|  |  |
| --- | --- |
| 屏幕快照%202016-10-08%2018  **互联网酒店预订系统**  软件体系结构设计模型文档 | 鄢煜民，何慧娴，顾恺嘉，黄潇  [2016-10-08] |

**更新历史**

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **日期** | **变更原因** | **版本号** |
| 全体成员 | 2014/10/12 | 初步版，体系结构初步模型 | V0.1初步版 |
| 全体成员 | 2014/10/20 | 添加接口视角和信息视角 | V0.2 |
| 全体成员 | 2014/10/25 | 根据实际情况更改部分接口的方法 | V0.3 |
| 全体成员 | 2014/10/26 | 评审后的正式版 | V1.0正式版 |

**目录**

[1.引言 4](#_Toc20969)

[1.1编制目的 4](#_Toc10270)

[1.2词汇表 4](#_Toc3471)

[1.3参考资料 4](#_Toc19837)

[2.产品描述 4](#_Toc926)

[3. 逻辑视角 5](#_Toc27500)

[4. 组合视角 7](#_Toc4512)

[4.1开发包图 7](#_Toc20894)

[4.2运行时进程 11](#_Toc22402)

[4.3物理部署 12](#_Toc6958)

[5. 接口视角 12](#_Toc6992)

[5.1模块的职责 12](#_Toc6122)

[5.2用户界面层的分解 14](#_Toc20425)

[5.2.1用户界面层模块的职责 16](#_Toc28953)

[5.2.2用户界面层模块的接口规范 16](#_Toc31890)

[5.2.3用户界面模块设计原理 16](#_Toc27994)

[5.3.1业务逻辑层模块的职责 17](#_Toc32024)

[5.3.2业务逻辑层模块的接口规范 18](#_Toc5259)

[5.4.1数据层模块的职责 33](#_Toc29532)

[5.4.2数据层模块的接口 36](#_Toc15322)

[6. 信息视角 47](#_Toc30268)

[6.1数据持久化对象 47](#_Toc879)

[6.2 Txt持久化格式 50](#_Toc32509)

**1.引言**

**1.1编制目的**

本报告详细完成对互联网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 |
| 用户 | 使用该系统预订酒店的用户 |  |
| 系统用户 | 使用该系统的用户 | 包括用户、酒店工作人员、网站管理人员、网站营销人员 |

**1.3参考资料**

1. 互联网酒店预订系统 用例文档第三版
2. 互联网酒店预订系统 软件需求规格说明文档最终版

**2.产品描述**

参考互联网酒店预订系统用例文档和互联网酒店预订系统软件需求规格说明文档中对产品的概括描述。

互联网酒店预订系统开发的目标是便于人们完成日常的酒店预订，包括酒店的订单管理，用户的订单管理，网站人员的管理等等等等。

通过互联网酒店预订系统的应用，期望为全面展现任一个地域内的酒店信息，帮助顾客进行更好地比较与选择。

1. **逻辑视角**

互联网酒店预订系统中，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页面的实现，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图3-1和图3-2所示。



