酒店预订系统

HRS(Hotel Reservation System)

体系结构设计文档

V1.0 正式版

冰与火小组

2016-10-12

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **日期** | **变更原因** | **版本号** |
| 王洪凯 | 2016-10-12 | 最初草稿 | V1.0 草稿 |
| 王洪凯 | 2016-10-12 | 评审后的正式版 | V1.0 正式版 |

**目录**

[更新历史 2](#_Toc24142)

[1、 引言 5](#_Toc31544)

[1.1编制目的 5](#_Toc11933)

[1.2 词汇表 5](#_Toc20916)

[1.3 参考资料 5](#_Toc23948)

[2、 产品概述 5](#_Toc5085)

[3、 逻辑视角 5](#_Toc30668)

[4、 组合视角 7](#_Toc14859)

[4.1 开发包图 7](#_Toc24854)

[4.2 运行时进程 10](#_Toc15054)

[4.3 物理部署 10](#_Toc14479)

[5、 接口视角 11](#_Toc30780)

[5.1 模块的职责 11](#_Toc16726)

[5.2用户界面层的分解 13](#_Toc29911)

[5.2.1 用户界面层模块的职责 15](#_Toc30805)

[5.2.2 用户界面层模块的接口规范 15](#_Toc29834)

[5.2.3 用户界面模块设计原理 16](#_Toc2511)

[5.3 业务逻辑层的分解 16](#_Toc4739)

[5.3.1 业务逻辑层模块的职责 16](#_Toc11339)

[5.3.2 业务逻辑层的接口规范 17](#_Toc23292)

[5.4 数据层的分解 24](#_Toc21436)

[5.4.1 数据层模块的职责 24](#_Toc12337)

[5.4.2 数据层模块的接口规范 25](#_Toc1826)

[6、 信息视角 27](#_Toc15376)

[6.1 数据持久化对象 27](#_Toc13300)

[6.2 Txt 持久化格式 30](#_Toc31850)

[6.3 数据库表 31](#_Toc30226)

# 引言

## 1.1编制目的

本报告详细完成对酒店预订系统的概要设计，达到指导详细设计和开发的目的，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 |
| HRU | 酒店预订系统 |  |

## 1.3 参考资料

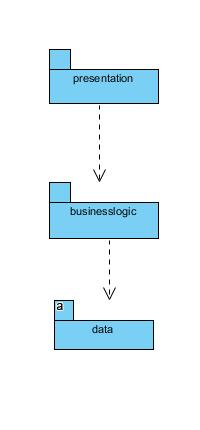
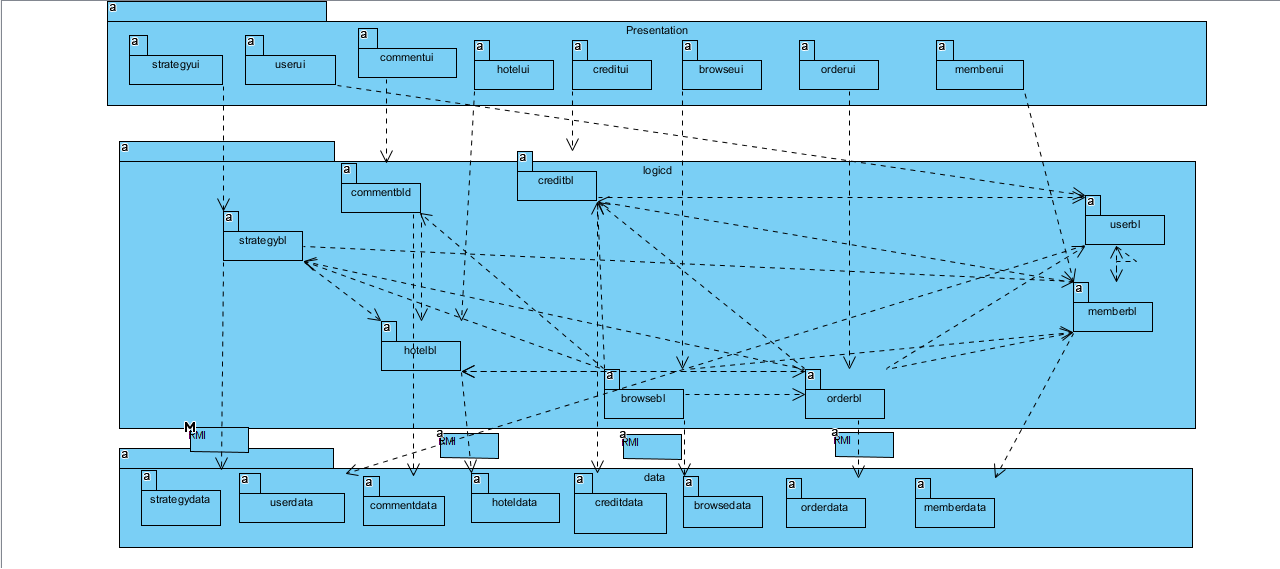
附录 D.3 连锁商店管理系统（MSCS） 软件体系结构描述文档

# 产品概述

参考酒店预订系统用例文档和酒店预订系统软件需求规格说明中对产品的概括描述。

# 逻辑视角

酒店预订系统中，选择了分层体系结构风格，将系统分为3层（展示层、业务逻辑层、数据层）能够很好地示意整个高层抽象。展示层包含GUI界面的实现，业务逻辑层包含业务逻辑处理的实现，数据层负责数据的持久化和访问。分层体系结构的逻辑视角和逻辑设计方案如下图所示。



