# 酒店管理系统详细设计文档

学 院：南京大学软件学院

成 员：宋欣建 叶涵 姚政 黄程尔

完成日期：2016年10月29日

## 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **修改日期** | **修改原因** | **版本号** |
| 宋欣建 | 2016-10-25 | 最初草稿（创建文档模板） | V0.1 |

## 一、引言

### 1.1 编制目的

本报告详细完成对酒店管理系统的详细设计，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

|  |  |  |
| --- | --- | --- |
| 词汇名称 | 词汇含义 | 备注 |
| \_ui | 表示某展示层 |  |
| \_bl | 表示某逻辑层 |  |
| \_data | 表示某数据层 |  |

### 1.3 参考资料

1. IEEE std 1471-2000
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

## 二、中层设计之逻辑层设计

### 2.1 User模块的静态结构与动态行为

#### 2.1.1 局部模块的职责

（1）模块概述

User模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。User模块的职责及接口参见软件系统结构描述文档表5.3.2.1

（2）整体结构

User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。User模块的职责及接口参见软件体系结构描述文档。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UserService, UserDAO 两个接口。UserPO是做为管理信息的持久化对象被添加到设计模型中的。

图2.1.1-1 User模块设计

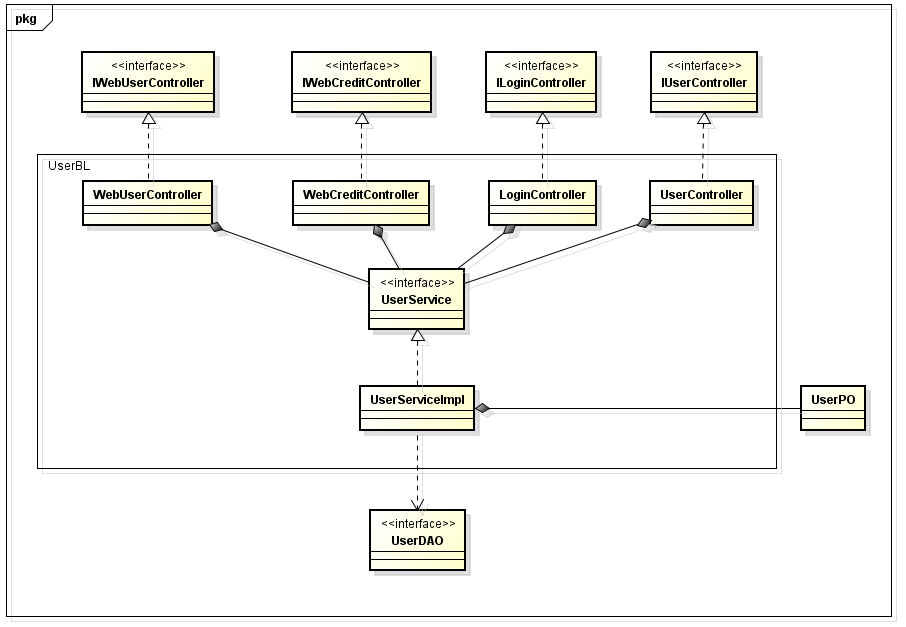


表2.1.1-2 User模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| UserController | 负责实现对应个人信息界面所需的方法 |
| LoginController | 负责实现对应登录注册界面所需的方法 |
| WebMarketController | 负责实现对应网站信用充值界面所需的方法 |
| WebStaffController | 负责实现对应网站用户信息界面所需的方法 |
| UserServiceImpl | 负责处理有关用户的功能 |

#### 2.1.2 局部模块的接口规范

表2.1.2 -1 UserService的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UserService.findByUsername | 语法 | public UserVO findByUsername(String username) | |
| 前置条件 | 已知用户名，需要获得该用户的详细信息 | |
| 后置条件 | 无 | |
| UserService.register | 语法 | public ResultMessage register(UserVO) | |
| 前置条件 | 已知用户信息，需要将其注册 | |
| 后置条件 | 无 | |
| UserService.update | 语法 | public ResultMessage update(UserVO) | |
| 前置条件 | 已知该用户修改后的信息，需要更新该用户信息 | |
| 后置条件 | 更新数据库中用户的信息 | |
| UserService.login | 语法 | public UserVO login(String username,String password) | |
| 前置条件 | 已知该用户的用户名和密码，需要将其登录 | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UserDAO.findByUsername(String username) | | | 查找单一持久化序列 |
| UserDAO.add(UserPO) | | | 添加单一持久化序列 |
| UserDAO.update(UserPO) | | | 更新单一持久化序列 |