图3-1 参照体系结构风格的包图表达逻辑视角

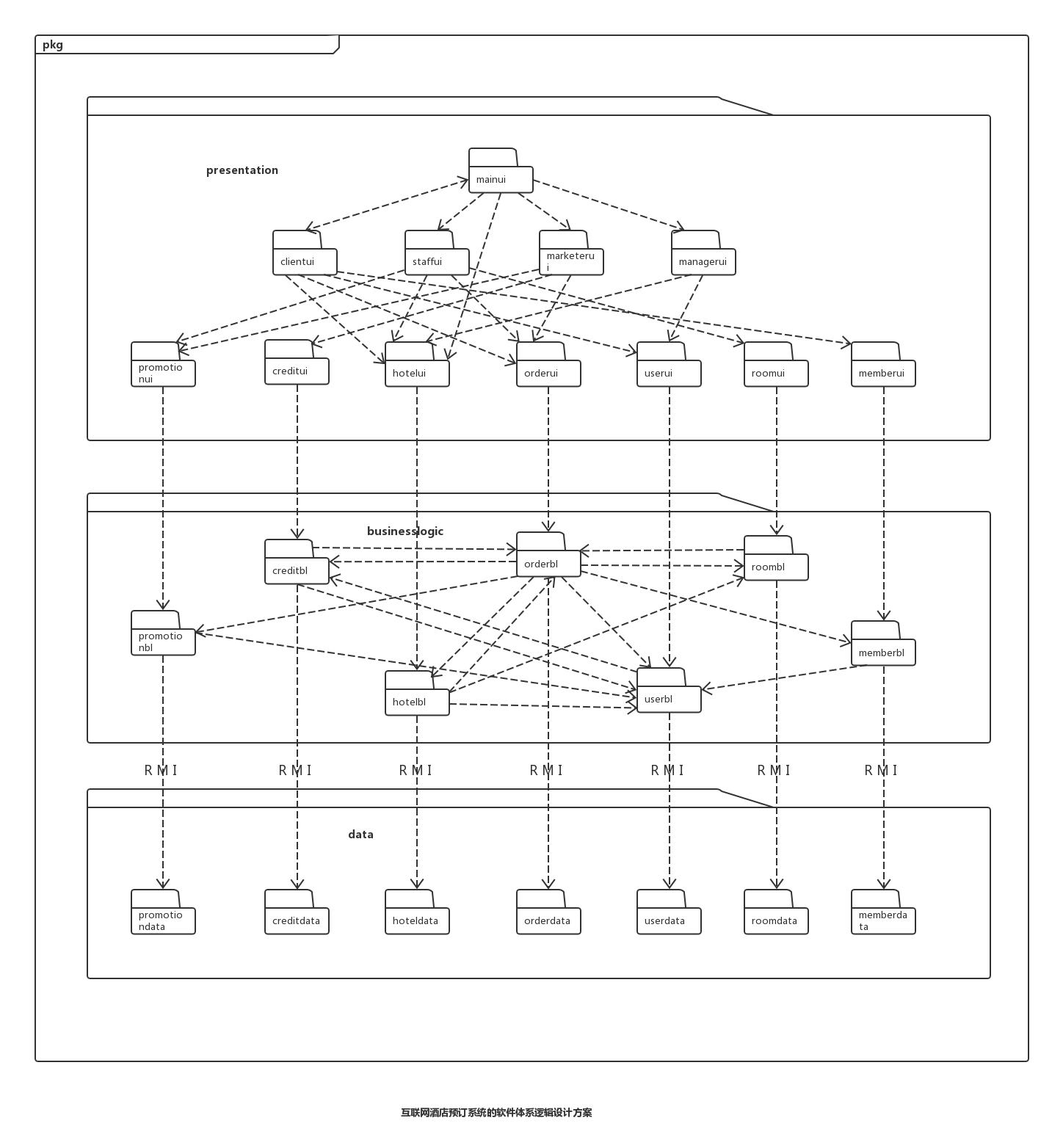


图3-2 软件体系结构逻辑设计方案

1. **组合视角**

**4.1开发包图**

互联网酒店预订系统的最终开发包设计如表4-1-1所示。

**表4-1-1 企业进销存管理系统的最终开发包设计**

|  |  |
| --- | --- |
| **开发（物理）包** | **依赖的其他开发包** |
| mainui | clientui,staffui,marketerui,manager,hotelui,vo |
| clientui | hotelui,orderui,userui,memberui,roomui,界面类库包 |
| staffui | promotionui,hotelui,orderui,room,界面类库包 |
| marketerui | promotionui,creditui,orderui,界面类库包 |
| managerui | hotelui,userui,界面类库包 |
| promotionui | promotionblservice,界面类库包,vo |
| promotionblservice |  |
| promotionbl | promotionblservice,promotiondataservice,userbl,vo |
| promotiondataservice | Java RMI,po |
| promotiondata | Java RMI,po,databaseutility |
| creditui | creditservice,界面类库包,vo |
| creditblservice |  |
| creditbl | creditblservice,creditdataservice,userbl,orderbl,vo |
| creditdataservice | Java RMI,po |
| creditdata | Java RMI,po,databaseutility |
| hotelui | hotelservice,界面类库包 |
| hotelblservice |  |
| hotelbl | hotelblservice,hoteldataservice,userbl,orderbl,roombl,vo |
| hoteldata | Java RMI,po |
| hoteldataservice | Java RMI,po,databaseutility |
| orderui | orderblservice,界面类库包,vo |
| orderblservice |  |
| orderbl | orderblservice,orderdataservice,promotionbl,creditbl,hotelbl,roombl,userbl,memberbl,vo |
| orderdataservice | Java RMI,po |
| orderdata | Java RMI,po,databaseutility |
| userui | userblservice,界面类库包,vo |
| userblservice |  |
| userbl | userblservice,userdataservice,creditbl,vo |
| userdataservice | Java RMI,po |
| userdata | Java RMI,po,databaseutility |
| roomui | roomblservice,界面类库包,vo |
| roomblservice |  |
| roombl | roomblservice,roomdataservice,order,vo |
| roomdataservice | Java RMI,po |
| roomdata | Java RMI,po,databaseutility |
| memberui | memberblservice,界面类库包,vo |
| memberblservice |  |
| memberbl | memberblservice,memberdataservice,userbl,vo |
| memberdataservice | Java RMI,po |
| memberdata | Java RMI,po,databaseutility |
| vo |  |
| po |  |
| utilitybl |  |
| datafactoryservice | Java RMI,po |
| 界面类库包 |  |
| Java RMI |  |
| dataioutility | Java IO |