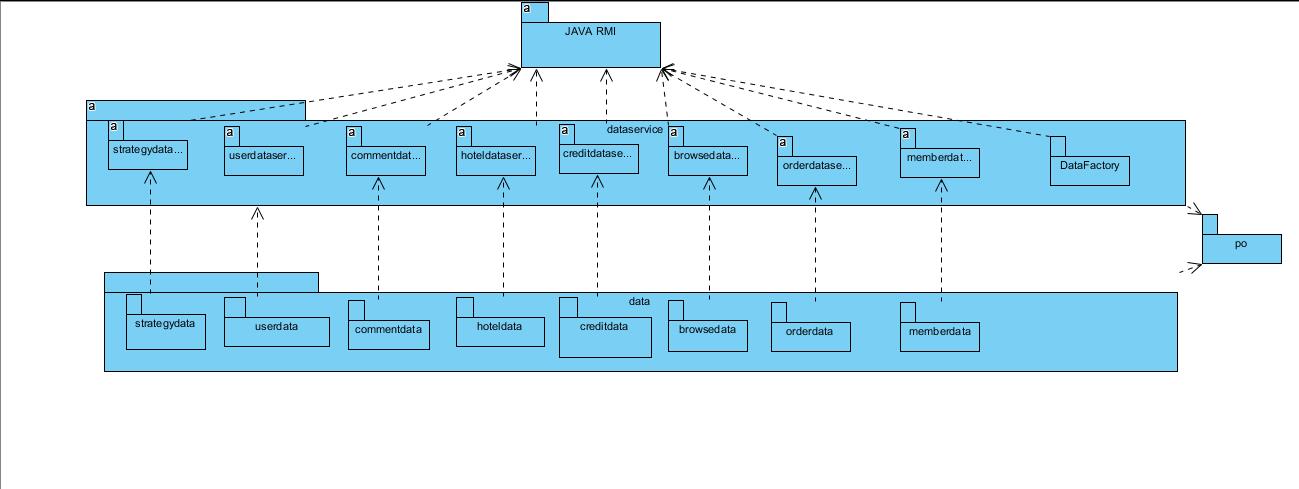
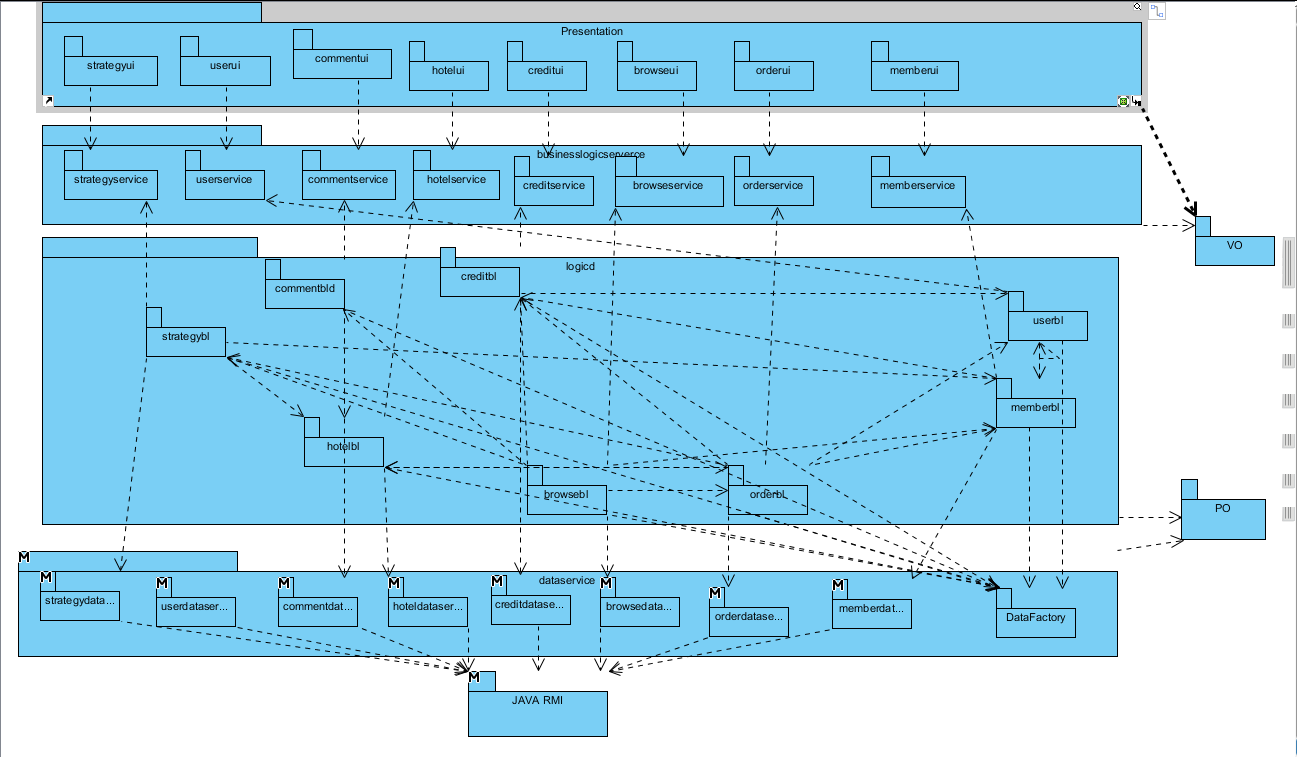
# 组合视角

## 4.1 开发包图

酒店预订系统最终开发包设计如下表所示：

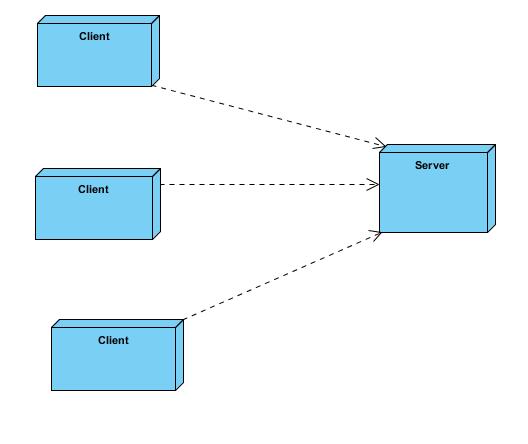
|  |  |
| --- | --- |
| 开发物理包 | 依赖的其他开发包 |
| Mainui | Userui,memberui,commentui,strategyui,orderui,browseui,hotelui,creditui,vo |
| Userui | Userblservice,界面类库包,vo |
| Memberui | memberblservice,界面类库包,vo |
| Commentui | commentblservice,界面类库包,vo |
| Strategyui | strategyblservice,界面类库包,vo |
| Orderui | orderblservice,界面类库包,vo |
| Browseui | browseblservice,界面类库包,vo |
| Hotelui | hotelblservice,界面类库包,vo |
| Creditui | creditblservice,界面类库包,vo |
| Memberblservice |  |
| Commentblservice |  |
| Strategyblservice |  |
| Orderblservice |  |
| Browseblservice |  |
| Hotelblservice |  |
| Creditblservice |  |
| Userblservice |  |
| Userbl | memberdata,memberbl  Po |
| Memberbl | creditdata,creditbl  userdata,userbl  po |
| Commentbl | hoteldata,hotelbl  memberdata,memberbl  po |
| Strategybl | hoteldata,hotelbl  memberdata,memberbl  po |
| Orderbl | hoteldata,hotelbl  Strategybl,Strategydata  memberdata,memberbl  creditdata,creditbl  commentdata,commentbl  po |
| Browsebl | hoteldata,hotelbl  userdata,userbl  memberdata,memberbl  Orderdata,orderbl  Strategybl,Strategydata  Creditdata,creditbl  commentdata,commentbl  po |
| Hotelbl | po |
| Creditbl | memberdata,memberbl,po |
| Userdataservice | JAVA RMI,po |
| Memberdataservice | JAVA RMI,po |
| Commentdataservice | JAVA RMI,po |
| Strategydataservice | JAVA RMI,po |
| Orderdataservice | JAVA RMI,po |
| Browsedataservice | JAVA RMI,po |
| Hoteldataservice | JAVA RMI,po |
| Creditdataservice | JAVA RMI,po |
| Userdata | JAVA RMI,po |
| Memberdata | JAVA RMI,po |
| commentdata | JAVA RMI,po |
| Strategydata | JAVA RMI,po |
| Orderdata | JAVA RMI,po |
| Browsedata | JAVA RMI,po |
| Hoteldata | JAVA RMI,po |
| Creditdata | JAVA RMI,po |
| Vo |  |
| Po |  |
| 界面类库包 |  |
| JAVA RMI |  |