#### 2.1.3 局部模块的行为

图2.1.3-1 User模块时序图1

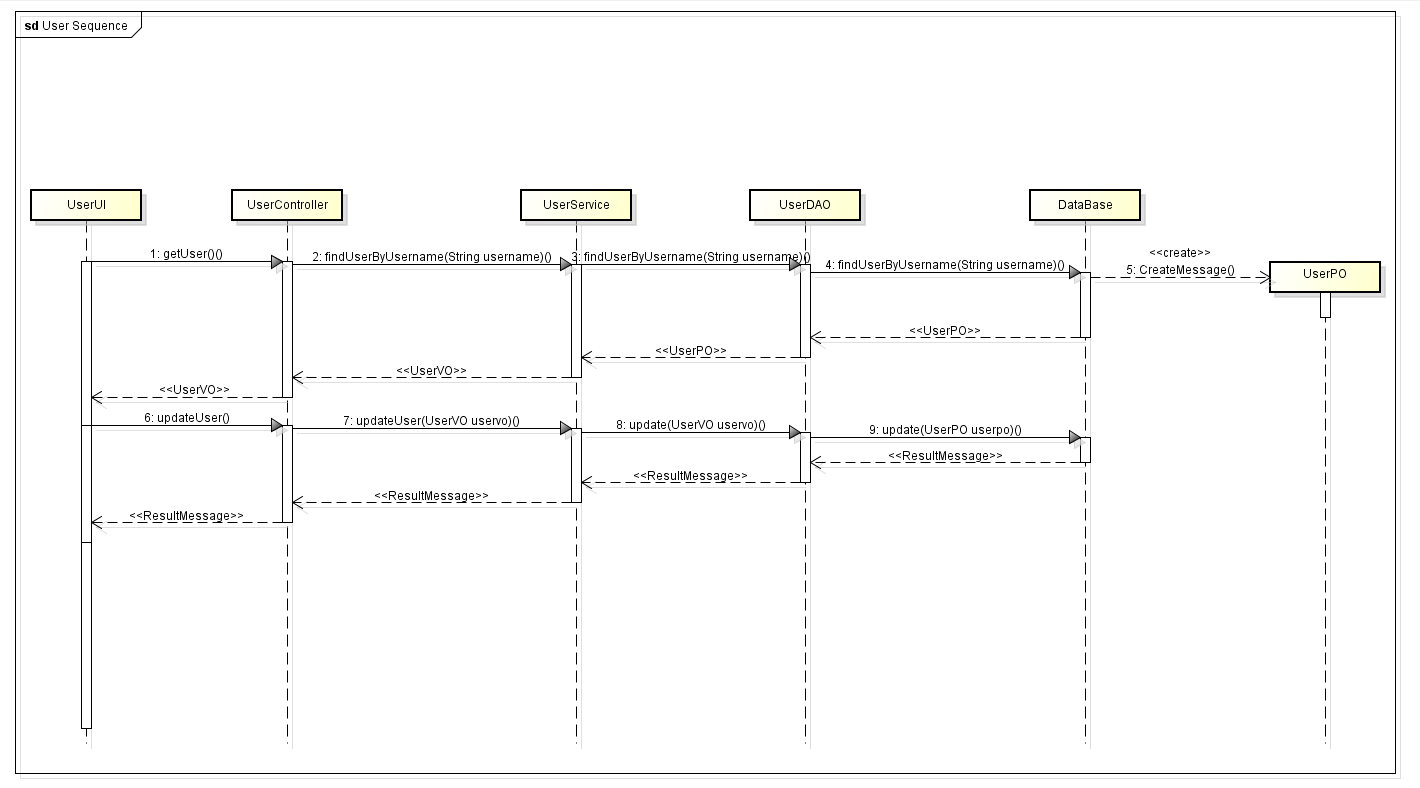


图2.1.3-1 表明了酒店管理系统中，当用户查询和修改用户信息时，用户业务逻辑处理的相关对象之间的协作。

图2.1.3-2 User模块时序图2

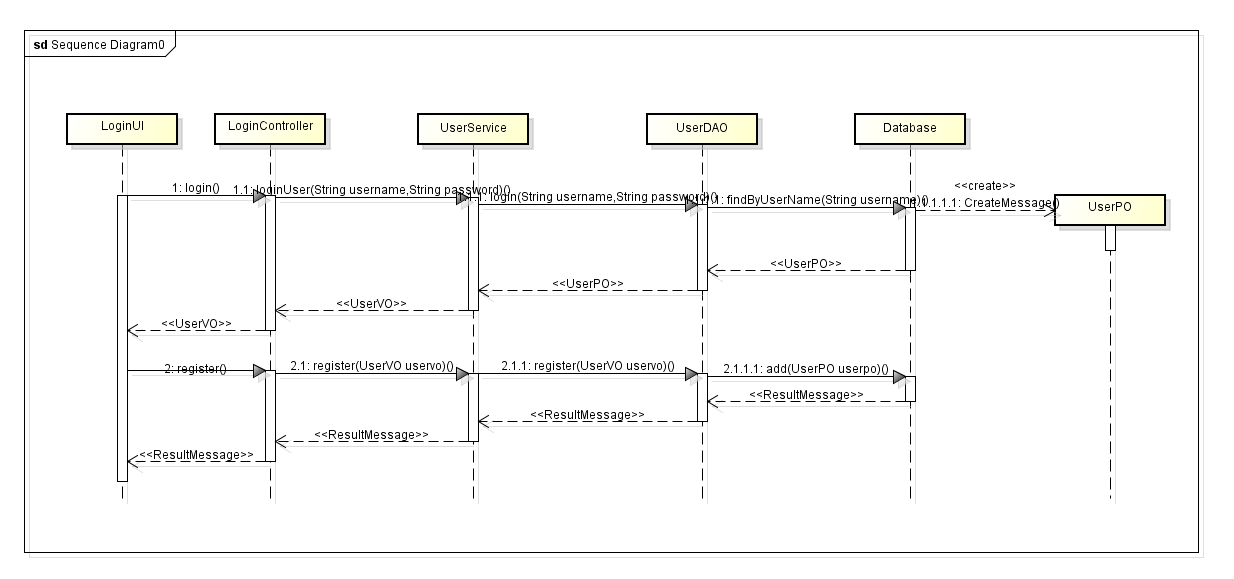
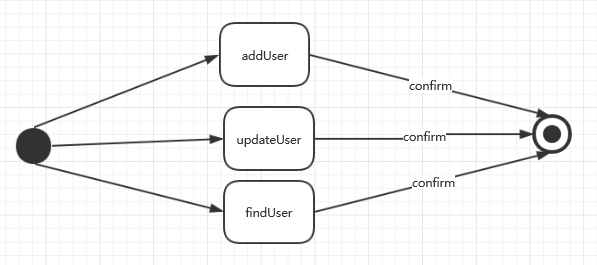


图2.1.3-2 表明了酒店管理系统中，当用户登录注册时，用户业务逻辑处理的相关对象之间的协作。

图2.1.3-2 User模块状态图



### 2.2 Order模块的静态结构与动态行为

### 2.3 Hotel模块的静态结构与动态行为

#### 2.3.1 HotelService

##### 2.3.1.1 局部模块的职责

（1）模块概述

HotelService模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。HotelService模块的职责及接口参见软件系统结构描述文档表5.3.2-3