互联网酒店预订系统客户端开发包图如图4-1-1所示，服务器端开发包图如图4-1-2所示。



图4-1-1 互联网酒店预订系统客户端开发包图



图4-1-2 互联网酒店预订系统服务器端开发包图

**4.2运行时进程**

在互联网酒店预订系统中，会有多个客户端进程和一个服务器端进程，其进程图如图4-2-1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。



图4-2-1 进程图

**4.3物理部署**

互联网酒店预订系统中客户端构件是放在客户端机器上，服务器端构件是放在服务器端机器上。在客户端节点上，还要部署RMIStub构件。由于Java RMI构件属于JDK 1.7的一部分。所以，在系统JDK环境已经设置好的情况下，不需要再独立部署。部署图如图4-3-1所示。



图4-3-1 部署图

1. **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图5-1-1和图5-1-2所示。客户端各层和服务端各层的职责分别入表5-1-1和表5-1-2所示。



图5-1-1 客户端模块视图



图5-1-2 服务器端模块视图

**表5-1-1 客户端各层的职责**

|  |  |
| --- | --- |
| **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 | 基于窗口的互联网酒店预订系统客户端用户界面 |
| 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 |

**表5-1-2 服务器端各层的职责**

|  |  |
| --- | --- |
| **层** | **职责** |
| 启动模块 | 负责初始化网络通信机制，启动服务器 |
| 数据层 | 负责数据的持久化及数据访问接口 |
| 服务器端网络模块 | 利用Java RMI机制开启RMI服务，注册RMI服务 |

每一层只是使用下方直接接触的层。层与层之间仅仅是通过接口的调用来完成的。层之间调用的接口如表5-1-3所示。

**表5-1-3 层之间调用的接口**

|  |  |  |
| --- | --- | --- |
| **接口** | **服务调用方** | **服务提供方** |
| Promotionblservice  Creditblservice  Hotelblservice  Orderblservice  Userblservice  Memberblservice | 客户端展示层 | 客户端业务逻辑层 |
| Promotiondataservice  Creditdataservice  Hoteldataservice  Orderdataservice  Userdataservice  Memberdataservice | 客户端业务逻辑层 | 服务端数据层 |

**5.2用户界面层的分解**

根据需求，系统存在23个用户界面：界面跳转如图5-2-1所示。



图5-2-1 用户界面跳转

服务器端和客户端的用户界面设计接口是一致的，只是具体的页面不一样。

用户界面类如图5-2-2所示。



图5-2-2 用户界面类

5.2.1用户界面层模块的职责

如表5-2-1-1所示为用户界面层模块的职责。

**表5-2-1-1 用户界面层模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| MainFrame | 界面Frame，负责界面的显示和界面的跳转 |

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表5-2-2-1所示。

**表5-2-2-1 用户界面层模块的接口规范**

|  |  |  |
| --- | --- | --- |
| MainFrame | 语法 | init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示Frame以及LoginPanel |

用户界面层需要的服务接口如表5-2-2-2所示。

**表5-2-2-2 用户界面层模块需要的服务接口**

|  |  |
| --- | --- |
| **服务名** | **服务** |
| businesslogicservice.LoginBLService | 登录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的Swing和AWT库来实现。