酒店预订系统客户端开发包图、服务器端开发包图如下图所示

服务器端开发包图：

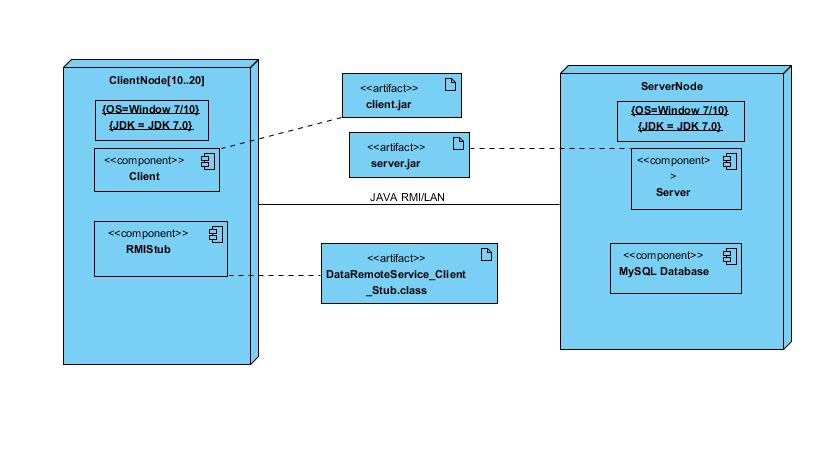
## 4.2 运行时进程

在酒店预订系统中，会有多个客户端进程和一个服务器端进程，其进程图如下图所示，结合部署图，客户端进程是在客户端机器上运行，服务器端进程在服务器端机器上运行



## 4.3 物理部署

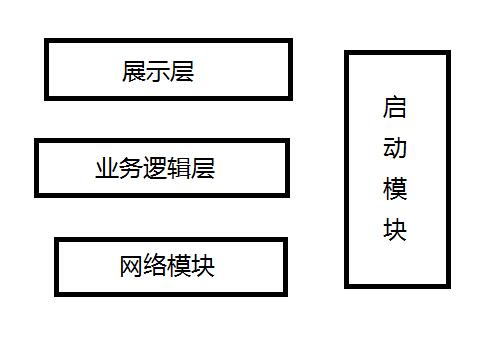
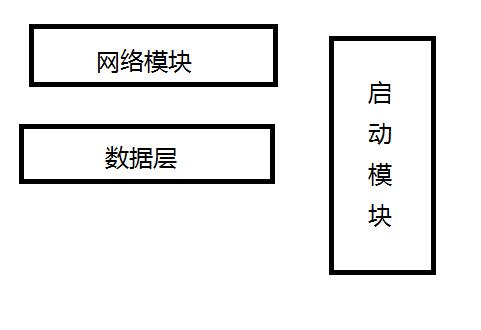
酒店预订系统中客户端构件是放在客户端机器上，服务器端构建是放在服务器端机器上，在客户端节点上，还要部署RMIStub构件。部署图如下图所示



# 接口视角

## 5.1 模块的职责

客户端模块和服务器端模块视图分别如下图所示



客户端各层和服务器各层的职责分别如下表所示：

客户端层：

|  |  |
| --- | --- |
| 层 | 职责 |
| 启动模块 | 负责初始化网络通信机制，启动用户界面 |
| 用户界面层 | 基于窗口的酒店预订客户端用户界面 |
| 业务逻辑层 | 对于用户界面的输入进行响应并进行业务处理逻辑 |
| 客户端网络模块 | 利用JAVA RMI机制查找RMI服务 |

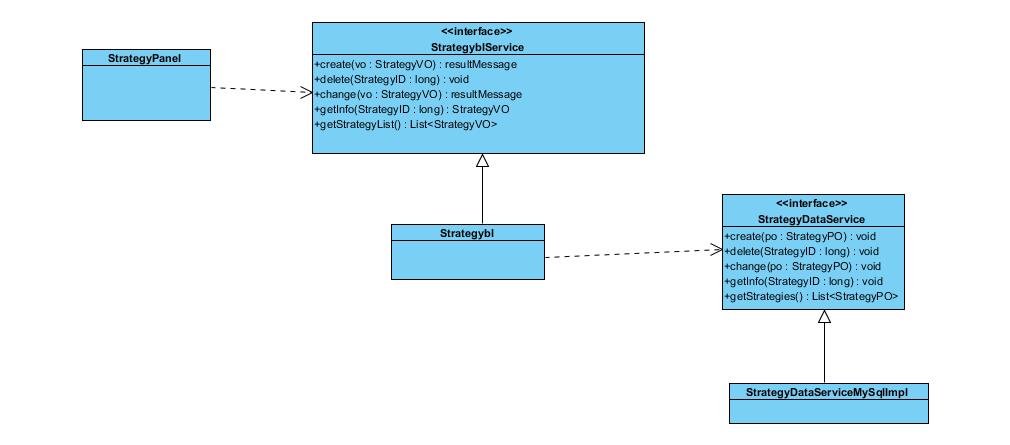
服务器端：

|  |  |
| --- | --- |
| 层 | 职责 |
| 启动模块 | 负责初始化网络通信机制，启动用户界面 |
| 数据层 | 负责数据的持久化和数据访问接口 |
| 服务端网络模块 | 利用JAVA RMI机制开启RMI服务，注册RMI服务 |

