 返回会员等级制定规则信息 | |
| WebGradeRule.setGradeRuleategy | 系统更新会员等级制定规则信息 | |
| Strategy .getWebStrategy | 返回网站策略信息 | |
| Strategy.setWebStrategy | 系统更新网站策略信息 | |

表4.1.2.3-3 Strategy的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Strategy. getWebStrategy | 语法 | Public WebStrVO getWebStrategy() |
| 前置条件 | 无 |
| 后置条件 | 无 |
| Strategy. setWebStrategy | 语法 | Public ResultMessage setWebStrategy（WebStrVO vo） |
| 前置条件 | 输入信息格式正确 |
| 后置条件 | 系统更新网站策略的数据 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | |
| WebstrDataService .getWebStrategy | 返回网站策略信息 | |
| WebstrDataService .setWebStrategy | 系统更新网站策略信息 | |

表4.1.2.3-4 WebGradeRule的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| WebGradeRule.getGradeRule | 语法 | Public GradeRuleVO getGradeRule（） |
| 前置条件 | 无 |
| 后置条件 | 无 |
| WebGradeRule.setGradeRule | 语法 | Public ResultMessage setGradeRule（GradeRuleVO vo） |
| 前置条件 | 输入信息格式正确 |
| 后置条件 | 系统更新会员等级规则 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | |
| WebstrDataService. getGradeRule | 获得会员等级制定规则信息 | |
| WebstrDataService.setGradeRule | 更新会员等级制定规则信息 | |

#### 4.1.2.4 webstrategybl业务逻辑层的动态模型

图4.1.2.4-1表明了酒店预订系统中，当用户生成订单获取最佳网站策略时，网站策略业务逻辑处理的相关对象之间的协作。

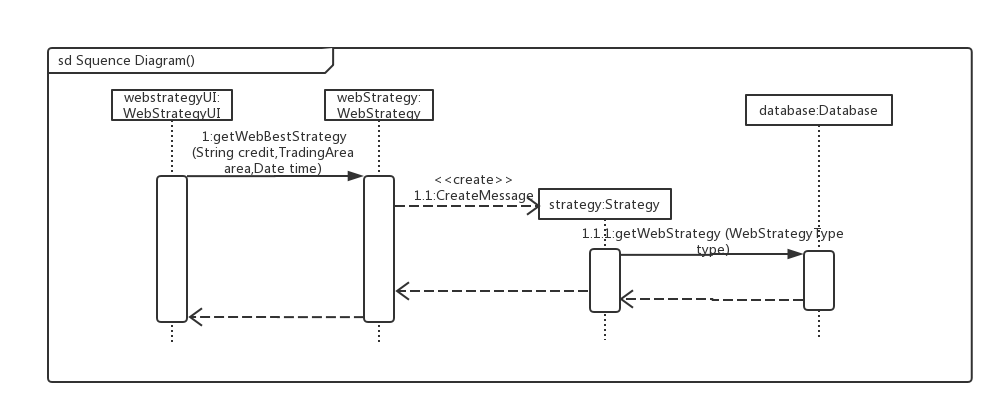


图4.1.2.4-1 获取最佳网站策略的顺序图

图4.1.2.4-1表明了在酒店预订系统中，当网站营销人员制定网站策略时，网站策略业务逻辑处理的相关对象之间的协作。

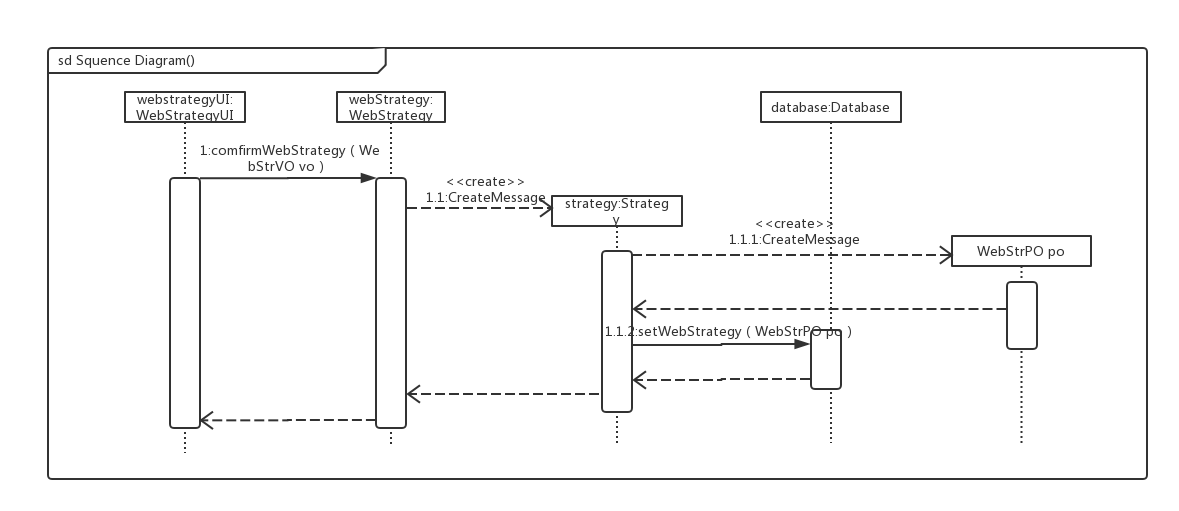


图4.1.2.4-2 制定网站策略的顺序图

如图4.1.2.4-3所示的状态图描述了WebStrategy对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。随着getWebBestStrategy方法被UI调用，进入WebStrategy状态，之后通过获取网站营销策略进入Strategy状态，也可通过获取或制定网站营销策略进入Strategy状态，可通过获取或制定会员等级规则进入GradeRule状态。

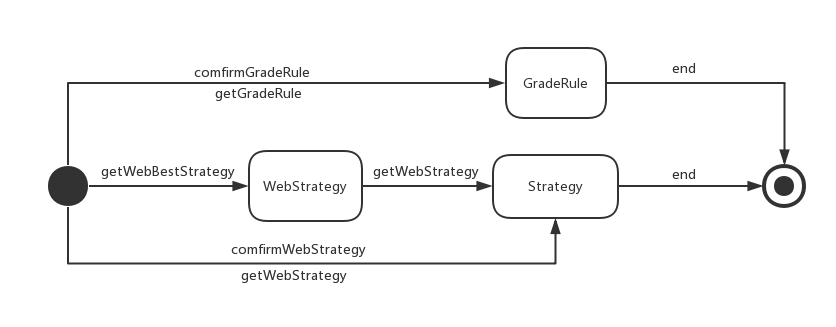


图4.1.2.4-3 WebStrategy对象状态图

#### 4.1.2.5 webstrategybl业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。

### 4.1.3 HotelStrategybl模块

#### 4.1.3.1 HotelStrategybl模块概述

HotelStrategybl模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

HotelStrategybl模块的职责及接口参见软件体系结构描述文档。

#### 4.1.3.2 HotelStrategybl整体结构

根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了HotelStrategyblservice,HotelStrategy dataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了HotelStrategyController，这样HotelStrategyController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给HotelStrategy 对象。HotelStrPO是做为酒店优惠策略信息的持久化对象被添加到设计模型中的。而Strategy的添加是所有策略对象的父类，HotelBirthStrategy,HotelAmountStrategy,HotelEnterpriseStrategy,HotelSpecialTimeStrategy继承Strategy。

HotelStrategybl模块的设计如图4．1.3.2-1所示。

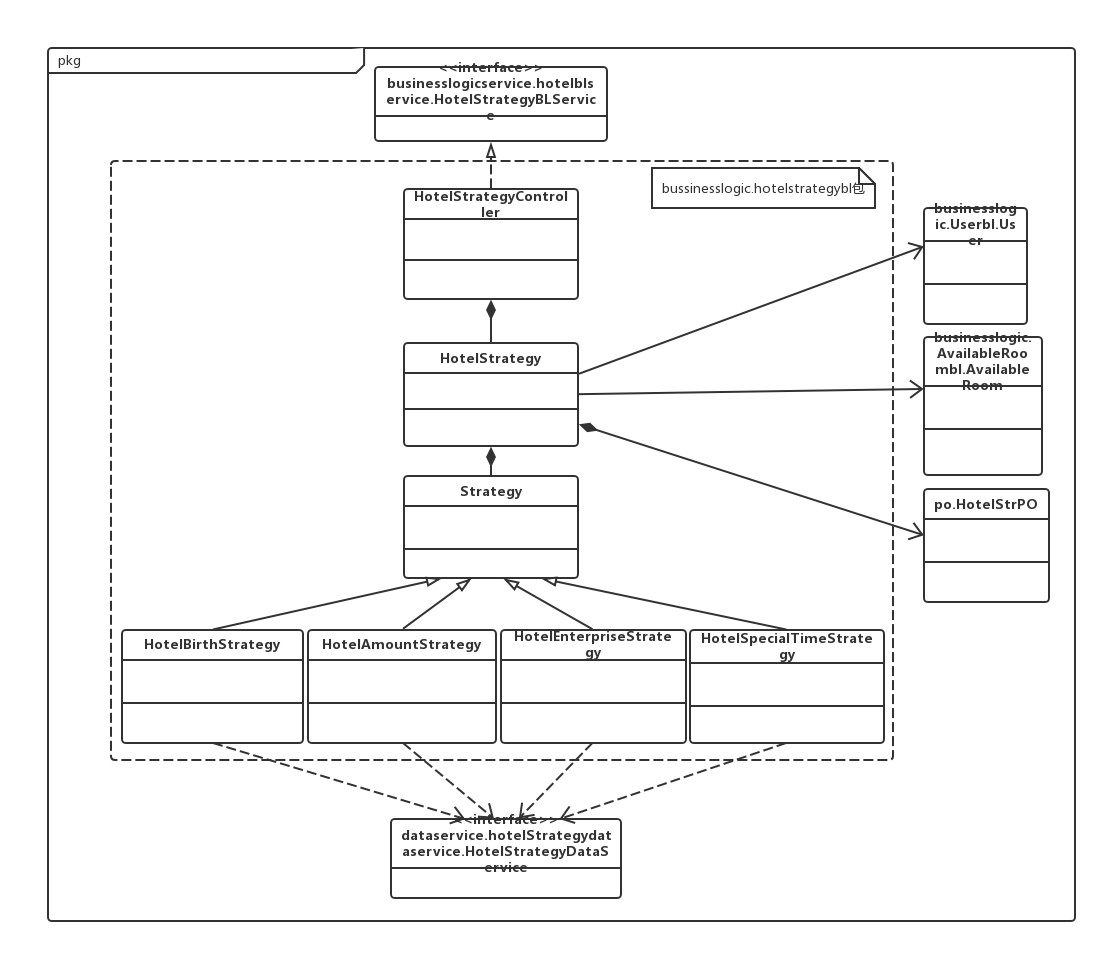


图4．1.3.2-1 HotelStrategybl模块类的设计

HotelStrategybl模块各个类的职责如表4．1.3.2-1所示

表4．1.3.2-1 HotelStrategybl模块各个类的职责

|  |  |
| --- | --- |
| **模块** | **职责** |
| HotelStrategyController | 负责实现酒店策略界面所需要的服务 |
| HotelStrategy | 负责实现酒店策略逻辑部分，计算酒店房间最低价格，返回最优策略 |
| Strategy | 系统策略的领域模型对象，持有酒店编号，折扣值，策略类型，负责增加、修改、返回策略 |

#### 4.1.3.3 HotelStrategybl模块内部类的接口规范

表4.1.3.3-1 HotelStrategyController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| HotelStrategyController.getBestHotelStrategy | 语法 | Public HotelBestStrVO getBestHotelStrategy(OrderProvidedVO vo) |
| 前置条件 | 已创建一个HotelStrategy领域对象，顾客已成功下订单 |
| 后置条件 | 无 |
| HotelStrategyController.getHotelStrategy | 语法 | Public HotelStrVO getHotelStrategy(String hotelID, HotelStrategyType type) |
| 前置条件 | 已创建一个HotelStrategy领域对象，酒店制定过优惠策略 |
| 后置条件 | 无 |
| HotelStrategyController.confirmHotelStrategy | 语法 | Public ResultMessage confirmHotelStrategy(HotelStrVO hotelStrVO) |
| 前置条件 | 已创建一个HotelStrategy领域对象，在presentation层已经输入该策略信息并确认 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| HotelStrategy.getHotelStrategy | 从数据库中获得该酒店的营销策略 | | |
| HotelStrategy. setHotelStrategy | 更改单一化持久对象 | | |
| HotelStrategy.setBestPrice | 更新酒店房间的最低价格 | | |
| HotelStrategy.getUserID | 获得用户编号 | | |

表4.1.3.3-2HotelStrategy的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| HotelStrategy.getBestHotelStrategy | 语法 | Public HotelBestStrVO getBestHotelStrategy(OrderProvidedVO vo) |
| 前置条件 | 顾客成功下订单 |
| 后置条件 | 无 |
| HotelStrategy.getHotelStrategy | 语法 | Public HotelStrVO getHotelStrategy(String hotelID, HotelStrategyType type) |
| 前置条件 | 酒店制定过优惠策略 |
| 后置条件 | 无 |
| HotelStrategy.confirmHotelStrategy | 语法 | Public ResultMessage confirmHotelStrategy(HotelStrVO hotelStrVO) |
| 前置条件 | 无 |
| 后置条件 | 系统数据库更新该酒店的策略 |
| HotelStrategy.setHotelStrategy | 语法 | Public ResultMessage setHotelStrategy(HotelStrPO po) |
| 前置条件 | 数据库中存在该酒店策略 |
| 后置条件 | 系统数据库修改该策略 |
| HotelStrategy.setBestPrice | 语法 | public ResultMessage setBestPrice (double discount) |
| 前置条件 | 折扣值符合输入语法规范 |
| 后置条件 | 系统持久化更新各种房型的最低价格 |
| HotelStrategy.getUserID | 语法 | Public String getUserID (String name) |
| 前置条件 | 用户已登录 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| Availableroom.setBestPrice | 更新酒店房间的最低价格 | | |
| User.getUserID | 获得用户编号 | | |
| Strategy.setHotelStrategy | 修改酒店策略 | | |
| Strategy.getHotelStrategy | 获得酒店策略 | | |

表4.1.3.3-3 Strategy的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Strategy.setHotelStrategy | 语法 | Public ResultMessage setHotelStrategy(HotelStrVO vo) |
| 前置条件 | 数据库中存在该酒店策略 |
| 后置条件 | 系统修改该策略 |
| Strategy.getHotelStrategy | 语法 | Public Strategy getHotelStrategy(String hotelId) |
| 前置条件 | 酒店制定过优惠策略 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| HotelStrategyDataService.getHotelStrategy | 从数据库中获得该酒店的营销策略 | | |
| HotelStrategyDataService. setHotelStrategy | 更改单一化持久对象 | | |

#### 4.1.3.4 HotelStrategybl业务逻辑层的动态模型

图4.1.3.4-1表明了在酒店预定系统中，酒店工作人员制定策略，酒店策略业务逻辑处理的相关对象的协作。

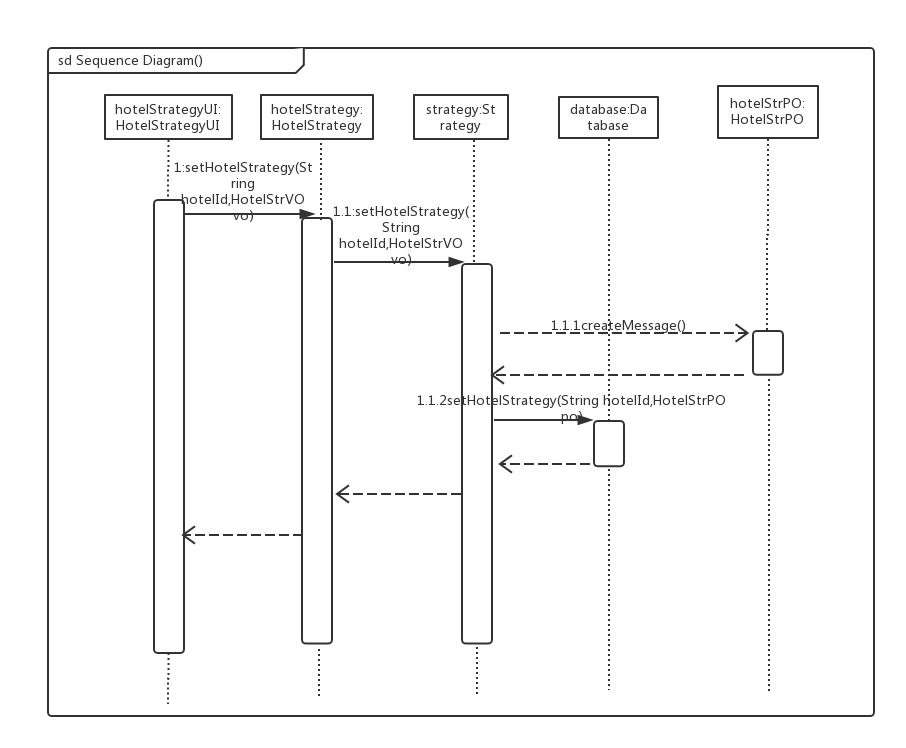


图4.1.3.4-1制定策略的顺序图

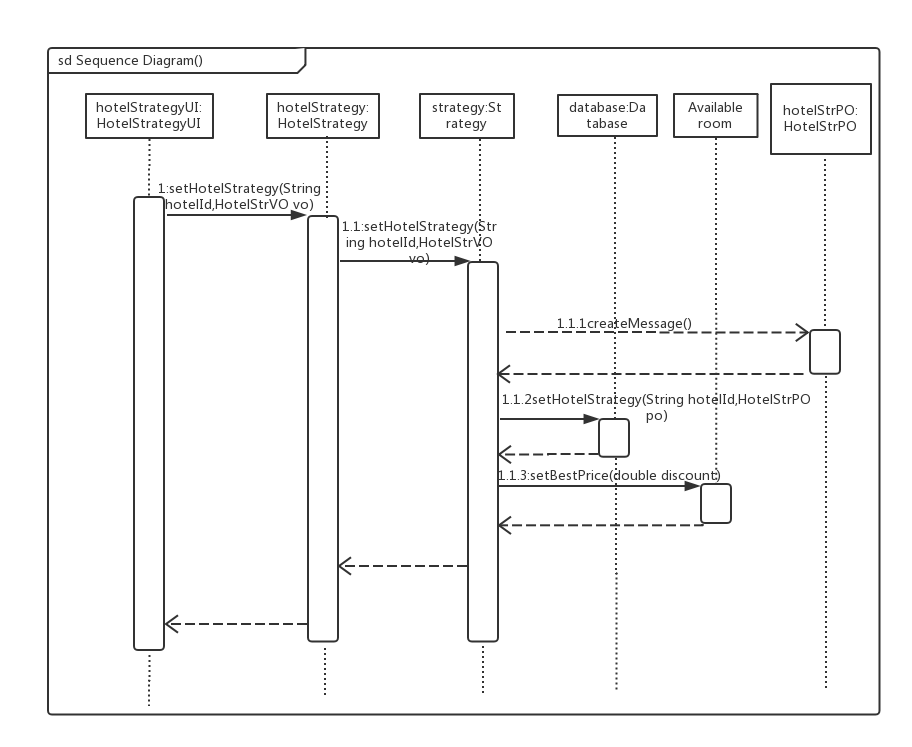
图4.1.3.4-2 表明了酒店工作人员制定策略后，更新可用房间的业务逻辑的相关对象的协作。