（2）整体结构

HotelService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。HotelService模块的职责及接口参见软件体系结构描述文档。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HotelService, HotelDAO两个接口。HotelPO是做为管理信息的持久化对象被添加到设计模型中的。

图2.3.1.1-1 HotelService模块设计

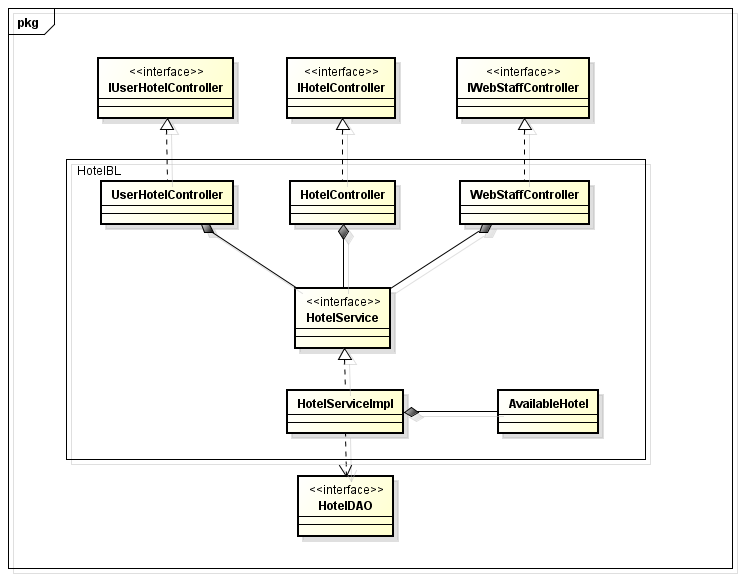


表2.3.1.1-2 HotelService模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| UserHotelController | 负责实现对应用户酒店信息界面所需的方法 |
| HotelController | 负责实现对应酒店管理界面所需的方法 |
| WebStaffController | 负责实现对应网站管理人员酒店添加界面所需的方法 |
| HotelServiceImpl | 负责处理有关酒店的功能 |

##### 2.3.1.2 局部模块的接口规范

表2.3.1.2 -1 HotelService的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| HotelService. findByID | 语法 | public HotelVO findByID(int hotelID) | |
| 前置条件 | 已知该酒店的编号，需要该酒店的详细信息 | |
| 后置条件 | 无 | |
| HotelService.update | 语法 | public ResultMessage update(HotelVO) | |
| 前置条件 | 已知该酒店的更新后的信息，需要更新该酒店信息 | |
| 后置条件 | 在数据库中更新该酒店信息 | |
| HotelService.add | 语法 | public ResultMessage add(HotelVO) | |
| 前置条件 | 已知该酒店的信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中增加该酒店信息 | |
| HotelService.findOrderedHotelAndOrder | 语法 | public Map<HotelVO,List<OrderVO>>>  findOrderedHotelAndOrder(String username) | |
| 前置条件 | 已知用户名，需要用户曾预订的酒店列表以及每个酒店对应的订单信息 | |
| 后置条件 | 返回酒店列表 | |
| HotelService. find | 语法 | public Map<HotelVO,List<RoomVO>>>  find (int loc,int circle,Date begin,Date end) | |
| 前置条件 | 已知该酒店的地址、商圈、订单时间，得到可用房间 | |
| 后置条件 | 无 | |
| HotelService. filter | 语法 | public Map<HotelVO,List<RoomVO>>  filter(List<FilterCondition>) | |
| 前置条件 | 已知酒店列表，需要按照给定的过滤条件进行筛选 | |
| 后置条件 | 无 | |
| HotelService. order | 语法 | public Map<HotelVO,List<RoomVO>>  order(OrderRule rule,boolean isDecrease); | |
| 前置条件 | 已知酒店列表，需要按照给定条件进行排序 | |
| 后置条件 | 返回排好序的酒店列表 | |
| HotelService. getRoomDetail | 语法 | public List<RoomVO>  getRoomDetail(,int hotelID) | |
| 前置条件 | 已知酒店编号，需要取出该酒店的对应的房间信息 | |
| 后置条件 | 无 | |
| LocationService.findAll | 语法 | public List<LocationVO> findAll() | |
| 前置条件 | 需要所有的地点信息 | |
| 后置条件 | 无 | |
| CommCircleService.findByLoc | 语法 | public List<CommercialCircleVO> findByLocation(int locID) | |
| 前置条件 | 已知地点，需要该地点的所有商圈 | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| HotelDAO.findByID(int hotelID) | | | 查找单一持久化序列 |
| HotelDAO.update(HotelPO) | | | 更新单一持久化序列 |
| HotelDAO.add(HotelPO) | | | 增加单一持久化序列 |
| HotelDAO. find (int loc,int circle,Date begin,Date end) | | | 查找单一持久化列表 |
| LocationDAO. findAll() | | | 查找单一持久化列表 |
| CommCircleDAO. findByLoc(int locID) | | | 查找单一持久化列表 |
| OrderService.findByUsername(String username) | | | 返回该用户的所有订单列表 |
| OrderService. findByHotelAndUsername(int hotelID,String username) | | | 返回该酒店的该用户的订单记录列表 |
| RoomService. findAvailableByHotelID (int hotelID) | | | 返回该酒店的所有可用房间信息 |