**5.3业务逻辑层的分解**

业务逻辑层包括多个针对界面的业务逻辑处理对象。业务逻辑层的设计如图5-3-1所示。



图5-3-1 业务逻辑层的设计

5.3.1业务逻辑层模块的职责

业务逻辑层模块的职责如表5-3-1-1所示。

**表5-3-1-1 业务逻辑层模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| promotionbl | 负责实现促销策略的制定和实施所需要的服务 |
| creditbl | 负责实现用户信用的录入和扣除所需要的服务 |
| hotelbl | 负责实现酒店信息管理和更新所需要的服务 |
| orderbl | 负责实现用户生成、执行、撤销、查看订单和酒店工作人员查看订单所需要的服务 |
| userbl | 负责实现用户注册登陆所需要的服务 |
| roombl | 负责实现客房管理界面所需要的服务 |
| memberbl | 负责实现会员注册和会员优惠所需要的服务 |

5.3.2业务逻辑层模块的接口规范

**表5-3-2-1 promotionbl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
|  | 语法 | |  |
| 前置条件 | |  |
| 后置条件 | |  |
|  | 语法 | |  |
| 前置条件 | |  |
| 后置条件 | |  |
| **需要的服务（需接口）** | | | |
| 服务名 | | 服务 | |

**5.4数据层的分解**

数据层主要给业务逻辑层提供数据访问服务，包括对于持久化数据的增、删、改、查。各业务逻辑需要的服务由各DataService接口提供。持久化数据的保存采用MySQL数据库进行保存。

以CommodityDataService为例，数据层模块的描述具体如图5-4-1所示。

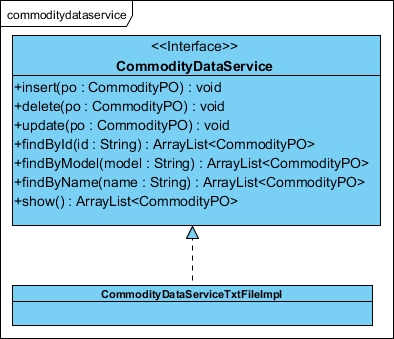


图5-4-1 CommodityDataService数据层模块

5.4.1数据层模块的职责

数据层模块的职责如表5-4-1-1 ~ 5-4-1-13所示。

**表5-4-1-1 数据层模块(DataService)的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
|  | 持久化数据库的接口，提供集体载入、集体保存、增、删、改、查服务 |
|  | 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表5-4-2-1 ~ 5-4-2-13所示。

**表5-4-2-1 数据层模块(DataService)的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
|  | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |

1. **信息视角**

**6.1数据持久化对象**

系统的PO类是对应的相关的实体类。系统主要的PO类如下所示：

**OrderPO: 订单PO类**

包含：订单ID，用户ID，用户名称，用户联系方式，会员类型，订单生成时间，订单完成时间，订单状态，所在酒店，所在房间，订单预订时间段，用户入住时间（订单执行时间），用户离开时间，订单价格，信用值变化

持久化信用记录对象如CreditPO的定义如图6-1-1所示。

|  |
| --- |
| /\* @author huangxiao  \* 2016年10月14日  \*/  package po;  public class CreditPO {  /\*\*  \* 客户id  \*/  private String clientID;    /\*\*  \* 信用改变时间  \*/  private String changeTime;    /\*\*  \* 信用改变值  \*/  private int changeValue;    /\*\*  \* 改变后信用值  \*/  private int credit;    /\*\*  \* 构造方法  \* @param clientID  \* @param changeTime  \* @param changeValue  \* @param credit  \*/  public CreditPO(String clientID, String changeTime, int changeValue, int credit) {  this.clientID = clientID;  this.changeTime = changeTime;  this.changeValue = changeValue;  this.credit = credit;  }  public String getClientID() {  return clientID;  }  public void setClientID(String clientID) {  this.clientID = clientID;  }  public String getChangeTime() {  return changeTime;  }  public void setChangeTime(String changeTime) {  this.changeTime = changeTime;  }  public int getChangeValue() {  return changeValue;  }  public void setChangeValue(int changeValue) {  this.changeValue = changeValue;  }  public int getCredit() {  return credit;  }  public void setCredit(int credit) {  this.credit = credit;  }    } |

图6-1-1 持久化订单对象CreditPO的定义

**6.2 MySQL持久化格式**

ID规范：  
订单号：   16位  
 1234567890123456

系统用户：  10位

1234567890  
酒店：     8位  
 12345678

密码：     8-16位，仅英文字母加数字

Yym123456789