层之间的接口如下表所示：

|  |  |  |
| --- | --- | --- |
| 接口 | 服务调用方 | 服务提供方 |
| UserblService  MemberblService  CommentblService  OrderblService  BrowseblService  HotelblService  CreditblService  StrategyblService | 客户端展示层 | 客户端业务逻辑层 |
| UserDataService  MemberDataService  CommentDataService  OrderDataService  BrowseDataService  HotelDataService  CreditDataService  StrategyDataService  DatabaseFactory | 客户端业务逻辑层 | 服务端数据层 |

借用制定促销策略来说明层之间的调用。每一层之间都是由上层依赖了一个接口，而下层只是实现这个接口。StrategyBLService 提供了StrategyUI界面所需要的业务逻辑功能。StrategyDataService提供了数据库的增删改查等操作。这样的实现大大降低了层间的耦合



## 5.2用户界面层的分解

根据需求，系统存在23个用户界面：

一、登录界面

二、客户界面

1.浏览订单信息

2.评价

3.搜索酒店和查看酒店基本信息

4.查看酒店详细信息

5.下订单

6.浏览信用记录

7.注册会员

8.维护基本信息

三、酒店工作人员界面

1.浏览促销策略列表

2.制定促销策略

3.录入客房

4.更新入住和退房

5.浏览订单信息

6.正常执行和补执行

7.维护酒店基本信息

四、网站营销人员界面

1.浏览订单信息

2.撤销订单

3.浏览促销策略列表

4.制定促销策略

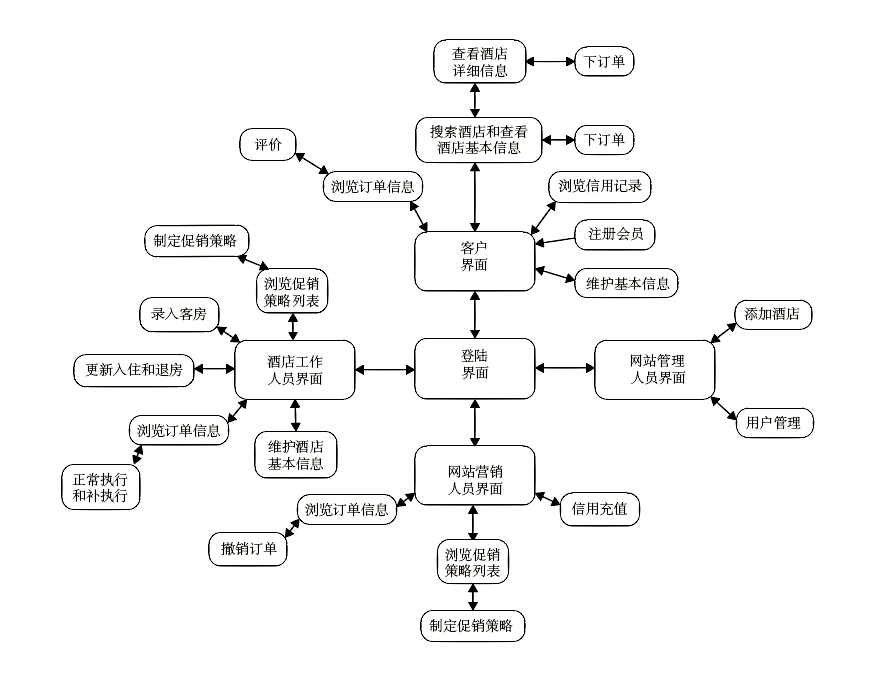
5.信用充值

五、网站管理人员界面

1.添加酒店

2.用户管理

各用户界面的跳转图如下所示



用户界面类即为各界面的init，由MainUI的MainFrame统一调度

### 5.2.1 用户界面层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| LoginFrame | 用户登录和用户注册 |
| ClientFrame | 选择一个客户操作 |
| HotelWorkerFrame | 选择一个酒店工作人员操作 |
| WebSalerFrame | 选择一个网站营销人员操作 |
| WebManagerFrame | 选择一个网站管理人员操作 |
| OrderInfoFrame | 搜索订单信息，显示订单列表 |
| CommentFrame | 评价订单 |
| HotelBasedInfoFrame | 搜索酒店，显示酒店列表及其基本信息 |
| HotelDetailedInfoFrame | 显示酒店详细信息 |
| NewOrderFrame | 生成新的订单 |
| BrowserCreditFrame | 显示信用操作记录 |
| SighupFrame | 会员注册 |
| ChangeClientInfoFrame | 修改客户基本信息 |
| BrowserStrategyFrame | 显示促销策略列表 |
| SetStrategyFrame | 生成新的促销策略 |
| CommitRoomFrame | 录入新的客房 |
| RenewRoomFrame | 更新客房入住信息 |
| OperateFrame | 设置订单执行状态 |
| ChangeHotelInfoFrame | 修改酒店基本信息 |
| CancelOrderFrame | 撤销订单 |
| RechargeCreditFrame | 信用充值 |
| AddHotelFrame | 添加新的酒店 |
| ManageUserFrame | 管理用户 |

### 5.2.2 用户界面层模块的接口规范

用户界面层模块的接口规范如下表所示：

每个用户界面都拥有一个init的初始化接口

|  |  |  |
| --- | --- | --- |
| \*Frame | 语法 | Public init(String []args); |
| 前置条件 | 为登陆界面或者跳转到该界面 |
| 后置条件 | 生成\*Frame和其内部的组件并显示 |

用户界面需要的服务接口

|  |  |
| --- | --- |
| 服务名 | 服务 |
| businesslogicservice.LoginBLservice | 登陆界面的业务逻辑接口 |
| businesslogicservice.\*BLservice | 每个界面都有一个相应的业务逻辑接口 |

### 5.2.3 用户界面模块设计原理

用户界面利用Java Fx相关类库包进行设计

## 5.3 业务逻辑层的分解

业务逻辑层包括多个对界面的业务逻辑处理对象。

### 5.3.1 业务逻辑层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| Userbl | 负责对应于登录界面及相应用户数据更改界面等所需要的服务 |
| Browsebl | 提供浏览各类信息的界面的服务 |
| Orderbl | 负责对应于关于订单的所有操作的服务 |
| Memberbl | 负责对应于会员注册界面及显示相应会员数据界面等所需要的服务 |
| Commentbl | 负责对应于酒店评价以及浏览某一酒店评价的界面等所需要的服务 |
| Creditbl | 负责对应于信用充值和查看记录等界面所需要的服务 |
| Strategybl | 负责对应于制定促销策略界面及显示策略列表和策略信息界面所需要的服务 |
| Hotelbl | 负责维护酒店信息，维护房间信息和添加酒店界面的服务 |