图4.1.3.4-2更新房型价格的顺序图

图4.1.3.4-3表明了顾客生成订单时，系统需要返回最佳策略的酒店策略的业务逻辑的相关对象的协作。

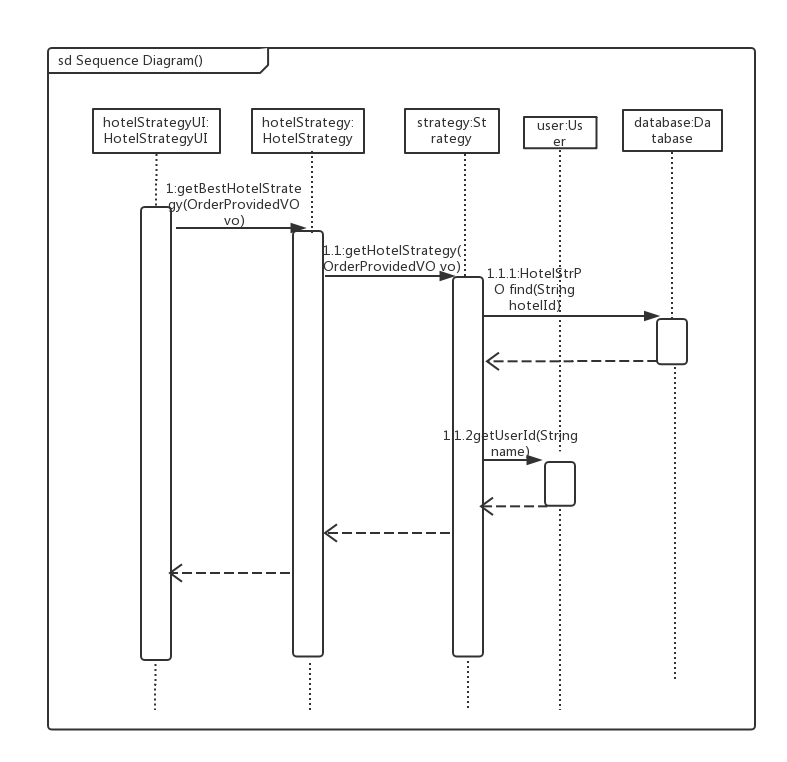


图4.1.3.4-3获得最佳策略的顺序图

图4.1.3.4-4所示的状态图描述了HotelStrategy对象的生存期间的状态序列、引起状态转移而伴随的动作。随着setHotelStrategy方法被UI调用，HotelStrategy进入Strategy状态。

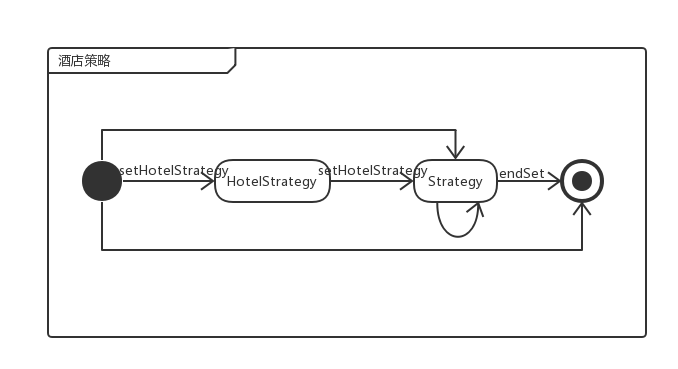


图4.1.3.4-4 HotelStrategy对象状态图

#### 4.1.3.5 HotelStrategybl业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。

### 4.1.4 personnelbl模块

#### 4.1.4.1 personnelbl模块概述

**personnelbl**模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

**personnelbl**模块的职责和接口参见系统结构描述文档内对该模块的描述。

#### 4.1.4.2 personnelbl整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。在展示层和业务逻辑层之间添加businesslogicservice.personnelblservice.PersonnelblService接口。在业务逻辑层和数据层之间添加dataservice.personneldataservice.PersonneldataService接口。为了隔离业务逻辑职责和逻辑控制职责，添加PersonnelController，这样PersonnelController会将对用户管理的业务逻辑处理委托给Personnel对象。CustomerDetailPO、HotelWorkerDetailPO、WebMarketManDetailPO、HotelWorkerListPO、WebMarketManListPO、CustomerListPO、personDetailPO、personListPO是作为人员信息的持久性对象被添加到设计模型中去的。PersonList是Person对象的容器类。Person保有用户详细信息的数据。WebMarketMan、Customer、HotelWorker继承Person分别保有网站营销人员、顾客、酒店工作人员的数据，并且可以对数据库进行数据操作。

personnelbl模块的设计如图4．1.4.2-1所示。

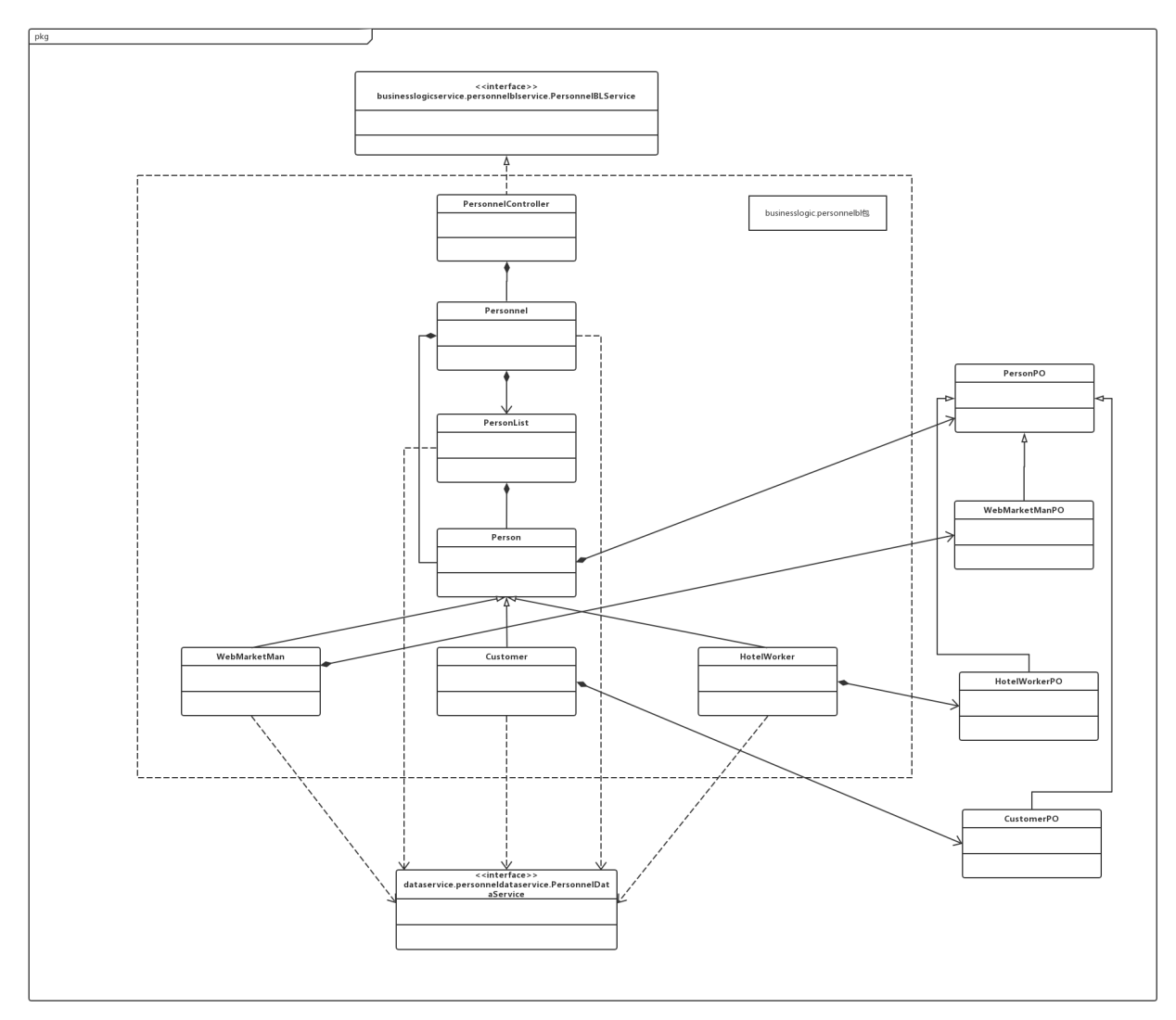


图4.1.4.2-1 personnelbl模块各个类的设计

personnelbl模块各个类的职责如表4．1.4.2-2所示。

|  |  |
| --- | --- |
| 类 | 职责 |
| PersonnelController | 负责实现对应用户管理界面所需要的服务 |
| Personnel | 用户管理的领域模型对象，拥有用户列表、用户等信息，可以帮助完成管理用户界面所需要的服务 |
| PersonList | 用户列表的领域模型对象 |
| Person | 用户的领域模型对象 |

图4．1.4.2-2 personnelbl各个类的职责

#### 4.1.4.3 personnelbl模块内部类的接口规范

表4.1.4.3-1 PersonnelController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| PersonnelController.checkUserName | 语法 | | Public ResultMessage checkUserName(String userName) |
| 前置条件 | | 已经创建一个Personnel领域对象，并且输入符合输入规则 |
| 后置条件 | | 无 |
| PersonnelController.checkTel | 语法 | | Public ResultMessage checkTel(String Tel) |
| 前置条件 | | 已经创建一个Personnel领域对象，并且输入符合输入规则 |
| 后置条件 | | 无 |
| PersonnelController.checkPassword | 语法 | | Public ResultMessage checkPassword(String password) |
| 前置条件 | | 已经创建一个Personnel领域对象，并且输入符合输入规则 |
| 后置条件 | | 无 |
| PersonnelController.addPerson | 语法 | | Public ResultMessage addPerson(PersonDetailVO personDetailVO) |
| 前置条件 | | 已经创建一个Personnel领域对象，并且输入符合输入规则 |
| 后置条件 | | 无 |
| PersonnelController.getPersonList | 语法 | Public ArrayList<PersonListVO> getPersonList(String keyWord) | |
| 前置条件 | 已经创建一个Personnel领域对象，并且输入符合输入规则 | |
| 后置条件 | 无 | |
| PersonnelController.getPersonDetail | 语法 | | Public PersonDetailVO getPersonDetail(String personID) |
| 前置条件 | | 已经创建一个Personnel领域对象，并且输入符合输入规则 |
| 后置条件 | | 无 |
| PersonnelController.modifyPerson | 语法 | | Public ResultMessage modifyPerson (PersonDetailVO personDetailVO) |
| 前置条件 | | 已经创建一个Personnel领域对象，并且输入符合输入规则 |
| 后置条件 | | 无 |
| 需要的服务（需接口） | | | |
| 服务名 | 服务内容 | | | |
| Personnel.checkUserName | 检查用户名是否注册 | | | |
| Personnel.checkTel | 检查联系方式是否符合格式 | | | |
| Personnel.checkPassword | 检查密码长度 | | | |
| Personnel.addPerson | 增加用户 | | | |
| Personnel.getList | 获得用户列表信息 | | | |
| Personnel.getPersonDetail | 获得某一person的信息 | | | |
| Personnel.setPerson | 修改用户信息 | | | |

表4.1.4.3-2 Personnel的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Personnel.checkUserName | 语法 | Public ResultMessage checkUserName(String userName) |
| 前置条件 | 顾客输入用户名 |
| 后置条件 | 无 |
| Personnel.checkTel | 语法 | Public ResultMessage checkTel(String Tel) |
| 前置条件 | 用户输入联系方式 |
| 后置条件 | 无 |
| Personnel.checkPassword | 语法 | Public ResultMessage checkPassword(String password) |
| 前置条件 | 用户输入密码 |
| 后置条件 | 无 |
| Personnel.addPerson | 语法 | Public ResultMessage addPerson(PersonDetailVO personDetailVO) |
| 前置条件 | 顾客已经注册成功或者管理人员已经被识别 |
| 后置条件 | 无 |
| Personnel.getPersonList | 语法 | Public ArrayList<PersonListVO> getPersonList(String keyWord) |
| 前置条件 | 管理人员已经被识别 |
| 后置条件 | 无 |
| Personnel.getPersonDetail | 语法 | Public PersonDetailVO getPersonDetail(String personID) |
| 前置条件 | 管理人员已经被识别 |
| 后置条件 | 无 |
| Personnel.setPerson | 语法 | Public ResultMessage setPerson (PersonDetailVO personDetailVO) |
| 前置条件 | 输入的信息格式符合规范 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| Person.addPerson | 添加用户 | | |
| PersonList.init | 初始化列表 | | |
| PersonList.getPerson | 获得某一person的信息 | | |
| Person.setPerson | 修改用户信息 | | |

表4.1.4.3-3 PersonnelList的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| PersonList.init | 语法 | Public PersonList init(String keyWord) |
| 前置条件 | 无 |
| 后置条件 | 无 |
| PersonList.getPerson | 语法 | Public Person getPerson (String personID) |
| 前置条件 | 无 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| Person.getDetail | 获取并设置详细信息 | | |
| PersonnelDataService.getHotelWorkerList | 获得hotelworker的列表信息 | | |
| PersonnelDataService.getWebMarketManList | 获得webmarketman的列表信息 | | |
| PersonnelDataService.getCustomerList | 获得Customer的列表信息 | | |

表4.1.4.3-4 Person的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Person.addPerson | 语法 | Public ResultMessage addPerson(PersonDetailVO personDetailVO) |
| 前置条件 | 无 |
| 后置条件 | 保存账户信息 |
| Person.getDetail | 语法 | Public ResultMessage getDetail () |
| 前置条件 | 无 |
| 后置条件 | 设置详细信息 |
| Person.setPerson | 语法 | Public ResultMessage setPerson (PersonDetailVO personDetailVO) |
| 前置条件 | 无 |
| 后置条件 | 保存账户信息 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| PersonnelDataService.addCustomer | 添加Customer单一化持久对象 | | |
| PersonnelDataService.addHotelWorker | 添加HotelWorker单一化持久对象 | | |
| PersonnelDataService.addWebMarketMan | 添加WebMarketMan单一化持久对象 | | |
| PersonnelDataService.getHotelworkerDetail | 获得某一hotelworker的持久化对象 | | |
| PersonnelDataService.getWebMarketManDetail | 获得某一webmarketman的持久化对象 | | |
| PersonnelDataService.getCustomerDetail | 获得某一customer的持久化对象 | | |
| PersonnelDataService. setHotelWorker | 更改HotelWorker单一化持久对象 | | |
| PersonnelDataService. setWebMarketMan | 更改webMarketMan单一化持久对象 | | |
| PersonnelDataService. setCustomer | 更改Customer单一化持久对象 | | |