##### 2.3.1.3 局部模块的行为

图2.3.1.3-1 HotelService模块时序图1

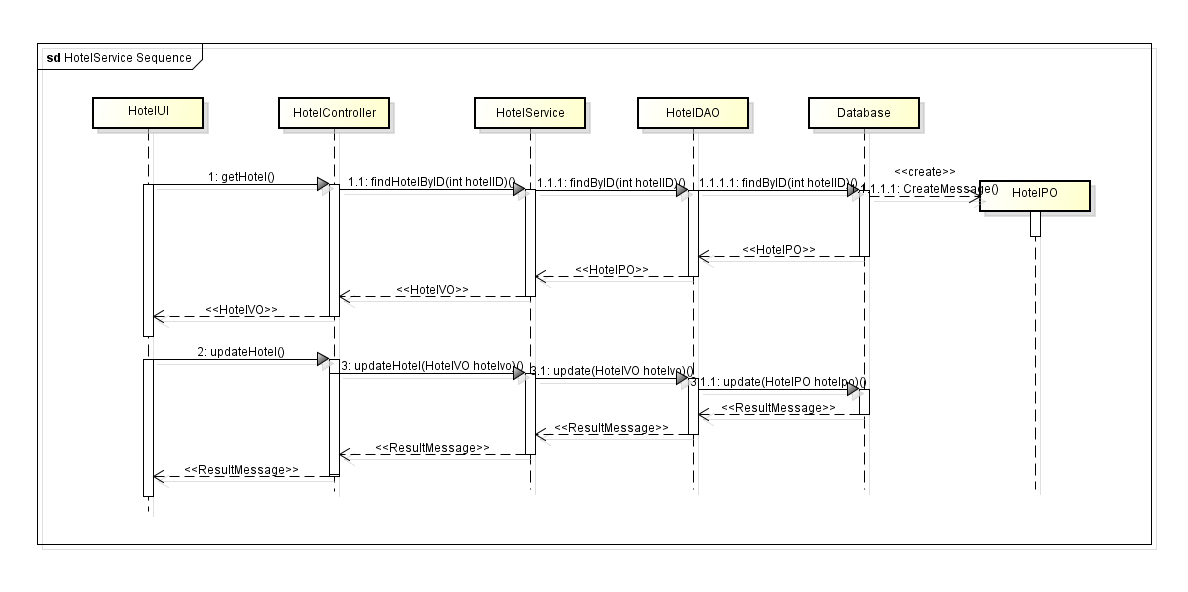


图2.1.3-1 表明了酒店管理系统中，当酒店管理人员查询和修改酒店信息时，酒店业务逻辑处理的相关对象之间的协作。

图2.3.1.3-2 HotelService模块时序图2

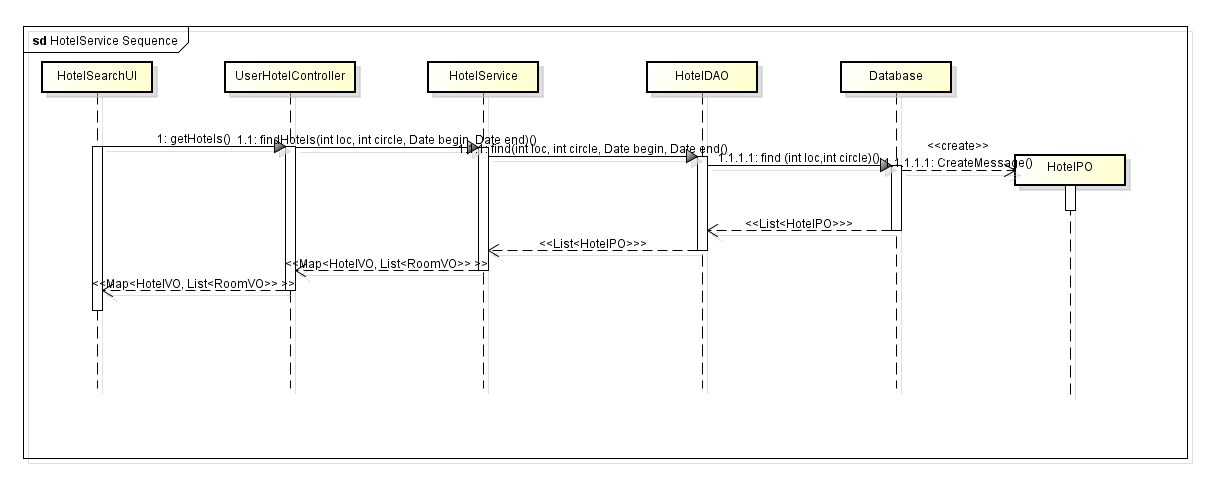
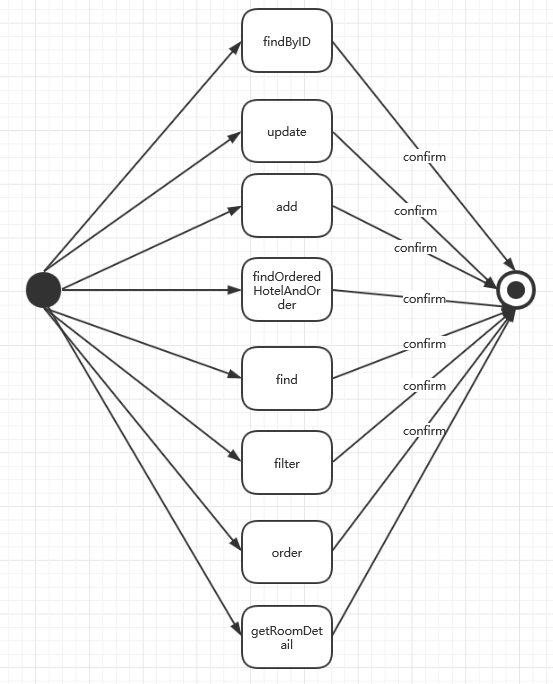


图2.3.1.3-2 表明了酒店管理系统中，当用户查询酒店信息时，酒店业务逻辑处理的相关对象之间的协作。

图2.3.1.3-3 HotelService模块状态图



#### 2.3.2 LocationService

##### 2.3.2.1 局部模块的职责

（1）模块概述

LocationService模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。LocationService模块的职责及接口参见软件系统结构描述文档表5.3.2-3