### 5.3.2 业务逻辑层的接口规范

业务逻辑层的接口规范如下：

Userbl

|  |  |  |
| --- | --- | --- |
| 提供的服务(供接口) | | |
| User.login | 语法 | Public ResultMessage login(String username,String password); |
| 前置条件 | Username和password输入符合规范 |
| 后置条件 | 查找是否有相对应的user,根据输入的password返回登录验证的结果 |
| User.register | 语法 | Public ResultMessage register(String username,String password); |
| 前置条件 | 用户未登陆 |
| 后置条件 | 先根据username查询是否注册过，若无，则经过加密后将信息写入数据库 |
| 需要的服务(需接口) | | |
| DatabaseFactory.getUserDatabase | 得到User数据库服务的引用 | |
| UserDataService.insert(UserPO po) | 在数据库中插入po对象 | |
| Member.getInfo(long memberid) | 得到对应的会员信息 | |

Browsebl

|  |  |  |
| --- | --- | --- |
| 提供的服务(供接口) | | |
| Browse.getOrdersInfo | 语法 | Public OrderVO getOrdersInfo(long userid,OrderType type); |
| 前置条件 | 用户已登录，且这里用户为客户 |
| 后置条件 | 根据对应的orderType参数查找对应的Order信息并返回 |
| Browse.getHotelOrdersInfo | 语法 | Public OrderVO getHotelOrdersInfo(long hotelid,OrderType type); |
| 前置条件 | 酒店工作人员已登录 |
| 后置条件 | 根据对应的orderType参数查找对应的Order信息并返回 |
| Browse.getWEBOrdersInfo | 语法 | Public OrderVO getWEBOrdersInfo(); |
| 前置条件 | 网站营销人员已登录 |
| 后置条件 | 查找对应类型的当天当时间异常订单信息并返回 |
| Browse.getCreditInfo(long userid) | 语法 | Public CreditVO getCreditInfo(long userid); |
| 前置条件 | 客户已登录 |
| 后置条件 | 根据对应的user查找这个user的所有的信用纪录并返回 |
| Browse.getHotels | 语法 | Public List<HotelVO> getHotels(RuleVO vo); |
| 前置条件 | 无 |
| 后置条件 | 根据RuleVO的规则，查找相应的酒店列表信息，对数据处理后返回相应粗略酒店信息 |
| Browse.getHotel | 语法 | Public HotelVO getHotel(long hotelid); |
| 前置条件 | 无 |
| 后置条件 | 根据对应的id,查询相应酒店的详细信息，处理后返回 |
| Browse.getStrategyInfo | 语法 | Public StrategyVO getStrategyInfo(int orderid); |
| 前置条件 | 用户选择一个订单 |
| 后置条件 | 根据对应订单，查看该订单生效的策略信息 |
| Browse.getMemberInfo | 语法 | Public MemberVO getMemberInfo(long memberid); |
| 前置条件 | 用户已登录 |
| 后置条件 | 根据对应的id，查询相应用户的会员信息并返回 |
| Browse.getBrowseInfo | 语法 | Public List<BrowseVO> getBrowseInfo(long userid); |
| 前置条件 | 无 |
| 后置条件 | 查询对应用户的浏览记录并返回 |
| Browse.clear | 语法 | Public void clear(long userid); |
| 前置条件 | 无 |
| 后置条件 | 清空用户的浏览记录 |
| 需要的服务(需接口) | | |
| DatabaseFactory.getBrowseDatabase | 得到Browse数据库服务的引用 | |
| BrowseDataService.insert(BrowsePO po) | 在Browse数据库中插入po（搜索记录） | |
| BrowseDateService.getInfo(userid) | 返回一个用户的浏览/搜索记录 | |
| BrowseDataService.clear(long userid) | 清空一个用户的浏览搜索记录 | |
| Hotel.getHotels(RulePO po) | 得到基于rule的酒店信息列表 | |
| Hotel.getInfo(long Hotelid) | 得到hotel的详细信息 | |
| Credit.getInfo(long userID) | 得到user的信用信息 | |
| Order.getuserInfo(long userid,int method) | 得到user的对应method的对应类别的订单信息 | |
| Order.getHotelInfoloing hotelid,int method) | 得到hotel的对应method的对应类别的订单信息 | |
| Order.getWEBInfo() | 得到当天当前时间前所有的异常订单信息 | |
| Member.getInfo(long userid) | 得到对应的会员信息 | |
| Strategy.getInfo(long orderid) | 得到对应订单的策略信息 | |

Orderbl

|  |  |  |
| --- | --- | --- |
| 提供的服务(供接口) | | |
| Order.getuserInfo | 语法 | Public List<OrderVO> getuserInfo(long userid,OrderType type); |
| 前置条件 | 用户已登录，且这里用户为客户 |
| 后置条件 | 根据对应的type 查找对应类型的对应用户订单信息 |
| Order.getHotelInfo | 语法 | Public List<OrderVO> getHotelOrdersInfo(long hotelid,OrderType type); |
| 前置条件 | 酒店管理人员已登录 |
| 后置条件 | 根据对应的type 查找对应类型的对应用户订单信息 |
| Order.getWEBInfo | 语法 | Public List<OrderVO> getWEBOrdersInfo(); |
| 前置条件 | 网站营销人员已登录 |
| 后置条件 | 根据对应的TYPE 查找对应类型的当天当时间异常订单信息 |
| Order.create | 语法 | Public ResultMessage create(OrderVO vo); |
| 前置条件 | 客户已登录 |
| 后置条件 | 创建订单并存储 |
| Order.Abnormal | 语法 | Public void abnormal(OrderVO vo); |
| 前置条件 | 订单由未执行变为异常 |
| 后置条件 | 改变订单状态并存储 |
| Order.userCancel | 语法 | Public ResultMessage userCancel(OrderVO vo); |
| 前置条件 | 用户已登录 |
| 后置条件 | 所orderid所对应订单为未执行订单，撤销该订单 |
| Order.execute | 语法 | Public ResultMessage(OrderVO vo); |
| 前置条件 | 酒店工作人员已登录 |
| 后置条件 | 根据对应订单，判断订单状态，执行或补登记执行订单，并作出信用值的相应改变 |
| Order.WEBCancel | 语法 | Public ResultMessage WEBCancel(OrderVO vo); |
| 前置条件 | 网站营销人员已登录且线下已判断订单可撤销 |
| 后置条件 | 根据对应的订单号，撤销订单，并返回状态 |
| Order.getTotal() | 语法 | Public double getTotal(OrderVO vo); |
| 前置条件 | 无 |
| 后置条件 | 计算订单总价，并返回 |
| 需要的服务(需接口) | | |
| DatabaseFactory.getOrderDatabase | 得到Order数据库服务的引用 | |
| OrderDataService.insert(OrderPO po) | 在Order数据库中插入po | |
| OrderDataService.changeStatus(OrderPO po); | 改变订单的状态 | |
| OrderDataService.getInfo(long orderid); | 获取订单的信息 | |
| Hotel.getRooms(long hotelid) | 得到该hotel的房间信息 | |
| Hotel.setRoomInfo(RoomVO vo) | 锁定该酒店的相应的房间 | |
| Credit.add/decrese(long userid,int value); | 增加/减少该user的信用值信息 | |
| Member.getInfo(userid) | 得到对应的会员信息 | |
| Strategy.getInfo(orderid) | 得到对应订单的策略信息 | |