#### 4.1.4.4 personnelbl业务逻辑层的动态模型

图4.1.4.4-1表明了酒店预订系统中，当酒店管理人员增加用户时，personnelbl业务逻辑处理的相关对象之间的协作。

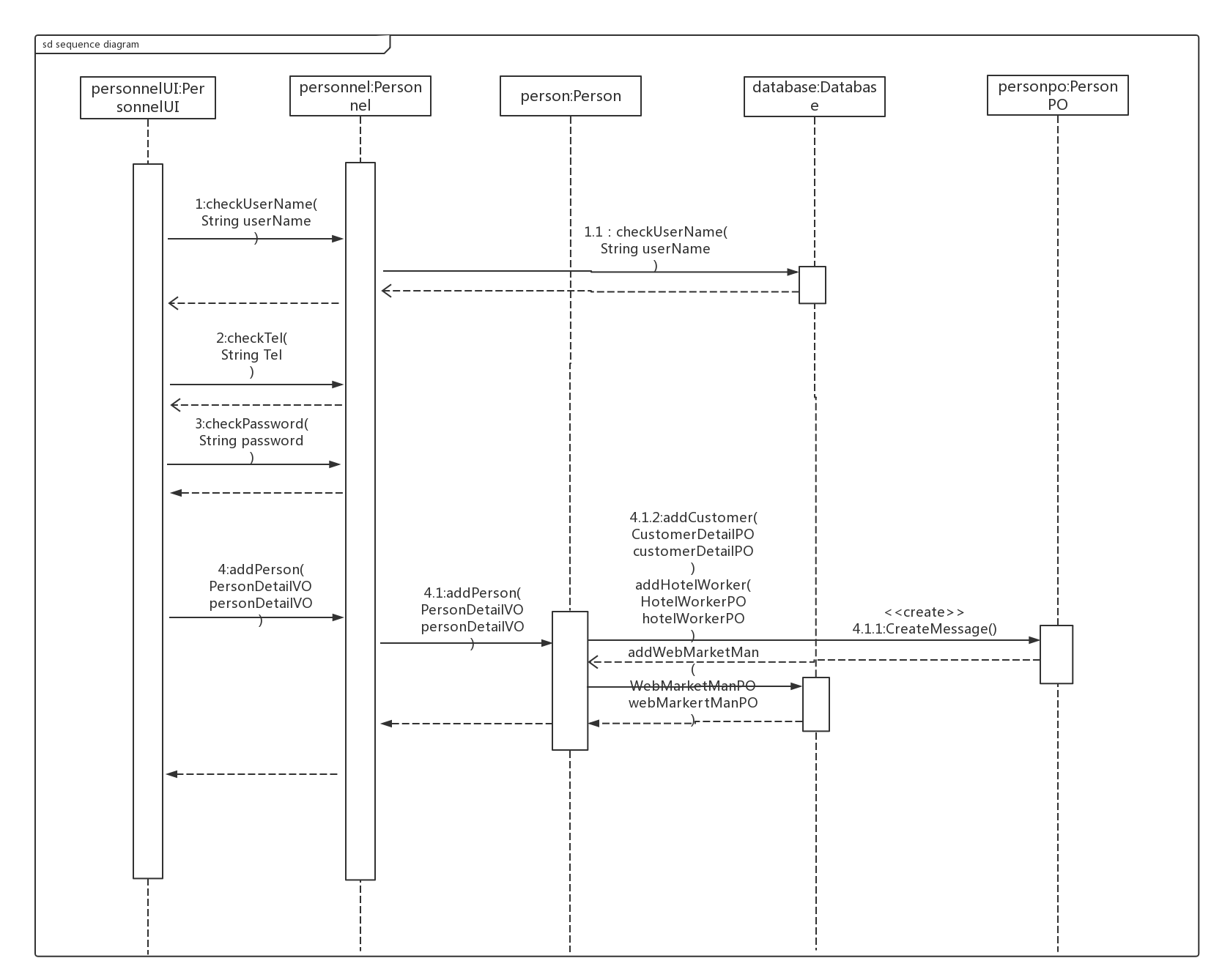


图4.1.4.4-1增加用户信息的顺序图

图4.1.4.4-2表明了酒店预订系统中，当酒店管理人员修改用户信息时，personnelbl业务逻辑处理的相关对象之间的协作。

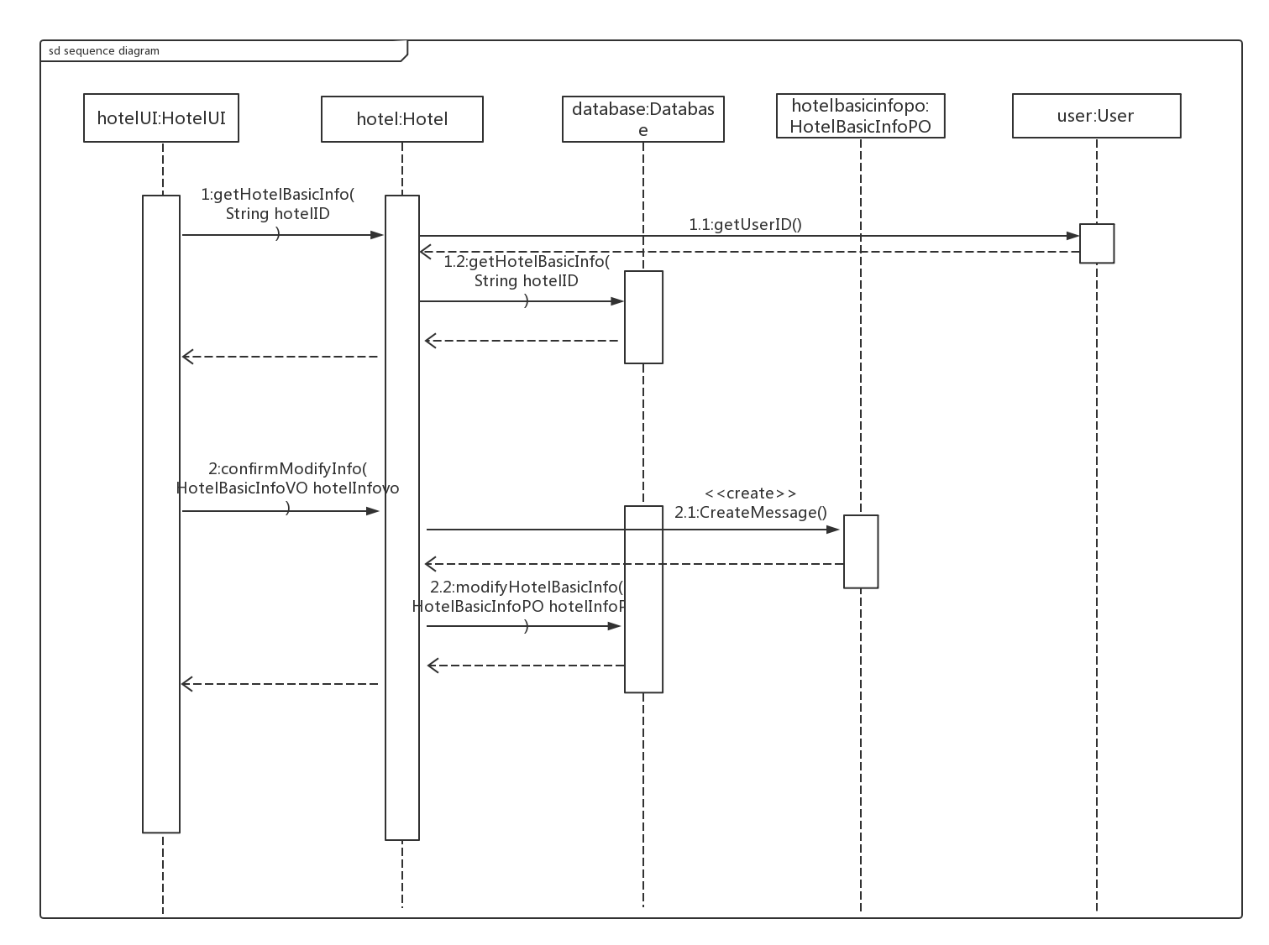


图4.1.4.4-2修改用户信息的顺序图

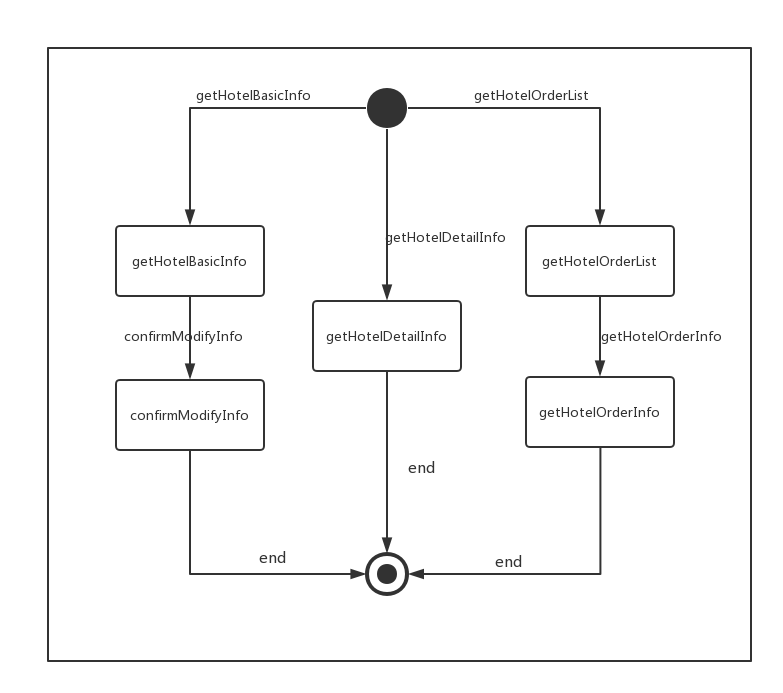
如图4.1.4.4-3所示的状态图描述了personnel对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。

图4.1.4.4-3Personnel对象状态图

#### 4.1.4.5 personnelbl业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。

### 4.1.5 searchhotelbl模块

#### 4.1.5.1 searchhotelbl模块概述

searchhotelbl 模块承担的需求见需求规格说明书文档功能需求及相关非功能需求。

searchhotelbl 模块的职责及接口参见软件体系结构描述文档。

#### 4.1.5.2 searchhotelbl整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。在展示层和业务逻辑层之间添加businesslogicservice.searchhotelblservice.SearchHotelBLService接口。在业务逻辑层和数据层之间添加dataservice.searchhoteldataservice.SearchHotelDataService接口。为了隔离业务逻辑职责和逻辑控制职责，添加SearchHotelController，这样SearchHotelController会将对用户的业务逻辑处理委托给SearchHotel对象。HotelSearchInfoPO是作为酒店搜索信息的持久性对象被添加到设计模型中去的。HotelListPO是作为酒店列表信息的持久性对象被添加到设计模型中去的。SearchHotel依赖Order和User来帮助获取酒店列表信息。

searchhotelbl模块的设计如图4．1.5.2-1所示。

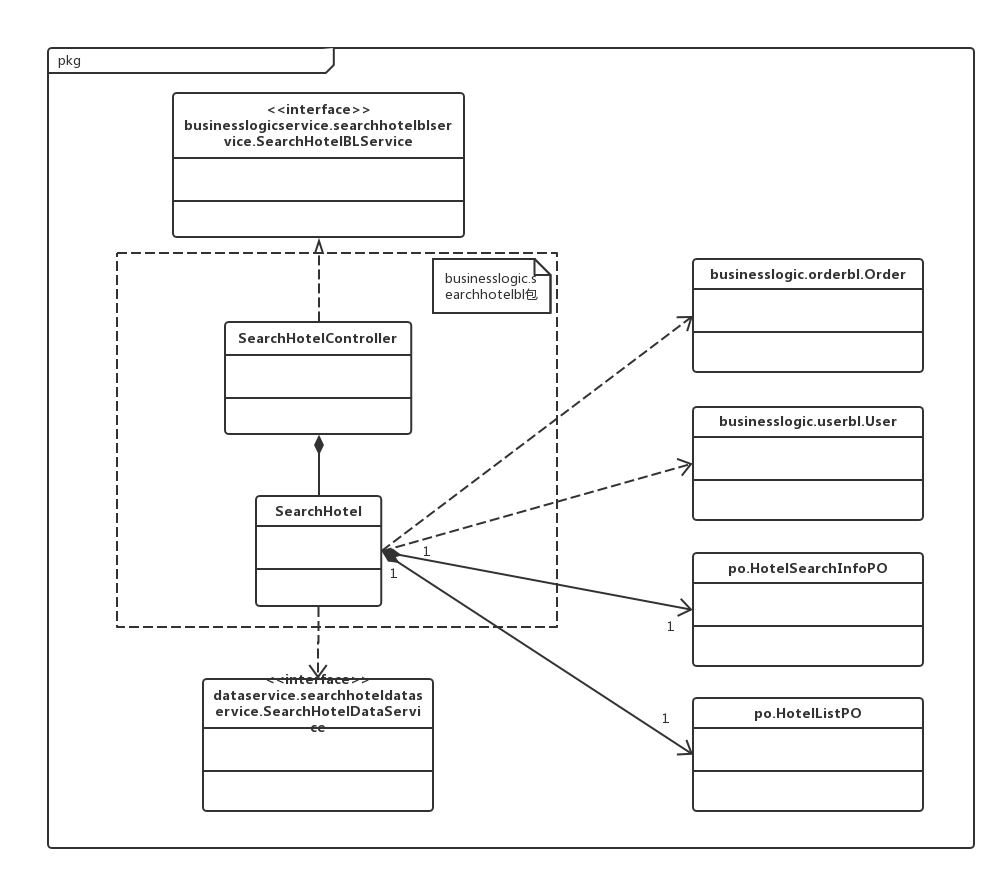


图4．1.5.2-1 searchhotelbl模块类的设计

searchhotelbl模块各个类的职责如表4．1.5.2-1所示

表4．1.5.2-1 searchhotelbl模块各个类的职责

|  |  |
| --- | --- |
| **模块** | **职责** |
| SearchHotelController | 负责实现搜索浏览酒店的服务。 |
| SearchHotel | 搜索酒店的领域模型对象，拥有酒店列表信息，可以解决搜索酒店的问题 |

#### 4.1.5.3 searchhotelbl模块内部类的接口规范

SearchHotelController和SearchHotel的接口规范如表4.1.5.3-1和表4.1.5.3-2所示

表4.1.5.3-1 SearchHotelController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| SearchHotelController. getHotelList | 语法 | Public ArrayList<HotelListVO> getHotelList（HotelSearchInfoVO vo） |
| 前置条件 | 已经创建一个SearchHotel领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| SearchHotelController. getSortedHotelList | 语法 | Public ArrayList<HotelListVO> getSortedHotelList（HotelSortType type） |
| 前置条件 | 已经创建一个SearchHotel领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| SearchHotelController. getBookedHotelList | 语法 | Public ArrayList<HotelListVO> getBookedHotelList(HotelSortType type,OrderState orderState) |
| 前置条件 | 已经创建一个SearchHotel领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | |
| SearchHotel.getHotelList | 获得酒店列表信息 | |
| SearchHotel. getSortedHotelList | 获得排序后的酒店列表信息 | |
| SearchHotel. getBookedHotelList | 获得预定过的酒店列表信息 | |

表4.1.5.3-2 SearchHotel的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| SearchHotel. getHotelList | 语法 | Public ArrayList<HotelListVO> getHotelList（HotelSearchInfoVO vo） |
| 前置条件 | 系统处于用户登录状态 |
| 后置条件 | 无 |
| SearchHotel. getSortedHotelList | 语法 | Public ArrayList<HotelListVO> getSortedHotelList（HotelSortType type） |
| 前置条件 | 系统处于用户登录状态 |
| 后置条件 | 无 |
| SearchHotel. getBookedHotelList | 语法 | Public ArrayList<HotelListVO> getBookedHotelList(HotelSortType type,OrderState orderState) |
| 前置条件 | 系统处于用户登录状态 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | |
| SearchHotelDataService.getHotelList | 获得酒店列表信息 | |
| SearchHotelDataService. getSortedHotelList | 获得排序后的酒店列表信息 | |
| SearchHotelDataService. getBookedHotelList | 获得预定过的酒店列表信息 | |
| Order.getOrderList | 获得预订过的订单列表信息 | |
| User.getUserID | 获得用户编号 | |

#### 4.1.5.4 searchhotelbl业务逻辑层的动态模型

图4.1.5.4-1表明了酒店预订系统中，当顾客查看预定过的酒店时，搜索浏览酒店业务逻辑处理的相关对象之间的协作。

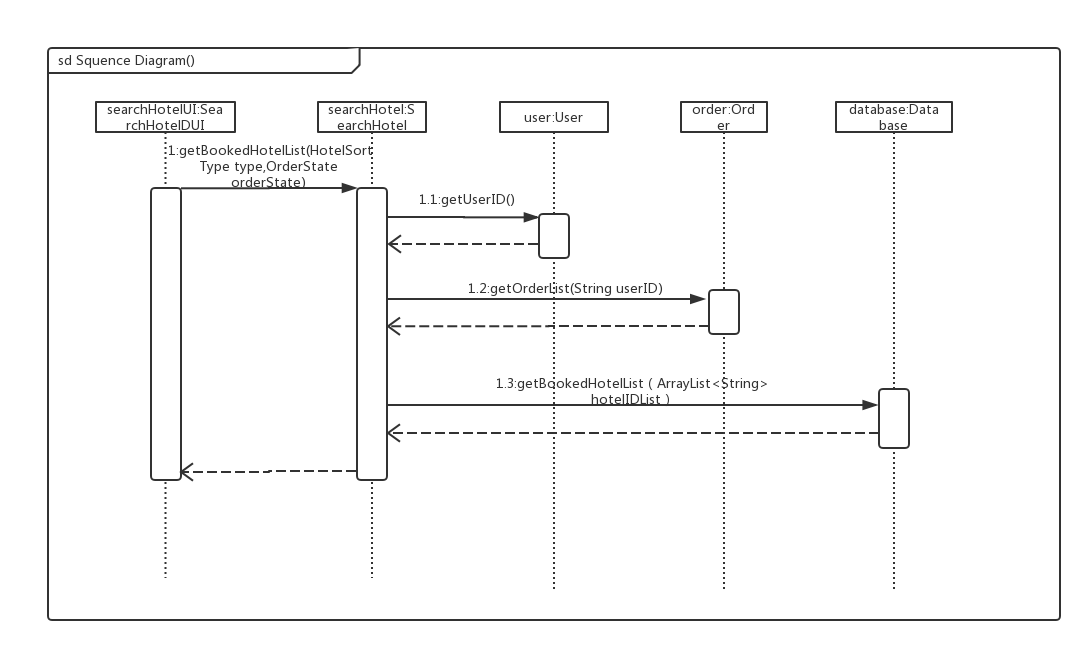


图4.1.5.4-1 查看预定过的酒店的顺序图

图4.1.5.4-2表明了酒店预订系统中，当顾客查看搜索后的酒店时，搜索浏览酒店业务逻辑处理的相关对象之间的协作。

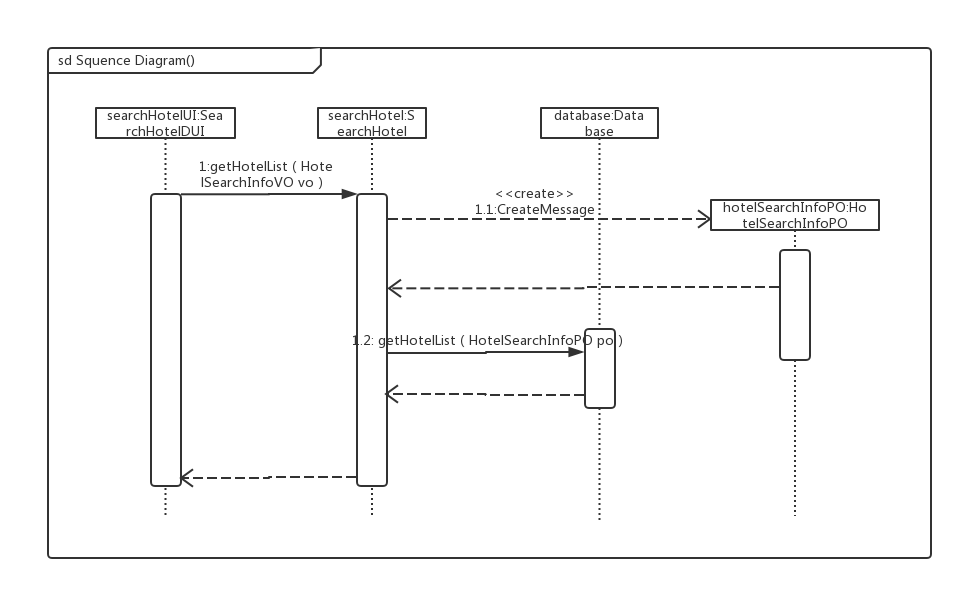


图4.1.5.4-2 查看搜索后的酒店的顺序图

如图4.1.5.4-3所示的状态图描述了SearchHotel对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。随着 UI调用，进入SearchHotel状态，之后通过列表排序进入SortedHotelList状态，也可通过输入搜索信息进入SearchedHotelList状态, 也可通过查看预定过的酒店进入BookedHotelList状态。