（2）整体结构

LocationService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。LocationService模块的职责及接口参见软件体系结构描述文档。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LocationService, LocationDAO两个接口。LocationPO是做为管理信息的持久化对象被添加到设计模型中的。

图2.3.1.1-1 LocationService模块设计

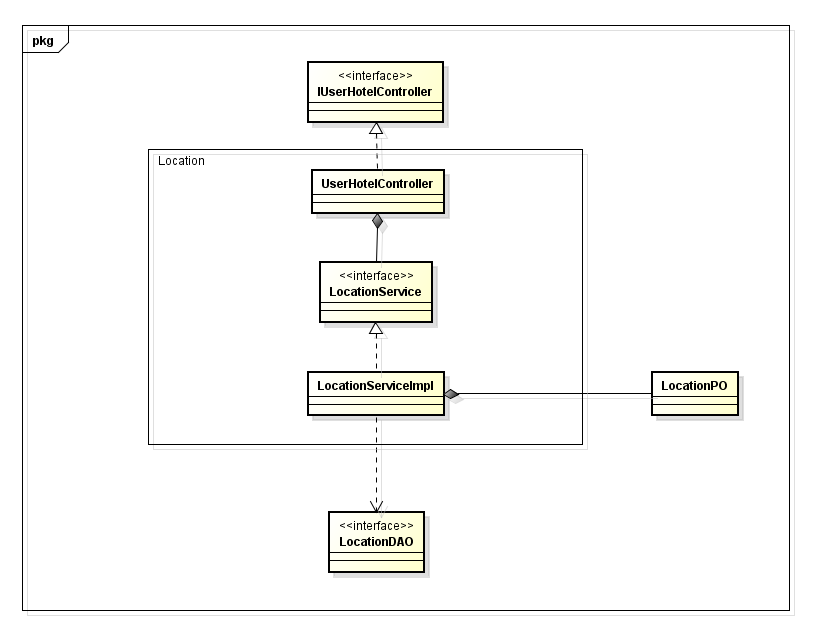


表2.3.1.1-2 LocationService模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| UserHotelController | 负责实现对应用户酒店信息界面所需的方法 |
| LocationServiceImpl | 负责处理有关城市的功能 |

##### 2.3.2.2 局部模块的接口规范

表2.3.2.2 -1 LocationService的接口规范

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 提供的服务（供接口） | | | | |
| LocationService.findAll | 语法 | public List<LocationVO> findAll() | | |
| 前置条件 | 需要所有的地点信息 | | |
| 后置条件 | 无 | | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| LocationDAO. findAll() | | | 查找单一持久化列表 |

##### 2.3.2.3 局部模块的行为

图2.3.2.3-1 LocationService模块时序图

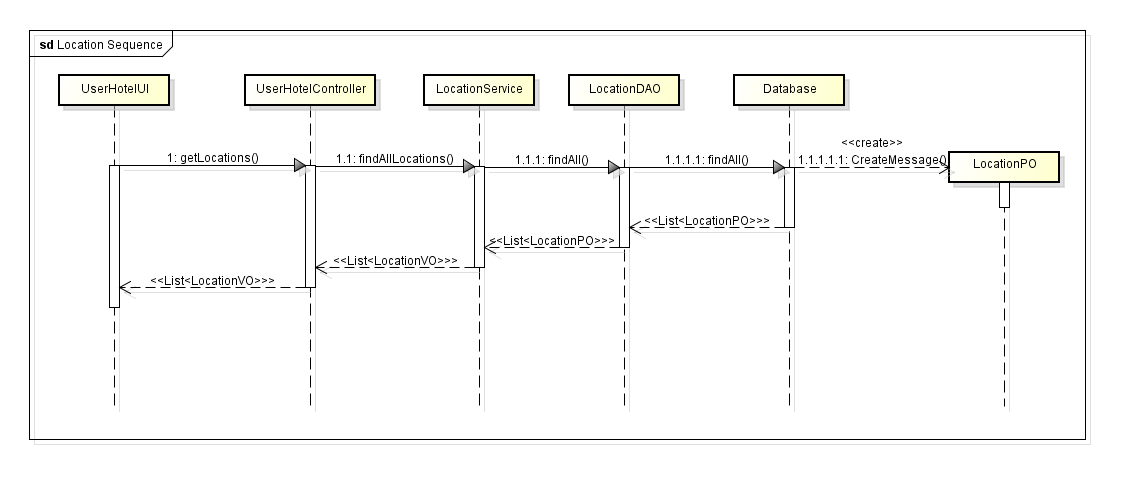
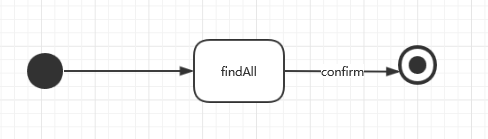


图2.3.2.3-1 表明了酒店管理系统中，当用户查询酒店信息时，城市业务逻辑处理的相关对象之间的协作。

图2.3.2.3-2 LocationService模块状态图



#### 2.3.3 CommCircleService

##### 2.3.3.1 局部模块的职责

（1）模块概述

CommCircleService模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。CommCircleService模块的职责及接口参见软件系统结构描述文档表5.3.2-3

（2）整体结构

CommCircleService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。CommCircleService模块的职责及接口参见软件体系结构描述文档。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CommCircleService, CommCircleDAO两个接口。CommercialCirclePO是做为管理信息的持久化对象被添加到设计模型中的。