Memberbl

|  |  |  |
| --- | --- | --- |
| 提供的服务(供接口) | | |
| Member.register | 语法 | Public resultmessage register(UserVO vo); |
| 前置条件 | 用户已经登录 |
| 后置条件 | 先检查信用值,若信用值达到相应标准，添加相应的会员信息 |
| Member.cancel | 语法 | Public resultMessage cancel(UserVO vo); |
| 前置条件 | 用户已经是会员但信用值不足 |
| 后置条件 | 更改会员状态 |
| Member.getInfo | 语法 | Public resultMessage getInfo(long memberID) |
| 前置条件 | 用户已经登录 |
| 后置条件 | 根据对应的memberID得到相应的会员信息 |
| 需要的服务(需接口) | | |
| DatabaseFactory.getMemberDatabase | 得到Member数据库服务的引用 | |
| MemberDataService.insert(MemberPO po) | 在数据库中增加po对象 | |
| MemberDataService.cancel(MemberPO po) | 取消member的vip会员资格 | |
| MemberDataService.changeInfo(MemberPO po) | 更改member的基本信息 | |
| MemberDataService.getInfo(long userID) | 得到member的基本信息 | |

Commentbl

|  |  |  |
| --- | --- | --- |
| 提供的服务(供接口) | | |
| Comment.getInfo | 语法 | Public List<Comment VO> getInfo(long hotelid); |
| 前置条件 | 客户或酒店工作人员已登录 |
| 后置条件 | 根据对应的hotelid查询酒店评价并返回 |
| Comment.review | 语法 | Public ResultMessage review(Comment VO); |
| 前置条件 | 客户对已执行未评论订单/已执行已评论订单进行评价或者酒店工作人员对订单评价进行反馈 |
| 后置条件 | 把评论存储到comment数据库并返回评论结果 |
| 需要的服务(需接口) | | |
| DatabaseFactory.getCommentDatabase | 得到Comment数据库服务的引用 | |
| CommentDataService.insert(CommentPO po) | 在Comment数据库中插入po | |
| CommentDataService.getInfo(long hotelID) | 在Comment数据库中得到一个po | |

Creditbl

|  |  |  |
| --- | --- | --- |
| 提供的服务(供接口) | | |
| Credit.getInfo | 语法 | Public List<CreditVO> getInfo(long userID); |
| 前置条件 | 客户已登录 |
| 后置条件 | 返回信用记录 |
| Credit.recharge | 语法 | Public ResultMessage recharge(int value); |
| 前置条件 | 网站营销人员已登录和授权 |
| 后置条件 | 存储并返回消息（成功或失败） |
| Credit.add | 语法 | Public void add(userID id,int value); |
| 前置条件 | 订单已执行或者酒店工作人员补登记订单或者网站工作人员撤销异常订单 |
| 后置条件 | 信用值增加并存储相应的结果 |
| Credit.decrease | 语法 | Public void decrease(userID id,int value); |
| 前置条件 | 订单由未执行状态变为异常状态 |
| 后置条件 | 信用值减少并存储相应的结果 |
| 需要的服务(需接口) | | |
| DatabaseFactory.getCreditDatabase | 得到Credit数据库服务的引用 | |
| CreditDataService.add(long userID,int value) | 增加数据库中的相应信用值 | |
| CreditDataService.decrease(long userID,int value) | 减少数据库中的相应信用值 | |
| CreditDataService.insert(CreditPO po) | 在数据库中插入CreditPO对象 | |

Strategybl

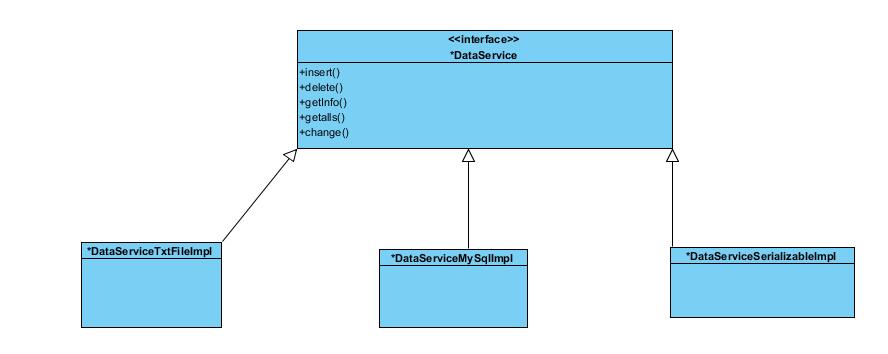
|  |  |  |
| --- | --- | --- |
| 提供的服务(供接口) | | |
| Strategy.create | 语法 | Public resultMessage create(StrategyVO vo); |
| 前置条件 | 工作人员已登录和授权 |
| 后置条件 | 返回相应结果，包括成功、失败和失败原因（如时间不匹配） |
| Strategy.delete | 语法 | Public void delete(long strategyID); |
| 前置条件 | 工作人员已登录和授权 |
| 后置条件 | 删除对应数据库里的信息 |
| Strategy.change | 语法 | Public resultMessage change(StrategyVO vo); |
| 前置条件 | 工作人员已登录和授权 |
| 后置条件 | 返回相应结果，包括成功、失败和失败原因（如时间不匹配），如果成功则更改对应数据库里的信息 |
| Strategy.getInfo | 语法 | Public SrategyVO(long strategyID); |
| 前置条件 | 工作人员已登录和授权 |
| 后置条件 | 返回对应数据库里的信息 |
| Strategy.getStrategyList | 语法 | Public List<StrategyVO>() |
| 前置条件 | 工作人员已登录和授权 |
| 后置条件 | 返回策略信息 |
| 需要的服务(需接口) | | |
| DatabaseFactory.getStrategyDatabase | 得到Strategy数据库服务的引用 | |
| Hotel.getInfo | 得到酒店信息 | |
| Member.getInfo(long memberID) | 得到对应的会员信息 | |
| StrategyDataService.creat(StrategyPO po) | 在数据库里创建一条策略记录 | |
| StrategyDataService.delete(long StrategyID) | 在数据库里删除一条策略记录 | |
| StrategyDataService.change(StategyPO po) | 在数据库里更改一条策略记录 | |
| StrategyDataService.getInfo(long StrategyID) | 查找数据库中的策略 | |
| StrategyDataService.getStrategys() | 返回数据库里的所有策略记录 | |