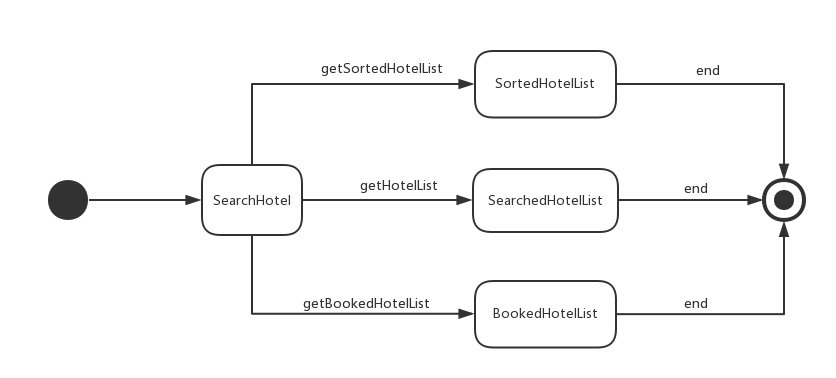


图4.1.5.4-3 SearchHotel对象状态图

#### 4.1.5.5 searchhotelbl业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。

### 4.1.6 Checkinbl模块

#### 4.1.6.1 Checkinbl模块概述

Checkinbl模块承担的需求见需求规格说明书文档功能需求及相关非功能需求。

Checkinbl模块的职责及接口参见体系结构描述文档。

#### 4.1.6.2 Checkinbl整体结构

根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了Checkinblservice, Checkindataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了CheckinController，这样CheckinController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给Checkin对象。Checkin对象实现对酒店所有入住信息的管理，委托CheckinInfo对象实现对某一入住信息的修改。CheckinInfoPO是作为酒店入住信息的持久化对象被添加到设计模型中。通过User得到用户信息，通过AvailableRoom修改可用客房信息，通过Order修改对应订单信息。

Checkinbl模块的设计如图4．1.6.2-1所示。

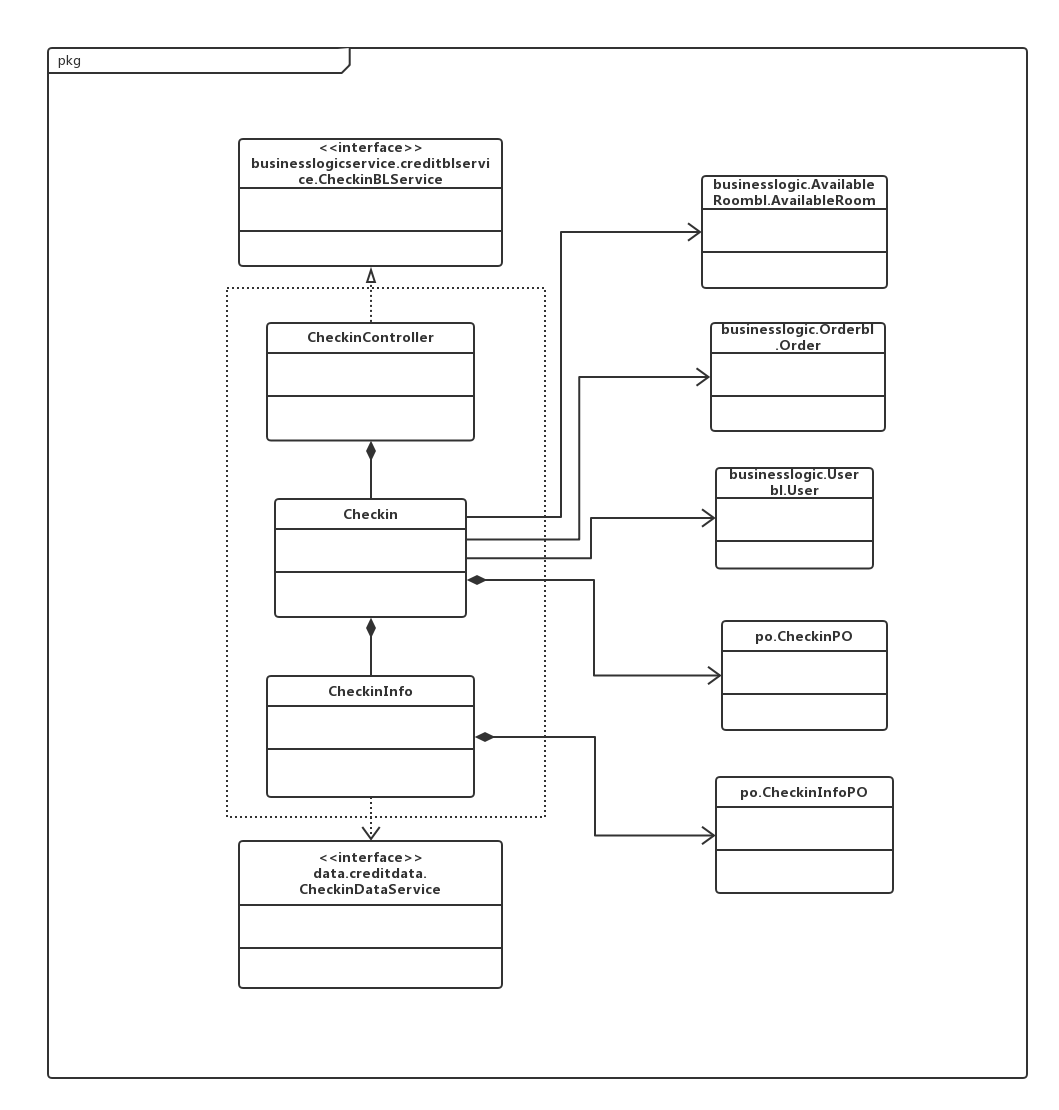


图4．1.6.2-1 Checkinbl模块类的设计

Checkinbl模块各个类的职责如表4．1.6.2-1所示

表4．1.6.2-1 Checkinbl模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| CheckinController | 负责实现酒店住房信息相关操作界面所需要的服务 |
| Checkin | 酒店住房信息的领域模型对象 |
| CheckinInfo | 每条酒店住房信息的领域模型对象 |

#### 4.1.6.3 Checkinbl模块内部类的接口规范

表4.1.6.3-1CheckinController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| CheckinController.getOrderInfo | 语法 | public OrderInfoVO getOrderInfo(String orderID) |
| 前置条件 | 订单号符合输入语法要求 |
| 后置条件 | 无 |
| CheckinController.getCheckinInfo | 语法 | public CheckinInfoVO getCheckinInfo(String orderID) |
| 前置条件 | 订单号符合输入语法要求 |
| 后置条件 | 无 |
| CheckinController.checkIDLength | 语法 | public ResultMessage checkIDLength(String ID) |
| 前置条件 | 输入顾客身份证ID |
| 后置条件 | 无 |
| CheckinController.confirmCheckinInfo | 语法 | public ResultMessage confirmCheckinInfo(CheckinInfoVO vo) |
| 前置条件 | 住房信息符合输入语法规范 |
| 后置条件 | 无 |
| CheckinController.comfirmCheckoutInfo | 语法 | public ResultMessage confirmCheckoutInfo(CheckinInfoVO vo) |
| 前置条件 | 住房信息符合输入语法规范 |
| 后置条件 | 无 |
| CheckinController.checkAvailableRoomNumber | 语法 | Public ResultMessage checkAvailableRoomNumber(BedType bedType,String number) |
| 前置条件 | 输入需要变化的可用客房数量 |
| 后置条件 | 无 |
| CheckinController.confirmAvailableRoomNumber | 语法 | public ResultMessage confirmAvailableRoomNumber(AvailableRoomNumberVO vo) |
| 前置条件 | 可用客房信息符合输入语法规范 |
| 后置条件 | 无 |
| CheckinController.getAvailableRoomInfo | 语法 | public AvailableRoomInfoVO getAvailableRoomInfo() |
| 前置条件 | 线下入住退房修改可用房间数量 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| Checkin.getOrderInfo | 根据订单号获取顾客订单信息 | | |
| Checkin.getCheckinInfo | 获取查找到的顾客住房信息 | | |
| Checkin.checkIDLength | 判断身份证格式是否正确 | | |
| Checkin.confirmCheckinInfo | 系统持久化新增顾客住房信息并自动更新订单信息。 | | |
| Checkin.comfirmCheckoutInfo | 系统持久化修改顾客住房信息并自动更新订单信息和可用客房信息 | | |
| Checkin.checkAvailableRoomNumber | 判断输入可用客房数量是否超过该类型客房总数 | | |
| Checkin.confirmAvailableRoomNumber | 系统持久化修改该酒店可用客房信息 | | |
| Checkin.getAvailableRoomInfo | 系统返回该酒店的可用房间信息 | | |

表4.1.6.3-2 Checkin的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Checkin.getOrderInfo | 语法 | public OrderInfoVO getOrderInfo(String orderID) |
| 前置条件 | 订单号符合输入语法要求 |
| 后置条件 | 无 |
| Checkin.getCheckinInfo | 语法 | public CheckinInfoVO getCheckinInfo(String orderID) |
| 前置条件 | 订单号符合输入语法要求 |
| 后置条件 | 无 |
| Checkin.checkIDLength | 语法 | public ResultMessage checkIDLength(String ID) |
| 前置条件 | 输入顾客身份证ID |
| 后置条件 | 无 |
| Checkin.confirmCheckinInfo | 语法 | public ResultMessage confirmCheckinInfo(CheckinInfoVO vo) |
| 前置条件 | 住房信息符合输入语法规范 |
| 后置条件 | 系统持久化新增顾客住房信息并自动更新订单信息。 |
| Checkin.comfirmCheckoutInfo | 语法 | public ResultMessage confirmCheckoutInfo(CheckinInfoVO vo) |
| 前置条件 | 住房信息符合输入语法规范 |
| 后置条件 | 系统持久化修改顾客住房信息并自动更新订单信息和可用客房信息。 |
| Checkin.checkAvailableRoomNumber | 语法 | Public ResultMessage checkAvailableRoomNumber(BedType bedType,String number) |
| 前置条件 | 输入需要变化的可用客房数量 |
| 后置条件 | 无 |
| Checkin.confirmAvailableRoomNumber | 语法 | public ResultMessage confirmAvailableRoomNumber(AvailableRoomNumberVO vo) |
| 前置条件 | 可用客房信息符合输入语法规范 |
| 后置条件 | 系统持久化修改该酒店可用客房信息 |
| Checkin.getAvailableRoomInfo | 语法 | public AvailableRoomInfoVO getAvailableRoomInfo() |
| 前置条件 | 线下入住退房修改可用房间数量 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| CheckinDataService. getCheckinInfo | 根据订单号获取顾客住房信息 | | |
| CheckinDataService. addCheckinInfo | 增加顾客住房信息持久化对象 | | |
| CheckinDataService. modifyCheckinInfo | 修改顾客住房信息持久化对象 | | |
| User.getUserID | 获取当前登录的用户信息 | | |
| Order.getOrderInfo | 根据订单号获取订单信息 | | |
| Order.setCheckInTime | 根据时间修改订单上实际入住时间的持久化对象 | | |
| Order.setCheckOutTime | 根据时间修改订单上实际退房时间的持久化对象 | | |
| AvailableRoom.setAvailableRoomNumber | 修改可用客房数量的持久化对象 | | |
| AvailableRoom.getAvailableRoomInfo | 根据酒店编号获取可用客房数量 | | |

### 4.1.6.4 Checkinbl业务逻辑层的动态模型

图4.1.6.4-1表明了顾客线上退房时，系统更新酒店住房记录的业务逻辑的相关对象的协作。

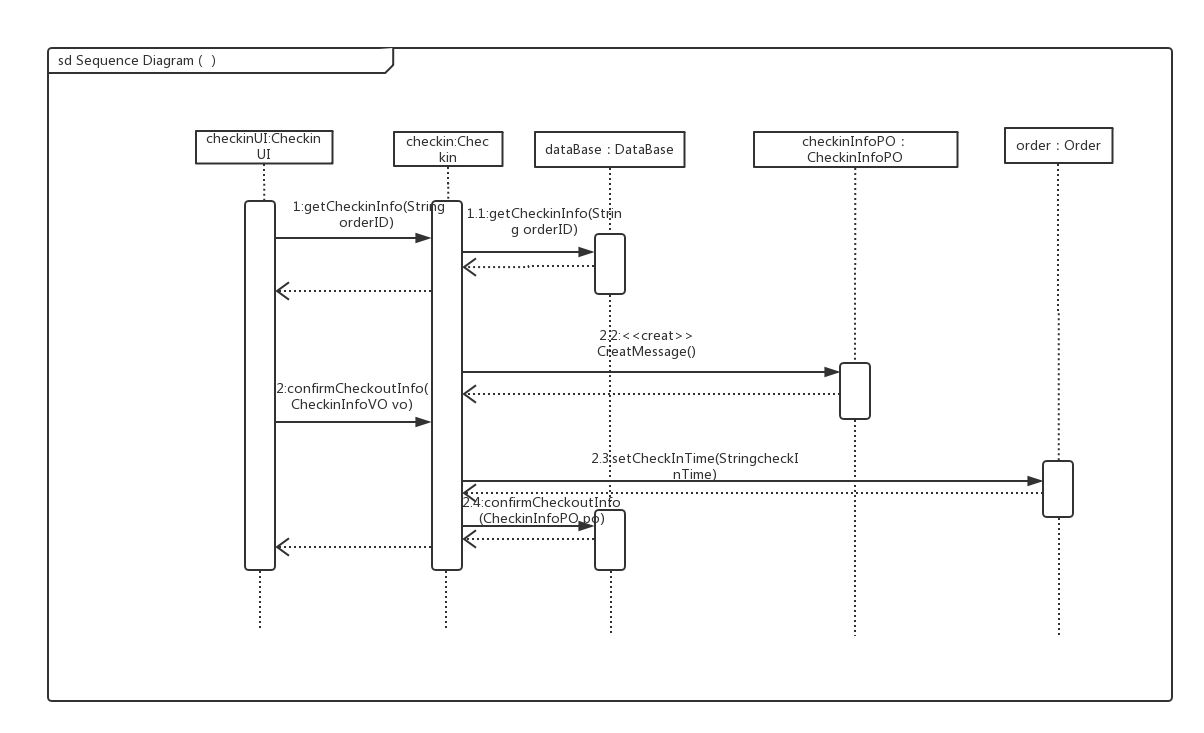


图4.1.6.4-1线上入住退房的顺序图

图4.1.6.4-2表明了顾客线下退房时，系统修改可用客房信息的业务逻辑的相关对象的协作。

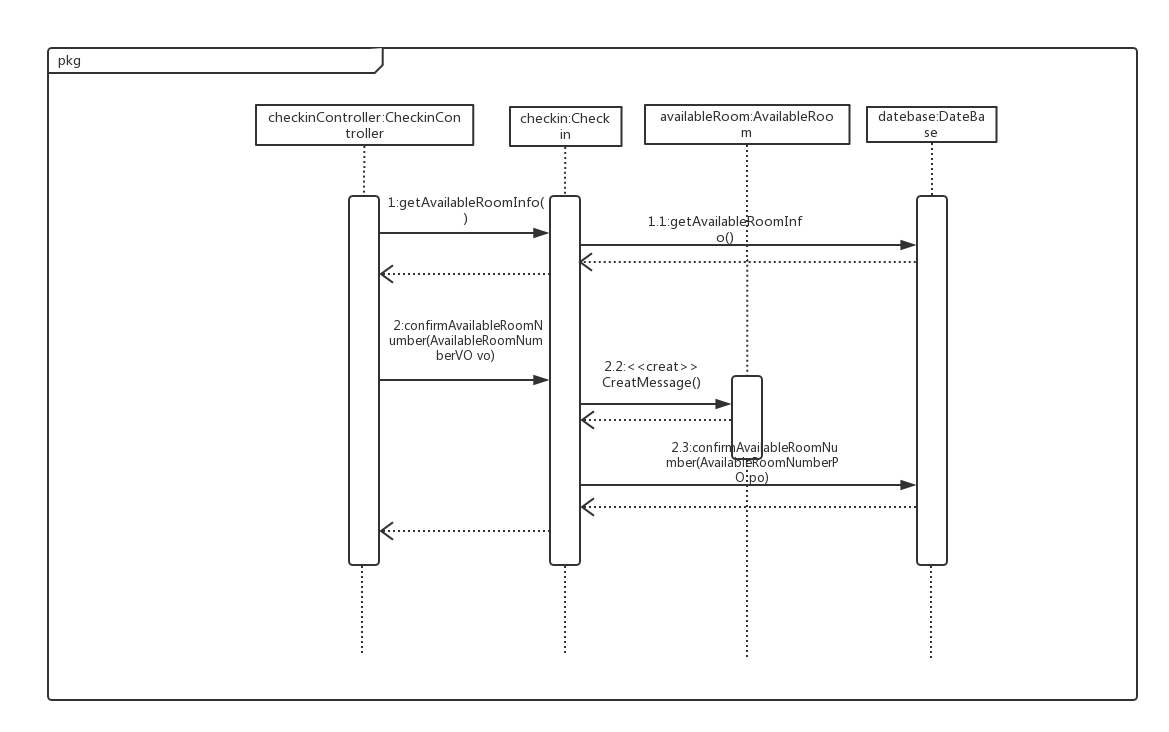


图4.1.6.4-2线下入住退房的顺序图

图4.1.6.4-3所示的状态图描述了Checkin对象的生存期间的状态序列、引起状态转移而伴随的动作。

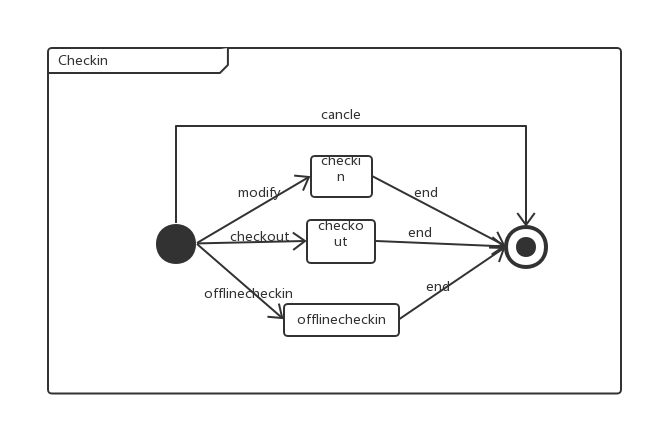


图4.1.6.4-3Checkin对象状态图

#### 4.1.6.5 Checkinbl业务逻辑层的设计原理

利用委托式控制风格，每个页面需要访问的业务逻辑有各自的控制器委托给不同的领域对象。

### 4.1.7 AvailableRoombl模块

#### 4.1.7.1 AvailableRoombl模块概述

AvailableRoombl模块承担的需求见需求规格说明书文档功能需求及相关非功能需求。

AvailableRoombl模块的职责及接口参见体系结构描述文档。

#### 4.1.7.2 AvailableRoombl整体结构

根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了AvailableRoomblservice, AvailableRoomdataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了AvailableRoomController，这样AvailableRoomController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给AvailableRoom对象。

AvailableRoom对象实现对酒店所有类型房间的操作，委SingleAvailableRoom实现对具体某种房间类型的操作。AvailableRoomPO是作为可用客房信息的持久化对象被添加到设计模型中，存储酒店客房所有信。SingleAvailableRoomPO是作为某种类型可用客房信息的持久化对象被添加到设计模型中，存储酒店每种客房可用的房间数信息。通过User得到当前用户信息。