图2.3.3.1-1 CommCircleService模块设计

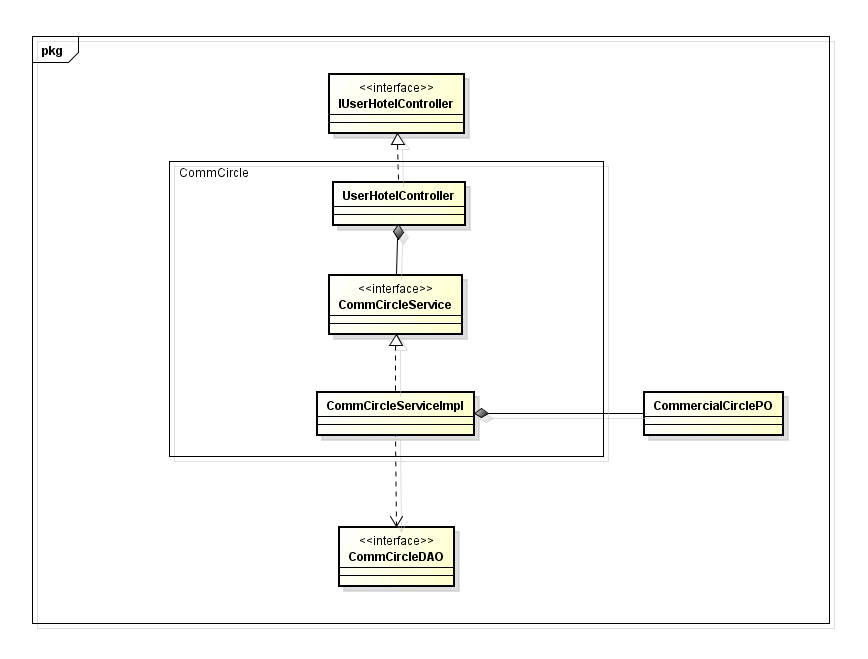


表2.3.1.1-2 CommCircleService模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| UserHotelController | 负责实现对应用户酒店信息界面所需的方法 |
| CommCircleServiceImpl | 负责处理有关商圈的功能 |

##### 2.3.3.2 局部模块的接口规范

表2.3.3.2 -1 CommCircleService的接口规范

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 提供的服务（供接口） | | | | |
| CommCircleService.findByLoc | 语法 | public List<CommercialCircleVO> findByLocation(int locID) | | |
| 前置条件 | 已知地点，需要该地点的所有商圈 | | |
| 后置条件 | 无 | | |
| 需要的服务（需接口） | | | |
| 服务名 | | | | 服务 |
| CommCircleDAO. findByLoc(int locID) | | | | 查找单一持久化列表 |

##### 2.3.2.3 局部模块的行为

图2.3.2.3-1 CommCircleService模块时序图

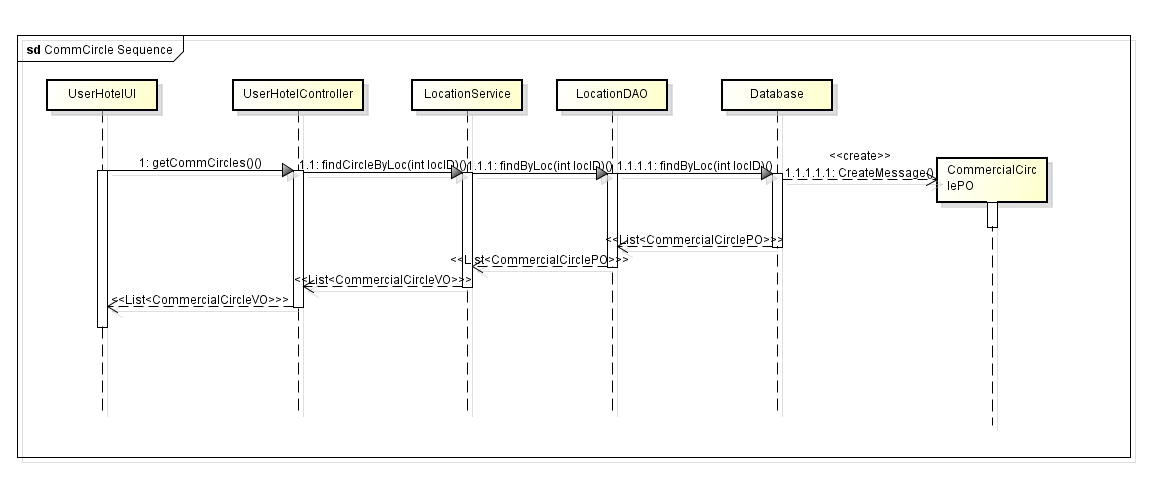
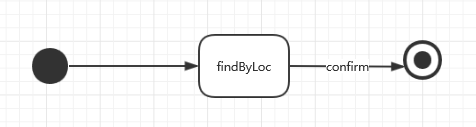


图2.3.2.3-1 表明了酒店管理系统中，当用户查询酒店信息时，商圈业务逻辑处理的相关对象之间的协作。

图2.3.2.3-2 CommCircleService模块状态图



### 2.4 Room模块的静态结构与动态行为

### 2.5 Promotion模块的静态结构与动态行为

### 2.6 CreditRecord模块的静态结构与动态行为

### 2.7 OfflineRecord模块的静态结构与动态行为

### 2.8 Staff模块的静态结构与动态行为

## 三、中层设计之控制器设计

下面来解释控制器Controller引入的原因：

为了避免按照用例分解导致的逻辑重复冗余问题，按照业务逻辑进行模块分解，这就导致了逻辑层与展示层并非是一一对应的，往往单一展示层模块会依赖于多个逻辑层模块。依据最小接口原则，且为了保证展示层能方便地调用逻辑层，整合展示层和逻辑层之间复杂的依赖关系，引入了Controller这一概念。展示层中每一界面与Controller一一对应，而Controller则持有Service，包含了对应界面所依赖的最小接口。为了保证接口隔离，设置Controller的接口IController，展示层仅依赖于IController接口而非Controller实现。