Hotelbl

|  |  |  |
| --- | --- | --- |
| 提供的服务(供接口) | | |
| Hotel.GetHotelInfo | 语法 | Public HotelVO GetHotelInfo(HotelID id); |
| 前置条件 | 无 |
| 后置条件 | 无 |
| Hotel.SetHotelInfo | 语法 | Public ResultMessage SetHotelInfo(HotelVO vo); |
| 前置条件 | 酒店工作人员已登录 |
| 后置条件 | 更新酒店信息 |
| Hotel.SetRoomInfo | 语法 | Public ResultMessage SetRoomInfo(HotelID id, RoomVO vo); |
| 前置条件 | 酒店工作人员已登录 |
| 后置条件 | 更新房间信息 |
| Hotel.AddHotel | 语法 | Public ResultMessage AddHotel(HotelVO vo); |
| 前置条件 | 网站管理人员已登录 |
| 后置条件 | 生成新的酒店信息 |
| Hotel.AddRoom | 语法 | Public ResultMessage AddRoom(RoomVO vo); |
| 前置条件 | 酒店工作人员已登录 |
| 后置条件 | 生成新的房间信息 |
| 需要的服务(需接口) | | |
| DatabaseFactory.getHotelDatabase | 得到Hotel数据库服务的引用 | |
| HotelDataService.insertHotel(HotelVO vo) | 在数据库中插入vo对象 | |
| HotelDataService.getHotelInfo(HotelID id) | 得到Hotel信息 | |
| HotelDataService.setHotelInfo(HotelVO vo) | 更改Hotel信息 | |
| HotelDataService.insertRoom(RoomVO vo) | 在数据库中插入vo对象 | |
| HotelDataService.getRoomInfo(HotelID id) | 得到Room信息 | |
| HotelDataService.setRoomInfo(RoomVO vo) | 更改Room信息 | |

## 5.4 数据层的分解

数据层主要给业务逻辑提供数据访问服务，包括对于持久化数据的增删改查，\*bl业务逻辑所需要的服务主要由\*DataService接口提供。由于持久化数据的保存可能存在多种形式：txt 文件，序列化文件，数据库等，所以抽象了数据服务，如下图所示：



### 5.4.1 数据层模块的职责

对于\*DataService数据层接口，其向下模块的实现具体描述见下表

|  |  |
| --- | --- |
| 模块 | 职责 |
| \*DataService | 持久化数据库的接口，提供集体载入、集体保存，增删改查等服务 |
| \*DataServiceTxtFileImpl | 基于Txt文件的持久化数据库的接口，提供集体载入、集体保存，增删改查等服务 |
| \*DataServiceSerializableFileImpl | 基于序列化文件的持久化数据库的接口，提供集体载入、集体保存，增删改查等服务 |
| \*DataServiceMySqlImpl | 基于MySql数据库的持久化数据库的接口，提供集体载入、集体保存，增删改查等服务 |

### 5.4.2 数据层模块的接口规范

数据层模块的接口规范如下表所示：

UserDataService

|  |  |  |
| --- | --- | --- |
| 提供的接口 | | |
| UserDataService.insert | 语法 | Public void insert(UserPO po) throws RemoteException |
| 前置条件 | 同样ID的po在数据库中不存在 |
| 后置条件 | 在数据库中增加一条PO记录 |
| UserDataService.getInfo | 语法 | Public UserPO getInfo(long userID) throws RemoteException |
| 前置条件 | 该ID的po在数据库中存在 |
| 后置条件 | 查询并返回这条PO |

BrowseDataService

|  |  |  |
| --- | --- | --- |
| 提供的接口 | | |
| BrowseDataService.insert | 语法 | Public void insert(BrowsePO po) throws RemoteException |
| 前置条件 | 同样ID的po在数据库中不存在 |
| 后置条件 | 在数据库中增加一条PO记录 |
| BrowseDataService.getInfo | 语法 | Public List<BrowsePO> getInfo(long userID) throws RemoteException |
| 前置条件 | 该ID至少有一条po在数据库中存在 |
| 后置条件 | 查询并返回该ID的所有PO |
| BrowseDataService.clear | 语法 | Public void clear(long userid) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 清空数据库中该ID的所有PO |

OrderDataService

|  |  |  |
| --- | --- | --- |
| OrderDataService.insert | 语法 | Public void insert(OrderPO po) throws RemoteException |
| 前置条件 | 同样ID的po在数据库中不存在 |
| 后置条件 | 在数据库中增加一条PO记录 |
| OrderDataService.changeInfo | 语法 | Public void changeInfo(OrderPO po) throws RemoteException |
| 前置条件 | 同样ID的PO在数据库中存在 |
| 后置条件 | 更新这条PO |
| OrderDataService.getInfo | 语法 | Public OrderPO getInfo(long OrderID) throws RemoteException |
| 前置条件 | 该ID的po在数据库中存在 |
| 后置条件 | 查询并返回这条PO |