AvailableRoombl模块的设计如图4．1.7.2-1所示。

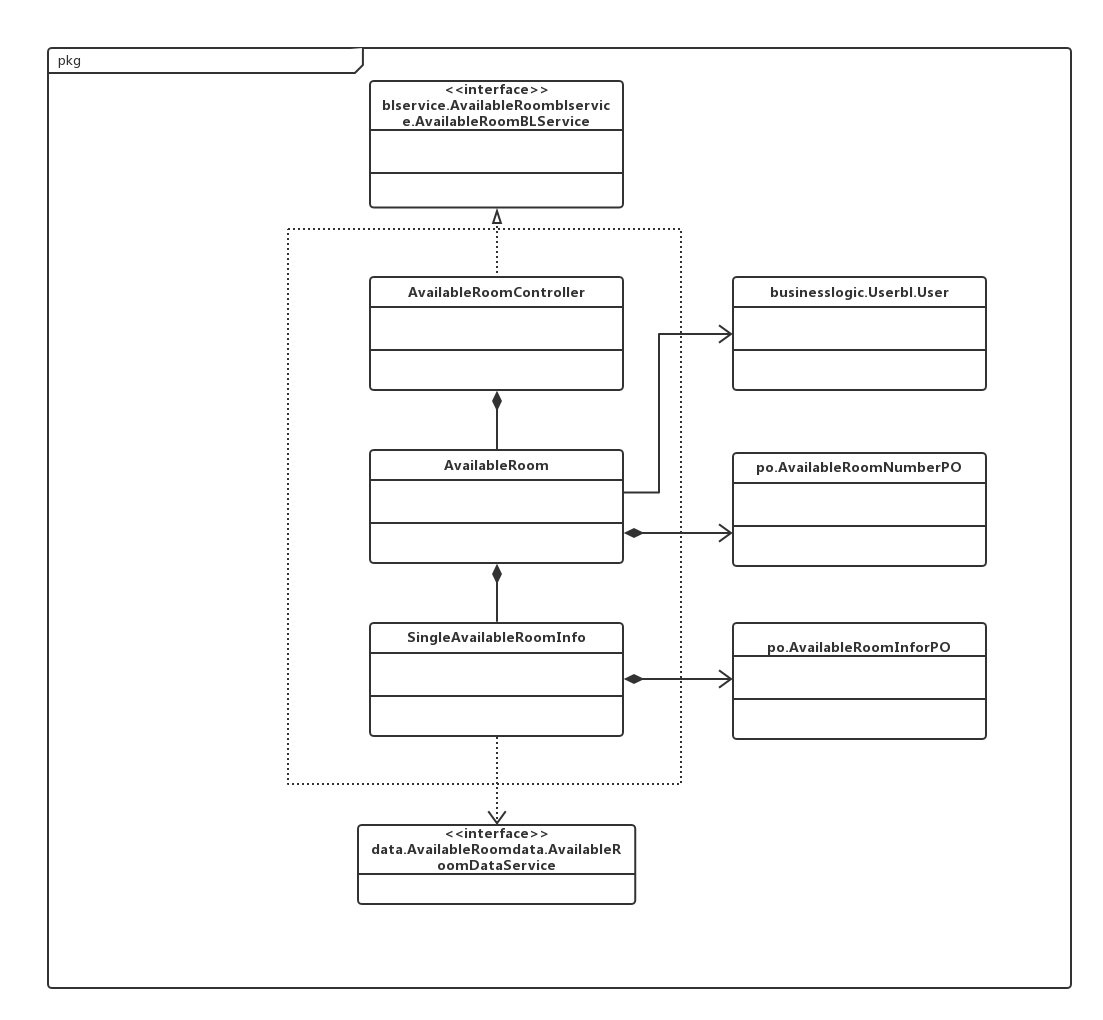


图4．1.7.2-1AvailableRoombl模块类的设计

availablebl模块各个类的职责如表4．1.7.2-2所示

表4．1.7.2-2AvailableRoombl模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| AvailableRoomController | 负责实现酒店可用客房信息相关操作界面所需要的服务 |
| AvailableRoom | 酒店可用客房信息的领域模型对象 |
| SingleAvailableRoomInfo | 某种可用客房信息的领域模型对象 |

#### 4.1.7.3 AvailableRoombl模块内部类的接口规范

表4.1.7.3-1AvailableRoomController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| AvailableRoomController. getAvailablerRoomInfo | 语法 | public AvailableRoomInfoVO getAvailableRoomInfo(String hotelID) |
| 前置条件 | 酒店编号符合输入语法要求 |
| 后置条件 | 无 |
| AvailableRoomController. confirmAvailableRoomInfo | 语法 | public ResultMessage confirmAvailableRoomInfo (AvailableRoomInfoVO vo) |
| 前置条件 | 房间信息符合输入语法要求，房间数量为正数 |
| 后置条件 | 无 |
| AvailableRoomController.setAvailableRoomNumber | 语法 | Public ResultMessage setAvailableRoomNumber( AvailableRoomNumberVO vo) |
| 前置条件 | 可用房间数量符合输入语法要求 |
| 后置条件 | 无 |
| AvailableRoomController.setBestPrice | 语法 | public ResultMessage setBestPrice (double discount) |
| 前置条件 | 折扣值符合输入语法规范 |
| 后置条件 | 无 |
| AvailableRoomController.getRoomPrice | 语法 | public double getRoomPrice (String hotelID,BedType bedType) |
| 前置条件 | 输入的酒店编号和房间类型合法且存在 |
| 后置条件 | 无 |
| AvailableRoomController. checkAvailableRoomNumber | 语法 | public ResultMessage checkAvailableRoomNumber (AvailableRoomNumberVO vo) |
| 前置条件 | 生成订单时，系统自动更新线上可用客房数量 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| AvailableRoom. getAvailablerRoomInfo | 根据酒店编号获取酒店可用客房信息 | | |
| AvailableRoom. confirmAvailableRoomInfo | 系统持久化新增酒店房间信息 | | |
| AvailableRoom.setAvailableRoomNumber | 系统持久化更新可用房间信息 | | |
| AvailableRoom.setBestPrice | 系统持久化更新各种房型的最低价格 | | |
| AvailableRoom.getRoomPrice | 系统返回该房型价格。 | | |
| AvailableRoom. checkAvailableRoomNumber | 判断是否有足够的可用客房 | | |

表4.1.7.3-2AvailableRoom的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| AvailableRoom. getAvailablerRoomInfo | 语法 | public AvailableRoomInfoVO getAvailableRoomInfo(String hotelID) |
| 前置条件 | 酒店编号符合输入语法要求 |
| 后置条件 | 无 |
| AvailableRoom. confirmAvailableRoomInfo | 语法 | public ResultMessage confirmAvailableRoomInfo (AvailableRoomInfoVO vo) |
| 前置条件 | 房间信息符合输入语法要求，房间数量为正数 |
| 后置条件 | 系统持久化新增酒店房间信息 |
| AvailableRoom.setAvailableRoomNumber | 语法 | Public ResultMessage setAvailableRoomNumber( AvailableRoomNumberVO vo) |
| 前置条件 | 可用房间数量符合输入语法要求 |
| 后置条件 | 系统持久化更新可用房间信息 |
| AvailableRoom.setBestPrice | 语法 | public ResultMessage setBestPrice (double discount) |
| 前置条件 | 折扣值符合输入语法规范 |
| 后置条件 | 系统持久化更新各种房型的最低价格 |
| AvailableRoom.getRoomPrice | 语法 | public double getRoomPrice (String hotelID,BedType bedType) |
| 前置条件 | 输入的酒店编号和房间类型合法且存在 |
| 后置条件 | 无 |
| AvailableRoom. checkAvailableRoomNumber | 语法 | public ResultMessage checkAvailableRoomNumber (AvailableRoomNumberVO vo) |
| 前置条件 | 生成订单时，系统自动更新线上可用客房数量 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| AvailableRoomDataService. getAvailableRoomInfo | 根据酒店编号获取酒店可用客房信息 | | |
| AvailableRoomDataService. addAvailableRoomInfo | 增加酒店可用客房信息持久化对象 | | |
| AvailableRoomDataService. modifyAvailableRoomInfo | 修改酒店可用客房信息持久化对象 | | |
| AvailableRoomDataService. setAvailableRoomNumber | 修改酒店可用客房数量持久化对象 | | |
| AvailableRoomDataService. getRoomPrice | 根据酒店编号和房间类型获取房间原始价格 | | |
| AvailableRoomDataService.  setBestPrice | 修改可用客房的最低价格 | | |
| User.getUserID | 获取当前登录的用户信息 | | |

#### 4.1.7.4 AvailableRoombl业务逻辑层的动态模型

图4.1.7.4-1表明了修改酒店房间信息时，系统修改酒店可用客房信息的业务逻辑的相关对象的协作。

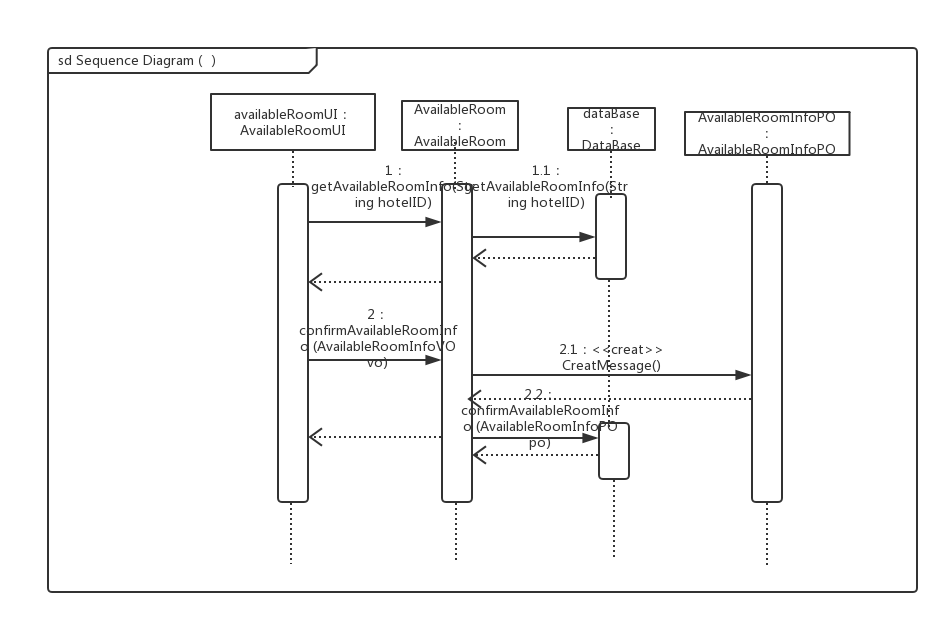


图4.1.7.4-1修改酒店房间信息的顺序图

图4.1.7.4-2表明了顾客线下退房时，系统修改可用酒店客房信息的业务逻辑的相关对象的协作。

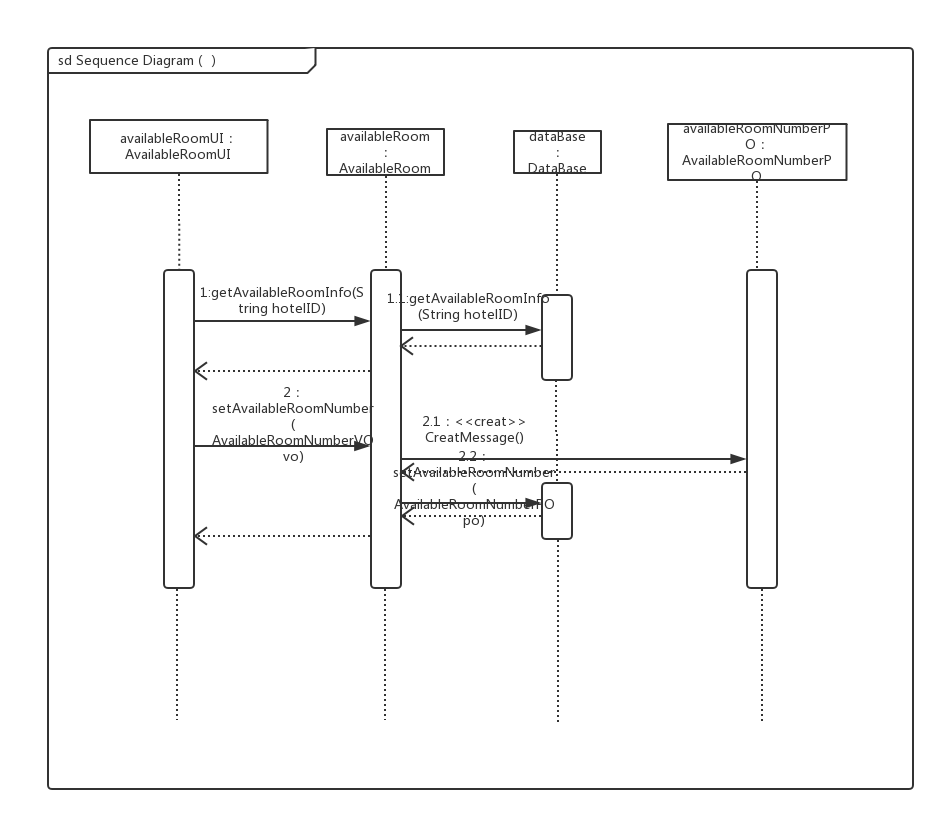


图4.1.7.4-2线下入住退房的顺序图

图4.1.7.4-3所示的状态图描述了Available对象的生存期间的状态序列、引起状态转移而伴随的动作。

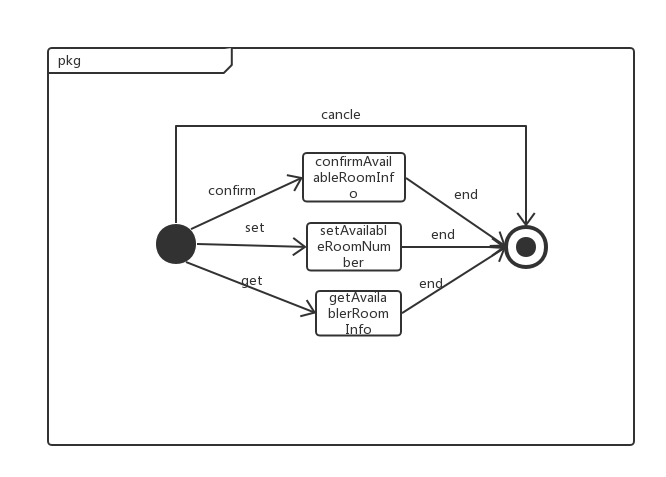


图4.1.7.4-3AvailableRoom对象状态图

#### 4.1.7.5AvailableRoombl业务逻辑层的设计原理

利用委托式控制风格，每个页面需要访问的业务逻辑有各自的控制器委托给不同的领域对象。

### 4.1.8Creditbl模块

#### 4.1.8.1 Creditbl模块概述

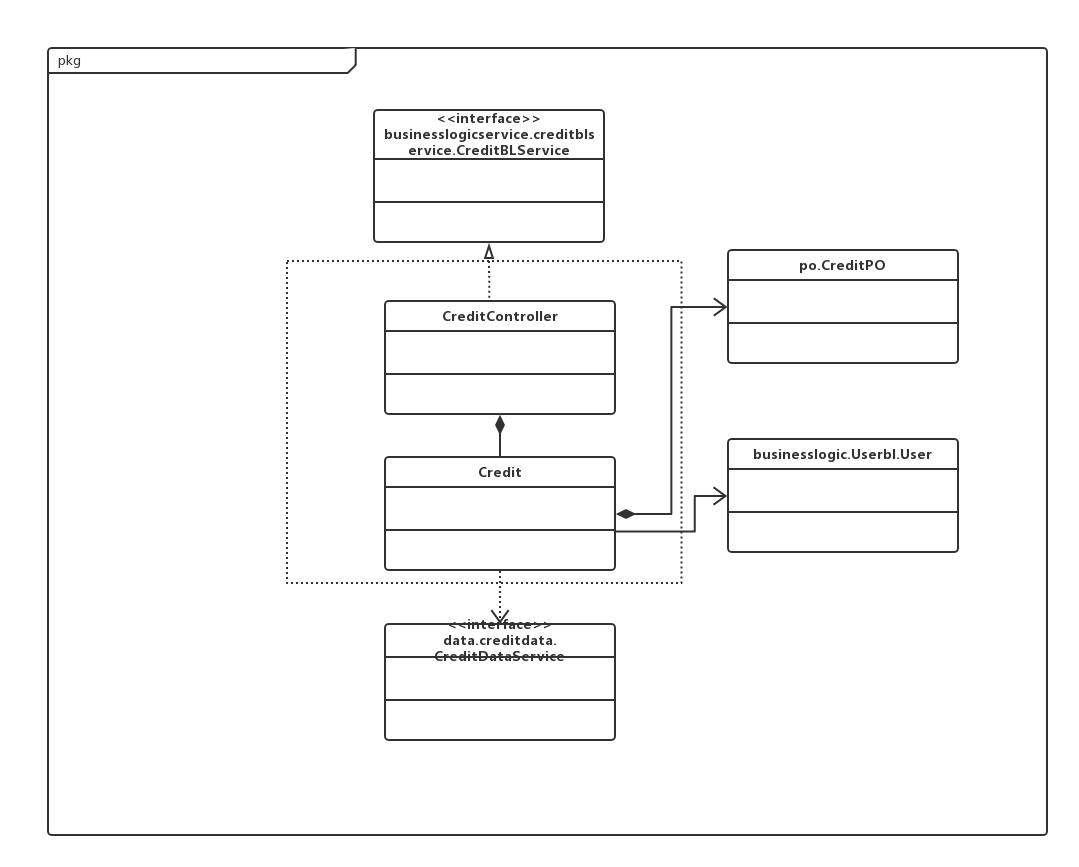
Creditbl模块承担的需求见需求规格说明书文档功能需求及相关非功能需求。

Creditbl模块的职责及接口参见体系结构描述文档。

#### 4.1.8.2 Creditbl整体结构

根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了Creditblservice, Creditdataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了CreditController，这样CreditController 将会将信用管理相关的业务逻辑职责和逻辑控制委托给Credit对象。Credit对象实现更改顾客信用值并增加信用记录的操作。CreditPO是作为信用信息的持久化对象被添加到设计模型中，存储顾客所有信用记录。通过User得到当前用户信息。

Creditbl模块的设计如图4．1.8.2-1所示。

图4．1.8.2-1Creditbl模块类的设计

Creditbl模块各个类的职责如表4．1.8.2-1所示

表4．1.8.2-1 Creditbl模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| CreditController | 负责管理信用相关操作界面所需要的服务 |
| Credit | 信用信息的领域模型对象 |