### 3.1 LoginUI包所对应的Controller

#### 3.1.1 LoginController

图3.1.1-1 LoginController 类图

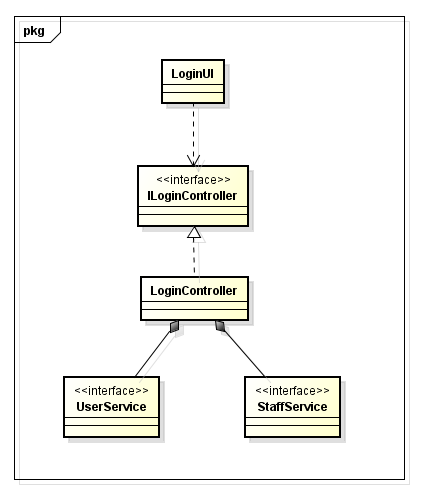


表3.1.1 -2 LoginController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| LoginController. register | 语法 | public ResultMessage register(UserVO uservo) | |
| 前置条件 | 已知用户信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中保存新增的用户信息 | |
| LoginController.loginUser | 语法 | public UserVO loginUser(String username, String password) | |
| 前置条件 | 已知用户名和密码，需要验证该用户的身份 | |
| 后置条件 | 无 | |
| LoginController.loginStaff | 语法 | public StaffVO loginStaff(String username, String password) | |
| 前置条件 | 已知用户名和密码，需要验证该职员的身份 | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UserService.login(String username,String  password) | | | 查找单一持久化序列 |
| UserService.register(UserPO) | | | 添加单一持久化序列 |
| StaffService.login(String username,String  password) | | | 查找单一持久化序列 |

### 3.2 UserUI包所对应的Controller

#### 3.2.1 UserController

图3.2.1-1 UserController 类图

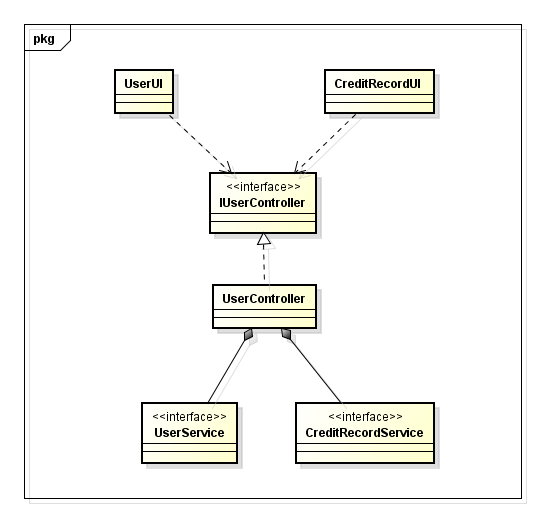


表3.2.1 -2 UserController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UserController.updateUser | 语法 | public ResultMessage updateUser(UserVO uservo) | |
| 前置条件 | 已知更新后的信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中更新用户信息 | |
| UserController.findUserByUsername | 语法 | public UserVO  findUserByUsername(String username) | |
| 前置条件 | 已知用户名，需要返回该用户的详细信息 | |
| 后置条件 | 无 | |
| UserController.findCreditRecordByUsername | 语法 | public List<CreditRecordVO>  findCreditRecordByUsername(String username) | |
| 前置条件 | 已知用户名，需要该用户所有的信用记录 | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UserService.update(UserVO) | | | 更新单一持久化序列 |
| UserService.findByUsername(String username) | | | 查找单一持久化序列 |
| CreditRecordService.findByUsername(String username) | | | 查找单一持久化列表 |

#### 3.2.2 UserOrderController

图3.2.2-1 UserOrderController 类图

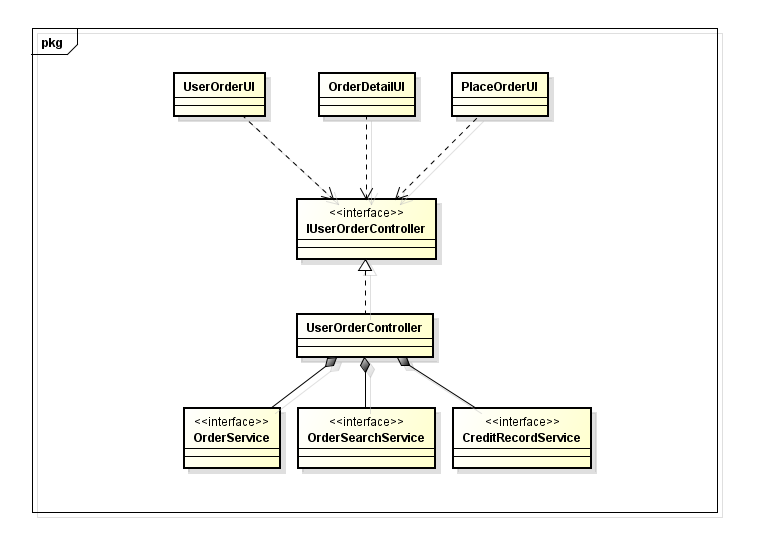


表3.2.2 -2 UserOrderController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UserOrderController.findOrdersByUsernameAndType | 语法 | public List<OrderVO>  findOrdersByUsernameAndType(String username, OrderStatus type) | |
| 前置条件 | 已知用户名和订单类型，需要所有满足条件的订单 | |
| 后置条件 | 无 | |
| UserOrderController. placeOrder | 语法 | public OrderVO placeOrder(OrderVO  ordervo) | |
| 前置条件 | 已知订单信息，需要对订单进行处理，为下单做准备 | |
| 后置条件 | 无 | |
| UserOrderController addOrder | 语法 | public ResultMessage addOrder(OrderVO ordervo) | |
| 前置条件 | 已知订单信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中保存新添加的订单 | |
| UserOrderController.remark | 语法 | public ResultMessage remark(OrderVO  ordervo) | |
| 前置条件 | 已知订单信息，需要添加其评论信息 | |
| 后置条件 | 在数据库中保存新的订单信息 | |
| UserOrderController.revoke | 语法 | public ResultMessage revoke(OrderVO ordervo) | |
| 前置条件 | 已知订单信息，需要将订单状态改为已撤销 | |
| 后置条件 | 在数据库中保存新的订单信息 | |
| UserOrderController.addCreditRecord | 语法 | public ResultMessage  addCreditRecord(CreditRecordVO creditrecordvo); | |
| 前置条件 | 已知信用记录信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中保存新添加的信用记录信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| OrderService. placeOrder(OrderVO ordervo) | | | 处理订单信息 |
| OrderService. add(OrderVO ordervo) | | | 添加单一持久化序列 |
| OrderService. revoke(OrderVO ordervo) | | | 撤销订单 |
| OrderService. remark(OrderVO ordervo) | | | 评价订单 |
| OrderSearchService. findByUsernameAndType(String username,  OrderStatus type) | | | 查找单一持久化列表 |
| CreditRecordService. add(CreditRecordVO  creditrecordvo) | | | 添加单一持久化序列 |