MemberDataService

|  |  |  |
| --- | --- | --- |
| 提供的接口 | | |
| MemberDataService.insert | 语法 | Public void insert(MemberPO po) throws RemoteException |
| 前置条件 | 同样ID的po在数据库中不存在 |
| 后置条件 | 在数据库中增加一条PO记录 |
| MemberDataService.cancel | 语法 | Public void cancel(MemberPO po) throws RemoteException |
| 前置条件 | 同样ID的PO在数据库中存在 |
| 后置条件 | 删除这条PO |
| MemberDataService.getInfo | 语法 | Public MemberPO getInfo(long userID) throws RemoteException |
| 前置条件 | 该ID的po在数据库中存在 |
| 后置条件 | 查询并返回这条PO |
| MemberDataService.changeInfo | 语法 | Public void changeInfo(MemberPO po) throws RemoteException |
| 前置条件 | 该ID的po在数据库中存在 |
| 后置条件 | 更新这条PO |

CommentDataService

|  |  |  |
| --- | --- | --- |
| 提供的接口 | | |
| CommentDataService.insert | 语法 | Public void insert(CommentPO po) throws RemoteException |
| 前置条件 | 同样ID的po在数据库中不存在 |
| 后置条件 | 在数据库中增加一条PO记录 |
| CommentDataService.i=getInfo | 语法 | Public CommentPO getInfo(long hotelID) throws RemoteException |
| 前置条件 | 同样ID的po在数据库存在 |
| 后置条件 | 返回这条PO数据 |

HotelDataService

|  |  |  |
| --- | --- | --- |
| 提供的接口 | | |
| HotelDataService.insertHotel | 语法 | Public void insertHotel (HotelVO vo) throws RemoteException |
| 前置条件 | 同样ID的vo在数据库中不存在 |
| 后置条件 | 在数据库中增加一条VO记录 |
| HotelDataService.getHotelInfo | 语法 | Public HotelVO getHotelInfo (long HotelID) throws RemoteException |
| 前置条件 | 同样ID的VO在数据库中存在 |
| 后置条件 | 查询并返回这条VO |
| HotelDataService.setHotelInfo | 语法 | Public void setHotelInfo (HotelVO vo) throws RemoteException |
| 前置条件 | 该ID的vo在数据库中存在 |
| 后置条件 | 更新这条VO |
| HotelDataService.insertRoom | 语法 | Public void insertRoom (RoomVO vo) throws RemoteException |
| 前置条件 | 同样ID的vo在数据库中不存在 |
| 后置条件 | 在数据库中增加一条VO记录 |
| HotelDataService.getRoomInfo | 语法 | Public RoomVO getRoomInfo (long RoomID) throws RemoteException |
| 前置条件 | 同样ID的VO在数据库中存在 |
| 后置条件 | 查询并返回这条VO |
| HotelDataService.setRoomInfo | 语法 | Public void setRoomInfo (RoomVO vo) throws RemoteException |
| 前置条件 | 该ID的vo在数据库中存在 |
| 后置条件 | 更新这条VO |

# 信息视角

## 6.1 数据持久化对象

系统的PO类就是对应的相关的实体类：

UserPO类包含用户的用户名、密码等信息；

MemberPO类包含用户的基本信息，包括ID，用户联系方式，名称，信用值等。

HotelPO类包含酒店的基本信息，包括酒店商圈，地址，星级，简介，设施服务，酒店ID，RoomPO等

OrderPO类包含订单的状态，订单的基本信息，订单ID等

CreditPO类包含信用记录，信用值等信息

BrowsePO类包含浏览记录，包括时间，浏览（搜索）内容

RoomPO类包含房间信息，包括房间数量，类型，状态等

StrategyPO类包含策略ID，影响时间，影响酒店，影响房间，策略类型，折扣信息等内容

CommentPO类包含评价记录，包括评价内容，评价用户，评价酒店，评价时间，评分信息等。

UserPO 定义：

Public class UserPO implement Serializable {

long id;

String username;

String password;

Role role;

Public UserPO(int id,String username,Stirng password,Role role){

This.id = id;

This.username = username;

This.password = password;

This.role = role;

}

Public void setRole(Role role){

This.role = role;

}

Public Role getRole(){return this.role};

Public long getID(){return this.id};

}

OrderPO 定义：

Public class orderPO implement Serializable {

Long id;

Status status;

String begintime;

String endtime;

String actualBeginTime;

Int expectedpeople;

Int actualPeople;

Boolean actualHaveChild;

Boolean isHaveChild;

long hotelid;

RoomInfo room;

…

Public OrderPO(String beginTime,String endTime,int expectedpeople,boolean isHaveChild,long hotelid,RoomInfo room){

This.begintime = begintime;

…

This.room = room;

}

}

BrowsePO 定义：

Public class BrowsePO implement Serializable {

Int browseID;

Int userID;

Int toUserID;

Time time;

Public BrowsePO(int userID,int toUserID,Time time){

Initialize browseID;

this.userID = userID;

This.toUserID = toUserID;

This.time = time;

}

Get/set method;

}

MemberPO 定义：

Public class MemberPO implement Serializable {

long id;

String memberbirthday;

String companyname;

String name;

Long phoneNumber;

Boolean isVIP;

Int VIPtype;

Role role;

Public MemberPO(String name,Long phoneNumber){

//create new id;

This.name=name;

this.phoneNumber = phoneNumber;

}

Public void register(int type,String info){

If (type==1) this.memberbirthday=info;

If (type==2) this.companyname=info;

this.VIPtype=type;

this.isVIP=true;

}

Get/set

}

CommentPO定义：

Public class CommentPO implement Serializable{

Int grade;

String comment;

Long time;

Long userid;

Public CommentPO(int grade,String comment,long userid){

Time = this.getTime;

This.userid = userid;

This.grade = grade;

This.comment = comment;

}

Get/set

}

HotelPO 定义：

Public class HotelPO implement Serializable {

long ID;

String name;

String circle;

String address;

String shoutcut;

List<String> equipment;

List<String> service;

int star;

long score;

RoomPO roompo;

…

Public HotelPO(long id,String name,String circle,String address,String shoutcut,List<String> equipment,List<String> service,int star,long score){

This.id = id

……

This.score = score;

}

Public void setRoomPO(RoomPO roompo){ this.roompo = roompo;}

Get/set

}

RoomPO 定义：

Public class RoomPO implement Serializable {

long ID;

Status status;

Int number;

long price;

Type type;

…

Public RoomPO(Status status,Int number,long price,Type type){

This.status = status;

…

This.type = type;

}

Get/set

}

## 6.2 Txt 持久化格式

以房间信息为例：

第一行记录酒店id，第二行开始为房间信息，格式为 type num status

如

双人房 100 空闲

单人房 53 在住

单人房 47 空闲

## 6.3 数据库表

数据库中包含 User表，Comment表，Member表，Hotel表，Room表，Browse表，Order表等表。