#### 4.1.8.3 Creditbl模块内部类的接口规范

表4.1.8.3-1 CreditController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| CreditController. getCreditInfo | 语法 | public CreditInfoVO getCreditInfo (String customerID) |
| 前置条件 | 顾客编号符合输入语法要求 |
| 后置条件 | 无 |
| CreditController. addCredit | 语法 | public ResultMessage addCredit (CreditVO vo) |
| 前置条件 | 无 |
| 后置条件 | 无 |
| CreditController. cutCredit | 语法 | Public ResultMessage cutCredit (CreditVO vo) |
| 前置条件 | 无 |
| 后置条件 | 无 |
| CreditController. confirmCreditDeposit | 语法 | public ResultMessage confirmCreditDeposit (double money,String customerName) |
| 前置条件 | 充值金额符合输入语法规范 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| Credit. getCreditInfo | 获得该顾客的信用记录 | | |
| Credit. addCredit | 系统持久化新增一条顾客的信用记录并更改顾客的信用值 | | |
| Credit. cutCredit | 系统持久化新增一条顾客的信用记录并更改顾客的信用值 | | |
| Credit. confirmCreditDeposit | 系统持久化新增一条顾客的信用记录并更改顾客的信用值 | | |

表4.1.8.3-2Credit的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Credit. getCreditInfo | 语法 | public CreditInfoVO getCreditInfo (String customerID) |
| 前置条件 | 顾客编号符合输入语法要求 |
| 后置条件 | 无 |
| Credit. addCredit | 语法 | public ResultMessage addCredit (CreditVO vo) |
| 前置条件 | 无 |
| 后置条件 | 系统持久化新增一条顾客的信用记录并更改顾客的信用值 |
| Credit. cutCredit | 语法 | Public ResultMessage cutCredit (CreditVO vo) |
| 前置条件 | 无 |
| 后置条件 | 系统持久化新增一条顾客的信用记录并更改顾客的信用值 |
| Credit. confirmCreditDeposit | 语法 | public ResultMessage confirmCreditDeposit (double money,String customerName) |
| 前置条件 | 充值金额符合输入语法规范 |
| 后置条件 | 系统持久化新增一条顾客的信用记录并更改顾客的信用值 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| CreditDataService.setCredit | 系统持久化新增一条顾客的信用记录并更改顾客的信用值 | | |
| CreditDataService. CreditInfoPO getCreditInfo | 系统返回顾客的信用记录 | | |
| User.getUserID | 获取当前登录的用户信息 | | |

#### 4.1.8.4Creditbl业务逻辑层的动态模型

图4.1.8.4-1表明了管理信用时，系统修改客户信用信息的业务逻辑的相关对象的协作。

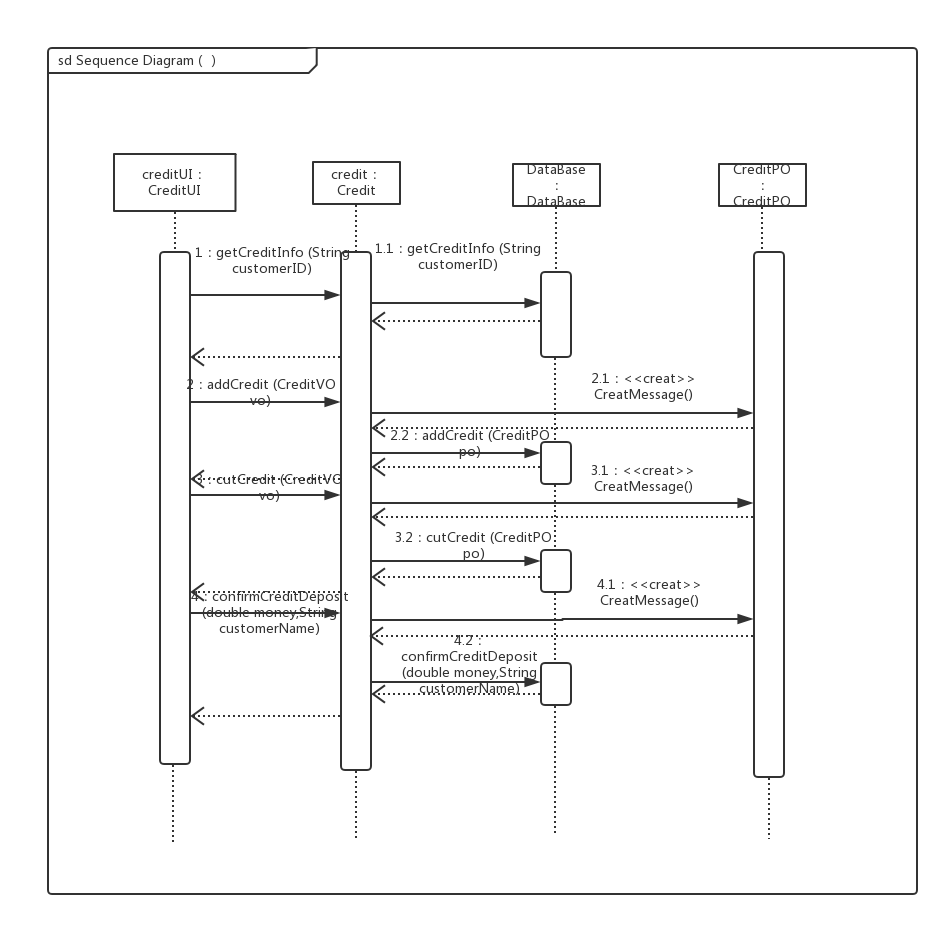


图4.1.8.4-1管理信用的顺序图

图4.1.8.4-2所示的状态图描述了Credit对象的生存期间的状态序列、引起状态转移而伴随的动作。

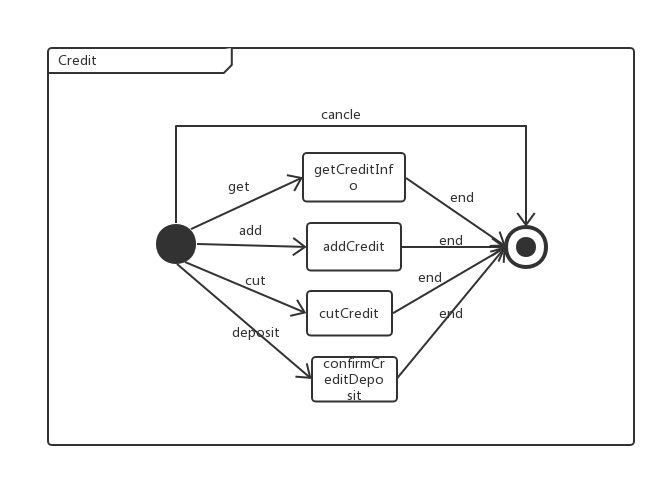


图4.1.8.4-2Credit对象状态图

#### 4.1.8.5Creditbl业务逻辑层的设计原理

利用委托式控制风格，每个页面需要访问的业务逻辑有各自的控制器委托给不同的领域对象。

### 4.1.9 hotelbl模块

#### 4.1.9.1 hotelbl模块概述

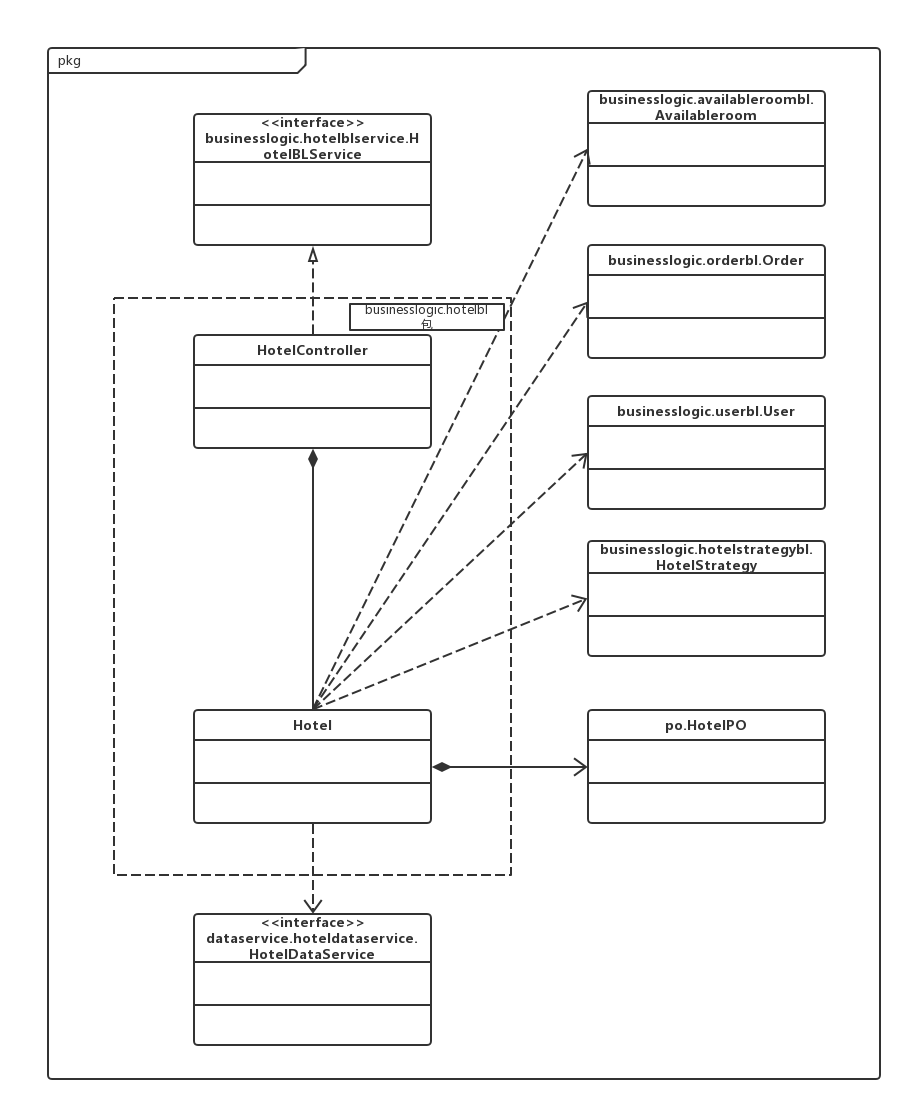
hotelbl模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

hotelbl模块的职责和接口参见系统结构描述文档内对该模块的描述。

#### 4.1.9.2 hotelbl整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。在展示层和业务逻辑层之间添加HotelblService接口。在业务逻辑层和数据层之间添加HoteldataService接口。为了隔离业务逻辑职责和逻辑控制职责，添加HotelController，这样HotelController会将对用户管理的业务逻辑处理委托给Hotel对象。HotelPO是作为酒店信息的持久性对象被添加到设计模型中去的。

hotelbl模块的设计如图4．1.9.2-1所示。



4．1.9.2-1hotelbl模块类的设计

hotelbl模块各个类的职责如表4．1.9.2-2所示

|  |  |
| --- | --- |
| 类 | 职责 |
| HotelController | 负责实现酒店界面相关操作界面所需  要的服务 |
| Hotel | 酒店的领域模型对象 |

表4．1.9.2-2 hotelbl模块各个类的职责

#### 4.1.9.3 hotelbl模块内部类的接口规范

HotelController和Hotel的结构规范如表4.1.9.3-1和4.1.9.3-2所示

表4.1.9.3-1HotelController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| HotelController.getHotelBasicInfo | 语法 | public HotelBasicInfoVO getHotelBasicInfo(String hotelID) |
| 前置条件 | 已经创建一个Hotel领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| HotelController.getHotelDetailInfo | 语法 | public HotelDetailInfoVO getHotelDetailInfo(String hotelID,String customerID) |
| 前置条件 | 已经创建一个Hotel领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| HotelController.checkTel | 语法 | public ResultMessage checkTel(String telNum) |
| 前置条件 | 已经创建一个Hotel领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| HotelController.confirmModifyInfo | 语法 | public ResultMessage confirmModifyInfo(HotelBasicInfoVO hotelInfovo) |
| 前置条件 | 已经创建一个Hotel领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| HotelController.getHotelOrderList | 语法 | public ArrayList<HotelOrderVO> getHotelOrder(String hotelID) |
| 前置条件 | 已经创建一个Hotel领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| HotelController.getHotelOrderInfo | 语法 | public HotelOrderInfoVO getHotelOrderInfo(String orderID) |
| 前置条件 | 已经创建一个Hotel领域对象，并且输入符合输入规则 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| Hotel.getHotelBasicInfo | 获得酒店基础信息 | | |
| Hotel.modifyHotelBasicInfo | 更新酒店基础信息 | | |
| Hotel.checkTel | 检查联系方式的格式 | | |
| Hotel.addHotelBasicInfo | 初始化酒店基础信息 | | |
| Hotel.addRemarkInfo | 更新酒店评价信息 | | |
| Hotel.setBestPrice | 更新酒店的最低价格 | | |
| Hotel.getAvailableRoomInfo | 获得酒店可用客房信息 | | |
| Hotel.getOrderList | 获得该酒店的订单列表信息 | | |
| Hotel.getOrderInfo | 获得该酒店某一订单的详细信息 | | |
| Hotel.getHotelStrategy | 获得酒店优惠策略 | | |

表4.1.9.3-2Hotel的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Hotel.getHotelBasicInfo | 语法 | public HotelBasicInfoVO getHotelBasicInfo(String hotelID) |
| 前置条件 | 酒店工作人员身份已经被识别 |
| 后置条件 | 无 |
| Hotel.getHotelDetailInfo | 语法 | public HotelDetailInfoVO getHotelDetailInfo(String hotelID,String customerID) |
| 前置条件 | 顾客已登入 |
| 后置条件 | 无 |
| Hotel.checkTel | 语法 | public ResultMessage checkTel(String telNum) |
| 前置条件 | 联系方式已被输入 |
| 后置条件 | 无 |
| Hotel.confirmModifyInfo | 语法 | public ResultMessage confirmModifyInfo(HotelBasicInfoVO hotelInfovo) |
| 前置条件 | 酒店工作人员身份已经被识别并且联系方式格式符合要求 |
| 后置条件 | 系统更新数据库中的酒店信息 |
| Hotel.getHotelOrderList | 语法 | public ArrayList<HotelOrderVO> getHotelOrderList(String hotelID) |
| 前置条件 | 酒店工作人员身份已经被识别 |
| 后置条件 | 无 |
| Hotel.getHotelOrderInfo | 语法 | public HotelOrderInfoVO getHotelOrderInfo(String orderID) |
| 前置条件 | 酒店工作人员身份已经被识别 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| HotelDataService.getHotelBasicInfo | 获得酒店基础信息 | | |
| HotelDataService.modifyHotelBasicInfo | 更新酒店基础信息 | | |
| HotelDataService.addHotelBasicInfo | 初始化酒店基础信息 | | |
| HotelDataService.addRemarkInfo | 更新酒店评价信息 | | |
| HotelDataService.setBestPrice | 更新酒店的最低价格 | | |
| Availableroom.getAvailableRoomInfo | 获得酒店可用客房信息 | | |
| Order.getOrderList | 获得该酒店的订单列表信息 | | |
| Order.getOrderInfo | 获得该酒店某一订单的详细信息 | | |
| User.getUserID | 获得当前用户信息（酒店工作人员） | | |
| HotelStrategy.getHotelStrategy | 获得酒店优惠策略 | | |