#### 3.2.3 UserHotelController

图3.2.3-1 UserHotelController 类图

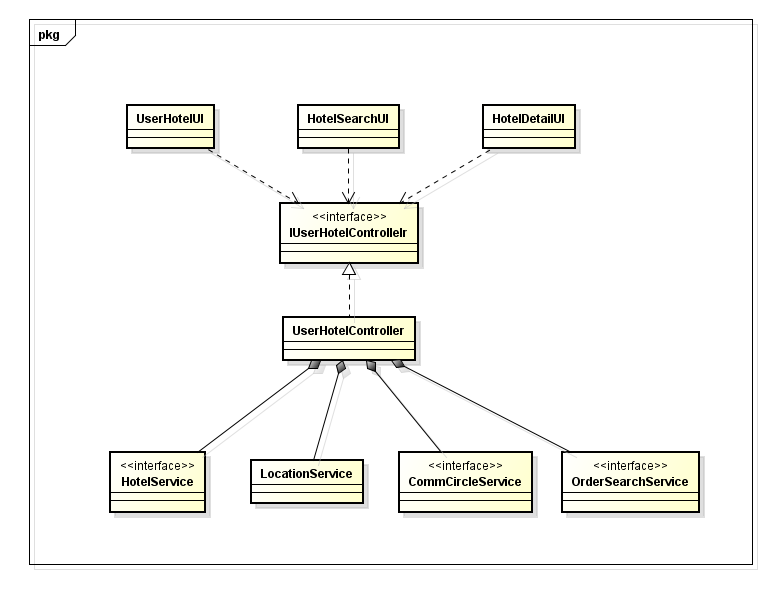


表3.2.3 -2 UserHotelController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UserHotelController.findOrderByHotelAndUsername | 语法 | public List<OrderVO>  findOrderByHotelAndUsername(int hotelID, String username) | |
| 前置条件 | 已知酒店编号和用户名，需要返回满足条件的订单列表 | |
| 后置条件 | 无 | |
| UserHotelController. getRoomDetail | 语法 | public List<RoomVO> getRoomDetail(int  hotelID) | |
| 前置条件 | 已知酒店编号，需要返回满足条件的房间列表 | |
| 后置条件 | 无 | |
| UserHotelController. findOrderedHotelAndOrder | 语法 | public Map<HotelVO, List<OrderVO>> findOrderedHotelAndOrder(String username) | |
| 前置条件 | 已知用户名，需要找到该用户曾预定的酒店和订单信息 | |
| 后置条件 | 无 | |
| UserHotelController.findHotels | 语法 | public Map<HotelVO, List<RoomVO>> findHotels(int loc, int circle, Date begin, Date end) | |
| 前置条件 | 已知酒店、商圈和时间，需要返回满足条件的酒店和房间列表 | |
| 后置条件 | 无 | |
| UserHotelController.filterHotels | 语法 | public Map<HotelVO, List<RoomVO>> filterHotels(List<FilterCondition> conditions) | |
| 前置条件 | 已知过滤条件列表，需要返回过滤后的酒店房间列表 | |
| 后置条件 | 无 | |
| UserHotelController. orderHotels | 语法 | public Map<HotelVO, List<RoomVO>> orderHotels(OrderRule rule, boolean  isDecrease) | |
| 前置条件 | 已知排序规则和升降序，需要排序后的房间酒店列表 | |
| 后置条件 | 无 | |
| UserHotelController. findAllLocations | 语法 | public List<LocationVO>  findAllLocations() | |
| 前置条件 | 需要所有城市列表 | |
| 后置条件 | 无 | |
| UserHotelController. findCircleByLoc | 语法 | public List<CommercialCircleVO>  findCircleByLoc(int locID) | |
| 前置条件 | 已知城市编号，需要当前城市的所有商圈 | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| HotelSearchService. findByHotelAndUsername(int hotelID, String username) | | | 查找单一持久化列表 |
| HotelService. getRoomDetail(int hotelID) | | | 查找单一持久化列表 |
| HotelService.findOrderedHotelAndOrder(String username) | | | 查找单一持久化列表 |
| HotelService. find(int loc, int circle, Date begin, Date end) | | | 查找单一持久化列表 |
| HotelService. filter(List<FilterCondition>  conditions) | | | 过滤订单 |
| HotelService. order(OrderRule rule, boolean  isDecrease) | | | 排序订单 |
| LocationService. findAll() | | | 查找单一持久化列表 |
| CommCircleService. findByLoc(int locID) | | | 查找单一持久化列表 |

### 3.3 HotelUI包所对应的Controller

#### 3.3.1 HotelController

#### 3.3.2 HotelOrderController

#### 3.3.3 HotelDiscountController

### 3.4 WebMarketUI包所对应的Controller

#### 3.4.1 WebDiscountController

#### 3.4.2 WebOrderController

#### 3.4.3 WebCreditController

### 3.5 WebStaffUI包所对应的Controller

#### 3.5.1 WebUserController

#### 3.5.2 WebStaffController