#### 4.1.9.4 hotelbl业务逻辑层的动态模型

图4.1.9.4-1表明了酒店预订系统中，当顾客查看酒店信息时，hotelbl业务逻辑处理的相关对象之间的协作。

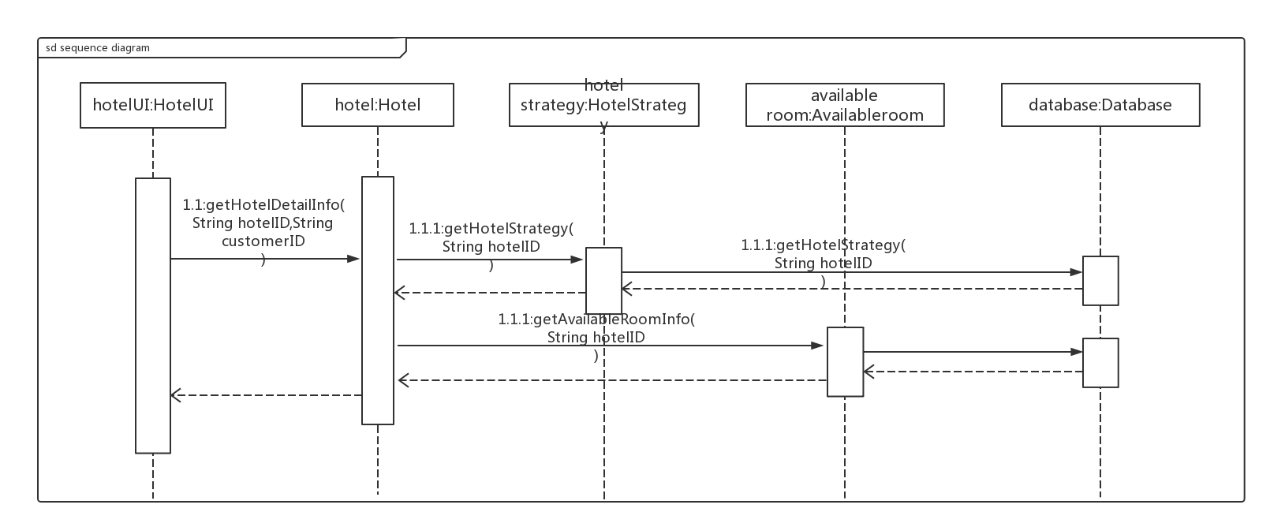


图4.1.9.4-1 顾客查看酒店信息的顺序图

图4.1.9.4-2表明了酒店预订系统中，当酒店工作人员修改酒店信息时，hotelbl业务逻辑处理的相关对象之间的协作。

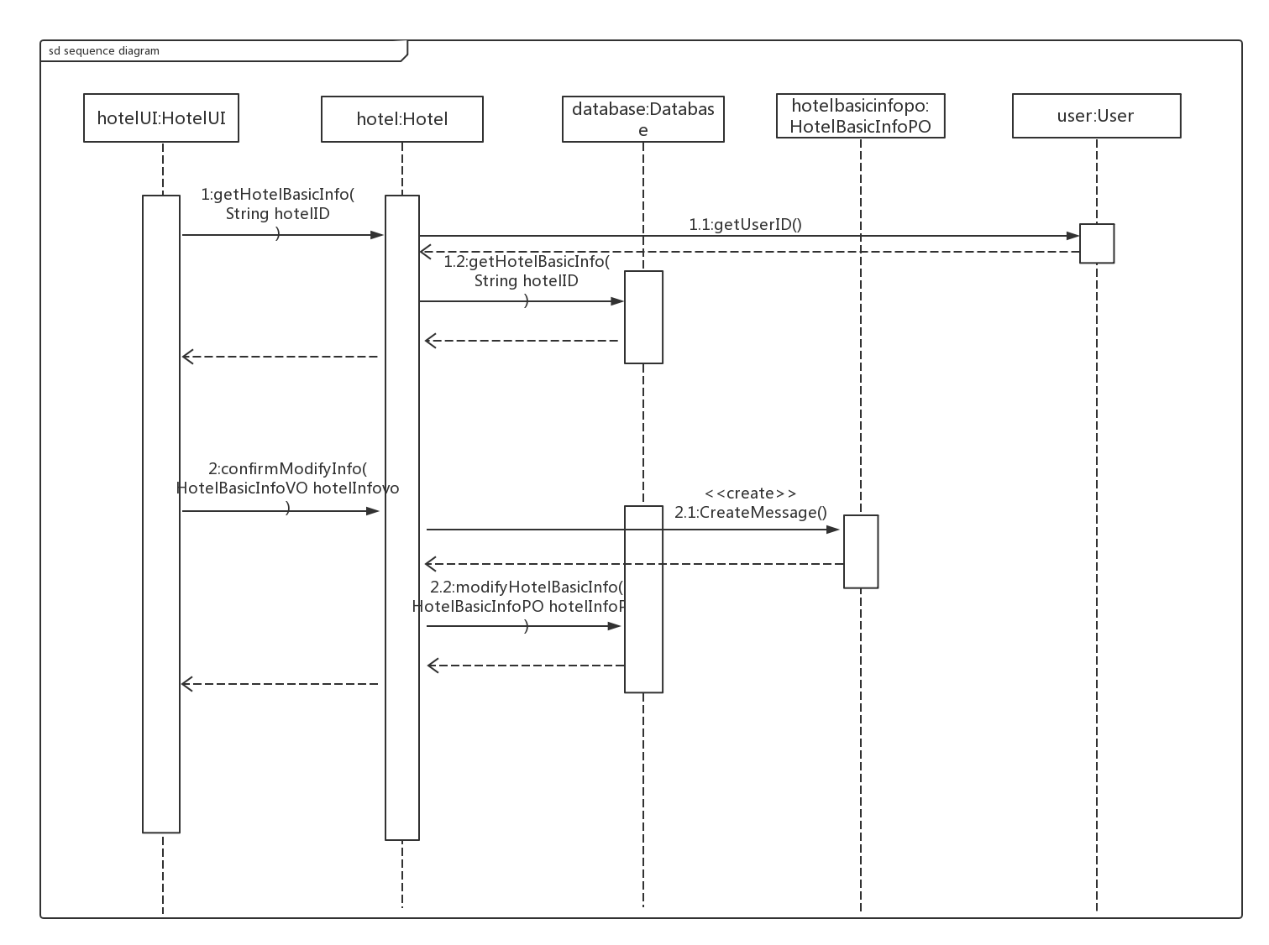


图4.1.9.4-2 酒店工作人员修改酒店信息的顺序图

图4.1.9.4-3表明了酒店预订系统中，当酒店工作人员查看酒店订单时，hotelbl业务逻辑处理的相关对象之间的协作。

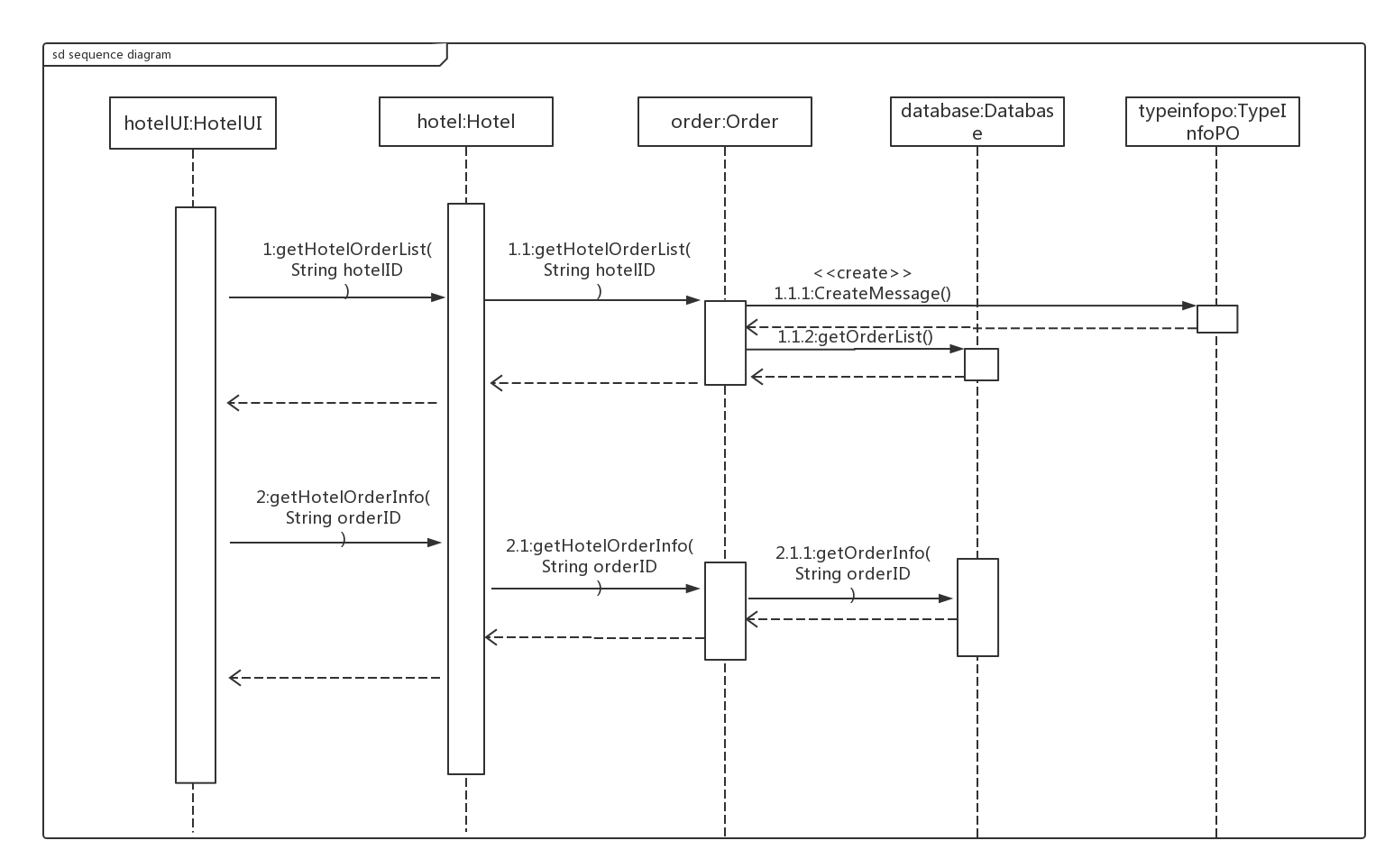


图4.1.9.4-3 酒店工作人员查看订单的顺序图

如图4.1.9.4-4所示的状态图描述了personnel对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。

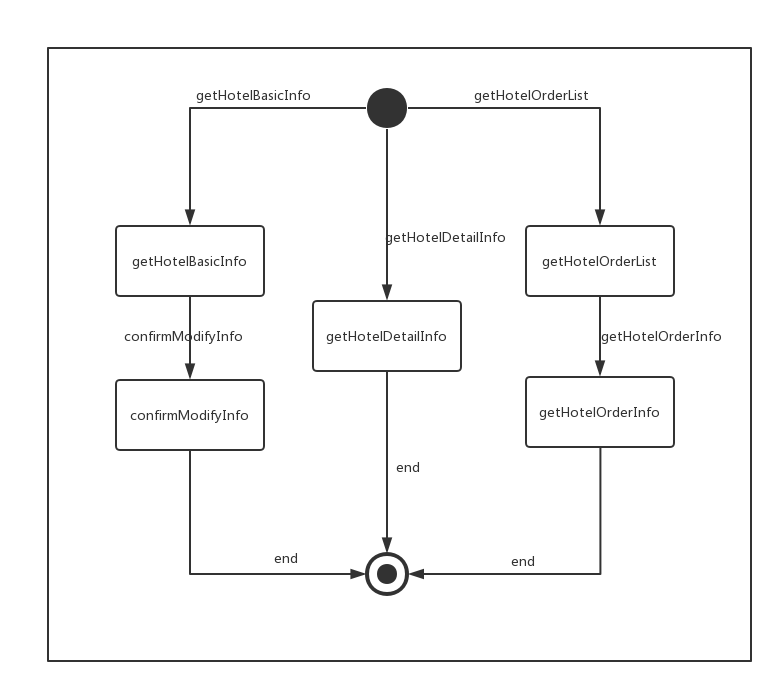


图4.1.9.4-4hotel对象状态图

#### 4.1.9.5 hotelbl业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。

### 4.1.10 Order模块

#### 4.1.10.1 Orderbl模块概述

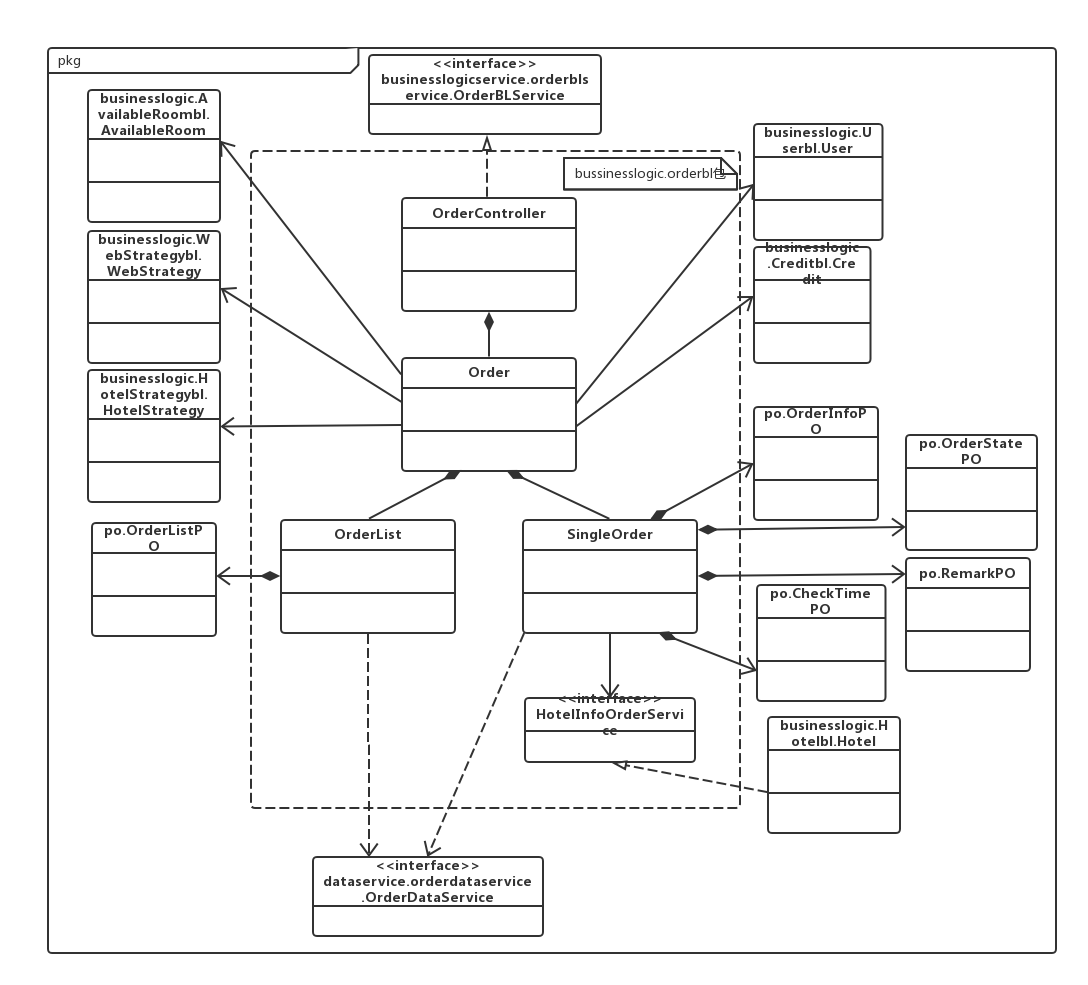
Orderbl模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

Orderbl模块的职责及接口参见软件体系结构描述文档。

#### 4.1.10.2 Orderbl整体结构

根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了Orderblservice, Orderdataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了OrderController，这样OrderController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给Order 对象。OrderListPO,OrderInfoPO,OrderStatePO,RemarkPO,CheckTimePO是做为订单信息的持久化对象被添加到设计模型中的。SingleOrder封装了关于订单的信息，而OrderList封装了SingleOrder对象的数据集合的数据结构的秘密和返回订单列表的职责，HotelInfoOrderService是根据依赖倒置原则，为了消除循环依赖而产生的接口。

Orderbl模块的设计如图4．1.10.2-1所示。



4．1.10.2-1 Orderbl模块类的设计

Orderbl模块各个类的职责如表4．1.10.2-1所示

表4．1.10.2-1Orderbl模块各个类的职责

|  |  |
| --- | --- |
| **模块** | **职责** |
| OrderController | 负责实现对应于订单界面所需的服务 |
| Order | 负责实现逻辑层订单所需的服务 |
| OrderList | 订单的数据结构，拥有某一类型的订单 |
| SingleOrder | 订单的领域模型对象，拥有订单的信息 |

#### 4.1.10.3 Orderbl模块内部类的接口规范

表4.1.10.3-1 OrderController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderController.checkUserCredit | 语法 | public ResultMessage checkUserCredit(String CustomerID) |
| 前置条件 | 已创建一个Order领域对象，顾客已登录 |
| 后置条件 | 无 |
| OrderController.checkTelephone | 语法 | public ResultMessage checkTelephone(String telNum) |
| 前置条件 | 已创建一个Order领域对象，顾客已填写联系方式 |
| 后置条件 | 无 |
| OrderController.checkTime | 语法 | public ResultMessage checkTime(Date time) |
| 前置条件 | 已创建一个Order领域对象，顾客已填写预计入住日期 |
| 后置条件 | 无 |
| OrderController.confirmAddOrder | 语法 | public ResultMessage confirmAddOrder() |
| 前置条件 | 已创建一个Order领域对象，顾客已确认提交订单且系统检查格式无误 |
| 后置条件 | 无 |
| OrderController.getOrderInfo | 语法 | public OrderInfoVO getOrderInfo(String orderID) |
| 前置条件 | 已创建一个Order领域对象，orderid符合系统要求 |
| 后置条件 | 无 |
| OrderController.getOrderList | 语法 | public ArrayList<OrderVO> getOrderList(TypeInfoVO vo) |
| 前置条件 | 已创建一个Order领域对象，用户请求浏览某一类型订单 |
| 后置条件 | 无 |
| OrderController.cancelOrderConfirm | 语法 | public ResultMessage cancelOrderConfirm (String orderID) |
| 前置条件 | 调用Order领域对象的cancelOrderConfirm方法,已创建一个Order领域对象，orderid符合系统要求 |
| 后置条件 | 无 |
| OrderController.remarkOrder | 语法 | public ResultMessage remarkOrder(RemarkVO vo) |
| 前置条件 | 已创建一个Order领域对象，顾客选择该订单进行评价 |
| 后置条件 | 无 |
| OrderController.next | 语法 | public StrategyVO next（OrderInfoVO orderinfovo) |
| 前置条件 | 已创建一个Order领域对象，生成订单 |
| 后置条件 | 无 |
| OrderController.setCheckinTime | 语法 | public ResultMessage setCheckinTime(Date time) |
| 前置条件 | 已创建一个Order领域对象，顾客入住 |
| 后置条件 | 无 |
| OrderController.setCheckoutTime | 语法 | Public ResultMessage setCheckoutTime(Date time) |
| 前置条件 | 已创建一个Order领域对象，顾客退房 |
| 后置条件 | 无 |
| OrderController.setReturnCredit | 语法 | public ResultMessage setReturnCredit(int creditNum) |
| 前置条件 | 已创建一个Order领域对象，异常订单被撤销 |
| 后置条件 | 无 |
| OrderController.getHotelDetailInfo | 语法 | Public HotelDetailInfoVO getHotelDetailInfo(String hotelID) |
| 前置条件 | 已创建一个Order领域对象，酒店房间被选择 |
| 后置条件 | 无 |
| OrderController.checkAvailableRoomNumber | 语法 | public ResultMessage checkAvailableRoomNumber (AvailableRoomNumberVO vo) |
| 前置条件 | 已创建一个Order领域对象，生成订单时，系统自动更新线上可用客房数量 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| Order.getOrderInfo | 获得订单信息 | | |
| Order.addOrder | 向数据库中新增订单 | | |
| Order.getOrderList | 获得订单列表 | | |
| Order.addOrderState | 更新订单状态 | | |
| Order.setOrderRemark | 更新订单的评价信息 | | |
| Order.setCheckintime | 更新订单的实际入住时间 | | |
| Order.setCheckouttime | 更新订单的实际退房时间 | | |
| Order.checkTelephone | 检查电话号码的长度 | | |
| Order.checkTime | 检查时间是否符合要求 | | |
| User.getUserID | 获得该订单所属的顾客信息 | | |
| Availableroom.setAvailableRoomNumber | 更新可用房间数 | | |
| Availableroom.checkAvailableRoomNumber | 检查可用房间数 | | |
| Availableroom.getRoomPrice | 获得酒店房间价格 | | |
| Credit.addCredit | 为顾客增加信用值 | | |
| Credit.cutCredit | 扣除顾客的信用值 | | |
| WebStrategy.getWebBestStrategy | 获得最大折扣的网站优惠策略 | | |
| HotelStrategy.getBestHotelStrategy | 获得最大折扣的酒店优惠策略 | | |

表4.1.10.3-2Order的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Order.checkUserCredit | 语法 | public ResultMessage checkUserCredit(String CustomerID) |
| 前置条件 | 顾客已登录 |
| 后置条件 | 若信用值大于0则数据库新增一条订单，若为负，无后置条件 |
| Order.checkTelephone | 语法 | public ResultMessage checkTelephone(String telNum) |
| 前置条件 | 顾客已填写联系方式 |
| 后置条件 | 若格式正确，顾客可继续填写，否则重新填写 |
| Order.checkTime | 语法 | public ResultMessage checkTime(Date time) |
| 前置条件 | 顾客已填写预计入住日期 |
| 后置条件 | 若格式正确，顾客可继续填写，否则重新填写 |
| Order.confirmAddOrder | 语法 | public ResultMessage confirmAddOrder() |
| 前置条件 | 顾客已确认提交订单且系统检查格式无误 |
| 后置条件 | 数据库新增一条订单 |
| Order.getOrderInfo | 语法 | public OrderInfoVO getOrderInfo(String orderID) |
| 前置条件 | orderid符合系统要求 |
| 后置条件 | 无 |
| Order.getOrderList | 语法 | public ArrayList<OrderVO> getOrderList(TypeInfoVO vo) |
| 前置条件 | 系统中有该类型订单 |
| 后置条件 | 无 |
| Order.cancelOrderConfirm | 语法 | public ResultMessage cancelOrderConfirm (String orderID) |
| 前置条件 | orderid符合系统要求 |
| 后置条件 | 系统将该订单状态置为已撤销并且更新数据库  如果是异常订单撤销，返还相应的信用值 |
| Order.remarkOrder | 语法 | public ResultMessage remarkOrder(RemarkVO vo) |
| 前置条件 | 评价信息已填写完整 |
| 后置条件 | 更新订单和对应酒店的评价信息 |
| Order.next | 语法 | public StrategyVO next（OrderInfoVO orderinfovo) |
| 前置条件 | 生成订单 |
| 后置条件 | 计算订单价格并取得相应信息 |
| Order.setCheckinTime | 语法 | public ResultMessage setCheckinTime(Date time) |
| 前置条件 | 顾客入住 |
| 后置条件 | 更新订单实际入住时间 |
| Order.setCheckoutTime | 语法 | Public ResultMessage setCheckoutTime(Date time) |
| 前置条件 | 顾客退房 |
| 后置条件 | 更新订单实际退房时间 |
| Order.setReturnCredit | 语法 | public ResultMessage setReturnCredit(int creditNum) |
| 前置条件 | 异常订单被撤销 |
| 后置条件 | 增加顾客订单总价全部或二分之一的信用值 |
| Order.getHotelDetailInfo | 语法 | Public HotelDetailInfo getHotelDetailInfo(String hotelID) |
| 前置条件 | 酒店房间被选择 |
| 后置条件 | 无 |
| Order.checkAvailableRoomNumber | 语法 | public ResultMessage checkAvailableRoomNumber (AvailableRoomNumberVO vo) |
| 前置条件 | 生成订单时，系统自动更新线上可用客房数量 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| SingleOrder.getOrderInfo | 获得订单信息 | | |
| SingleOrder.addOrder | 向数据库中新增订单 | | |
| OrderList. getOrderList | 获得订单列表 | | |
| SingleOrder.addOrderState | 更新订单状态 | | |
| SingleOrder.setOrderRemark | 更新订单的评价信息 | | |
| SingleOrder.setCheckintime | 更新订单的实际入住时间 | | |
| SingleOrder.setCheckouttime | 更新订单的实际退房时间 | | |
| User.getUserID | 获得该订单所属的顾客信息 | | |
| Availableroom.setAvailableRoomNumber | 更新可用房间数 | | |
| Availableroom.checkAvailableRoomNumber | 检查可用房间数 | | |
| Availableroom.getRoomPrice | 获得酒店房间价格 | | |
| Credit.addCredit | 为顾客增加信用值 | | |
| Credit.cutCredit | 扣除顾客的信用值 | | |
| WebStrategy.getWebBestStrategy | 获得最大折扣的网站优惠策略 | | |
| HotelStrategy.getBestHotelStrategy | 获得最大折扣的酒店优惠策略 | | |

表4.1.10.3-3 OrderList的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderList.getOrderList | 语法 | public ArrayList<OrderVO> getOrderList(TypeInfoVO vo) |
| 前置条件 | 系统中有该搜索关键字 |
| 后置条件 | 无 |
| OrderList.getOrderInfo | 语法 | public OrderInfoVO getOrderInfo(String orderId) |
| 前置条件 | 系统中存在该订单 |
| 后置条件 | 无 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| OrderdataService.getOrderList | 返回符合搜索关键字的订单 | | |

表4.1.10.3-4 SingleOrder的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| SingleOrder.getOrderInfo | 语法 | public OrderInfoVO getOrderInfo() |
| 前置条件 | 选择该订单 |
| 后置条件 | 无 |
| SingleOrder.addOrder | 语法 | Public PriceStrategyPO addOrder(OrderInfoVO vo) |
| 前置条件 | 确认增加订单 |
| 后置条件 | 计算订单价格并取得优惠策略 |
| SingleOrder.addOrderState | 语法 | Public ResultMessage addOrderState(OrderStatePO po) |
| 前置条件 | 系统中有该订单 |
| 后置条件 | 系统更新该订单状态 |
| SingleOrder.setOrderRemark | 语法 | public ResultMessage remarkOrder(RemarkVO vo) |
| 前置条件 | 顾客选择该订单进行评价 |
| 后置条件 | 更新订单和对应酒店的评价信息 |
| SingleOrder.setCheckintime | 语法 | public ResultMessage setCheckintime(CheckTimePO po) |
| 前置条件 | 系统中有该订单 |
| 后置条件 | 系统更新该订单实际入住时间 |
| SingleOrder.setCheckouttime | 语法 | public ResultMessage setCheckouttime(CheckTimePO po) |
| 前置条件 | 系统中有该订单 |
| 后置条件 | 系统更新该订单实际入住时间 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | | |
| OrderdataService.getOrderInfo | 获得订单信息 | | |
| OrderdataService.addOrderState | 新增订单状态并更改订单当前状态 | | |
| OrderdataService.setOrderRemark | 增加订单评价信息 | | |
| OrderdataService.setCheckintime | 更新订单实际入住时间 | | |
| OrderdataService.setCheckouttime | 更新订单实际退房时间 | | |
| OrderdataService.addOrder | 新增订单 | | |

#### 4.1.10.4 Orderbl业务逻辑层的动态模型

图4.1.10.4-1表明了酒店预定系统中，当用户填写完了相关的订单信息之后，订单业务逻辑处理的相关对象之间的协作。

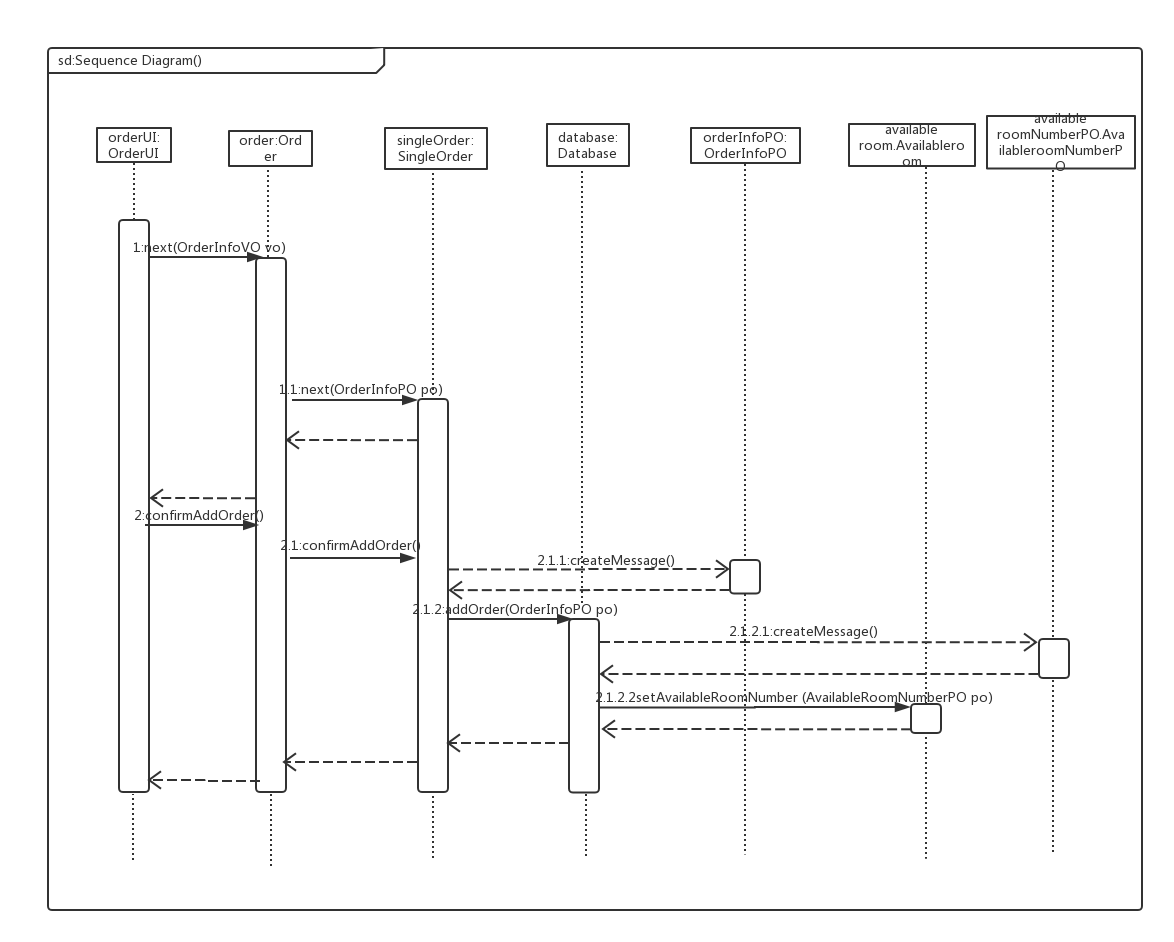


图4.1.10.4-1新增订单的顺序图

图4.1.10.4-2表明了在酒店预定系统中，当顾客填写完了对订单的评价信息之后，订单业务逻辑处理的相关对象之间的协作。

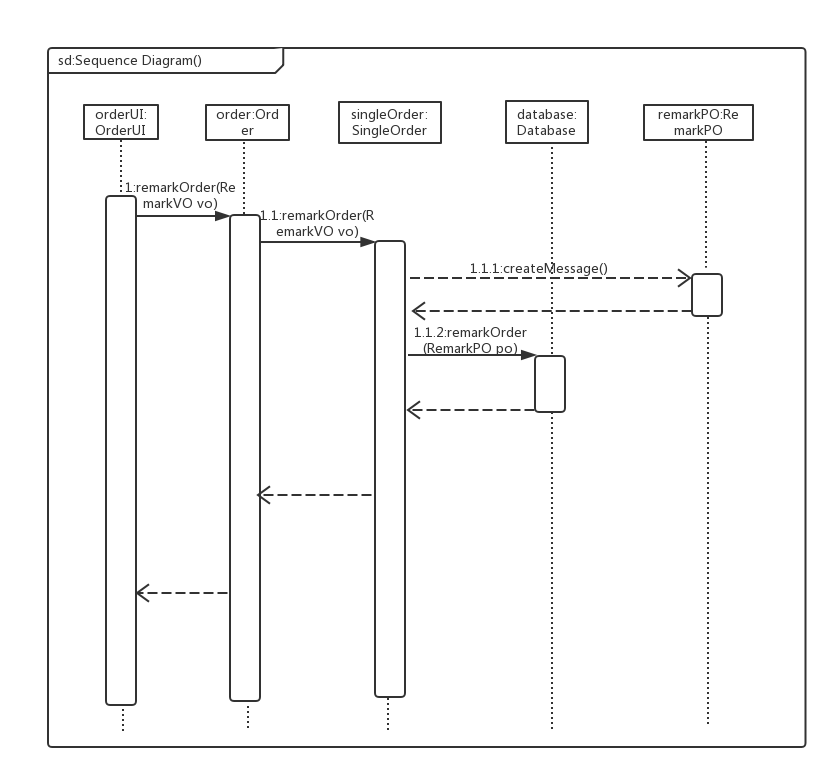


图4.1.10.4-2评价订单的顺序图

图4.1.10.4-3表明了在酒店预定系统中，当用户查看订单详细信息时，订单业务逻辑处理的相关对象之间的协作。

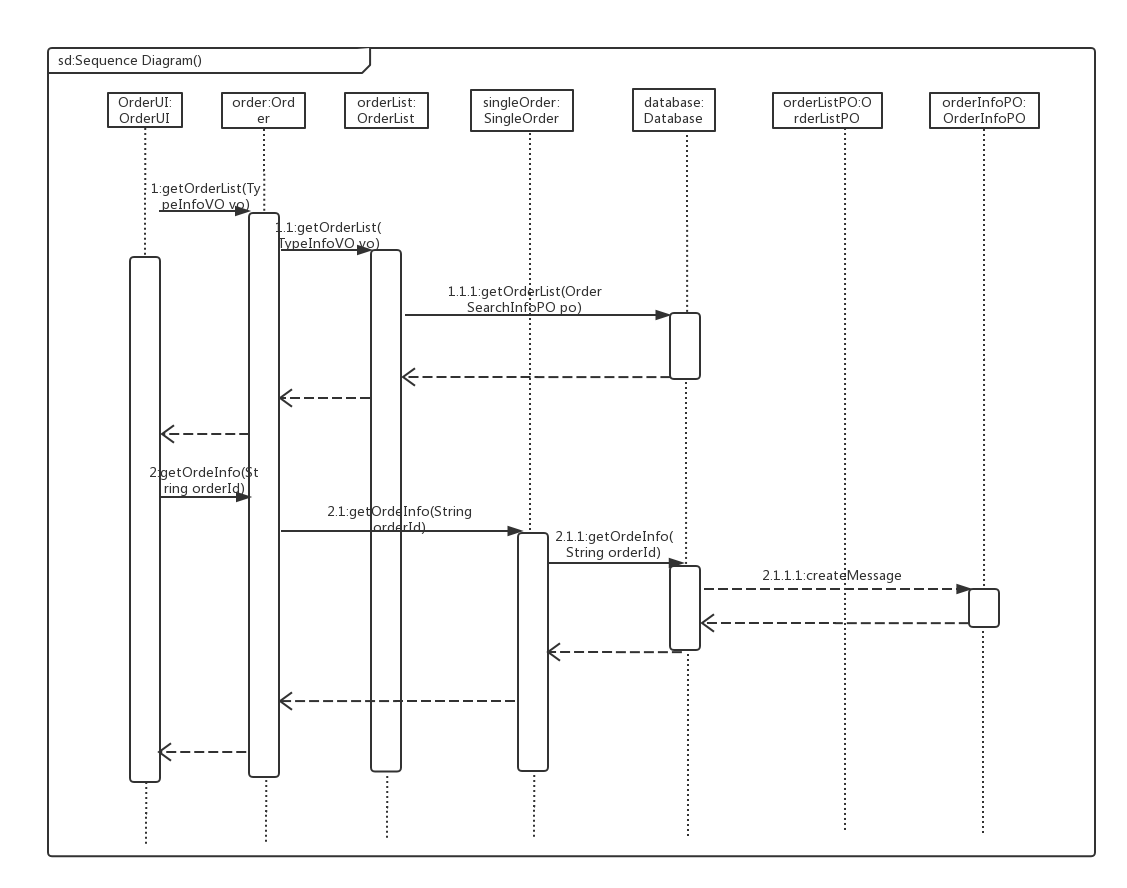


图4.1.10.4-3查看订单的顺序图

图4.1.10.4-4表明了在酒店预定系统中，当顾客撤销订单时，订单业务逻辑处理的相关对象之间的协作。

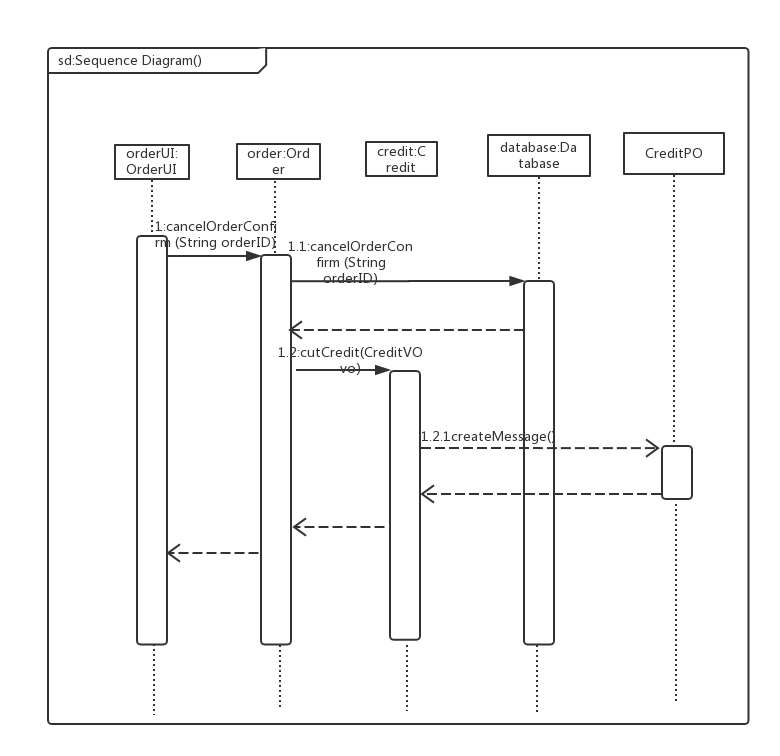


图4.1.10.4-4撤销订单的顺序图

图4.1.10.4-5所示的状态图描述了order对象的生存期间的状态序列，引起转移的事件，以及因状态转移而伴随的动作。随着getOrderList方法被UI调用，Order进入OrderList状态；之后通过获取订单详细信息进入SingleOrder状态。UI也可以不通过订单列表查看订单详细信息而直接进入SingleOrder状态。

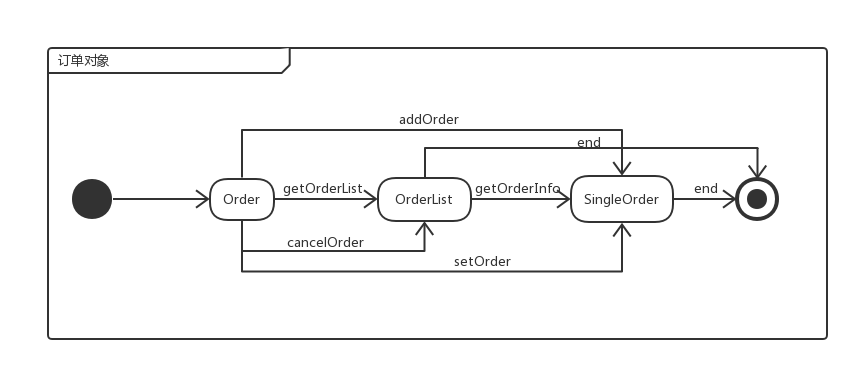


图4.1.10.4-5 order对象状态图

#### 4.1.10.5 Orderbl业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。