**酒店预订系统HRS**

**（Hotel Reserve System）**

**测试报告**

**冰与火小组**

**2016-12-18**

**版本变更记录**

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 版本 | 作者/修改者 | 描述 |
| 2016-12-18 | 1.0 | 王洪凯 | 创建 |
|  |  |  |  |
|  |  |  |  |

**目录**

[一、 引言 4](#_Toc30049)

[1.1 文档信息 4](#_Toc6590)

[1.2 范围 4](#_Toc29552)

[1.3参考资料 4](#_Toc16173)

[二、 详细情况 4](#_Toc2539)

[2.1测试结果概述 4](#_Toc13951)

[2.2详细测试结果 4](#_Toc1608)

[2.2.1 Comment模块 4](#_Toc25230)

[2.2.2 Credit模块 5](#_Toc17996)

[2.2.3 Hotel模块 6](#_Toc24337)

[2.2.4 Member模块 8](#_Toc32688)

[2.2.5 Order模块 9](#_Toc30781)

[2.2.6 Strategy模块 10](#_Toc28106)

[2.2.7 User模块 11](#_Toc25652)

[2.3 决策理由 11](#_Toc23991)

[2.4总结及建议 11](#_Toc29978)

# 引言

## 1.1 文档信息

本测试报告于2016-12-18日撰写，主要总结了测试阶段的测试和分析结果

## 1.2 范围

文档包含了对前几次作业所撰写的测试用例的分析和回顾以及对主要是逻辑层模块的测试设计和结果。

## 1.3参考资料

# 详细情况

## 2.1测试结果概述

测试项目为：酒店预订系统HRS 的服务端（Server）中的逻辑层与数据层的模块，测试设计是同模块的逻辑层与数据层同时一起测。

本次测试过程分成好几个阶段：

初次测试所有测试用例仅通过不到1/2，其中以failure居多，即程序运行时异常

经过修改后的第二次测试仍存在部分运行时异常错误，但通过用例增至一半

最后一次修改后，几乎所有测试用例都能通过。

经过多次修改与再测试，测试结果正确率不断提高。

其中存在部分wa错误（Assert不通过），原因为数据库中无相应数据，只能先造数据并只能测一次

## 2.2详细测试结果

### 2.2.1 Comment模块

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 基本要求 | 测试情况 | 测试通过 | |
| 是 | 否 | |
| 获取酒店的评论 | 若酒店存在评论则返回相应评论的集合，若不存在则返回NULL | 功能实现 | 是 |  | |
| 评论订单 | 根据传入的ordervo评论，若vo信息错误（查不到相应用户或订单），则返回FAIL否则成功返回SUCCESS | 功能实现 | 是 |  | |
|  |  |  |  |  | |

|  |  |  |  |
| --- | --- | --- | --- |
| 功能 | 缺陷 | 分析 | 缺陷情况 |
| 评论订单 | 空指针 | 传入的vo为空，未对此进行检查则出现此错误 | 已解决，增加对vo的检查 |
| 评论订单 | 评论内容为空 | 应不予受理内容为空的评论 | 已解决，在界面层确保不会传入空评论 |
|  |  |  |  |
|  |  |  |  |

总体分析：

在设计时未考虑到评论内容为空的情况，与规格出现了差异，原因是未对规格进行更为详细的设计。

### 2.2.2 Credit模块

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 基本要求 | 测试情况 | 测试通过 | |
| 是 | 否 | |
| 获取用户的信用记录 | 若用户userid存在信用记录则返回相应信用记录的集合，若不存在或用户为空则返回NULL | 功能实现 | 是 |  | |
| 信用充值 | 根据传入的userId和信用充值的金额数，对相应的用户进行充值，成功返回SUCCESS，若找不到用户则返回FAIL | 功能实现 | 是 |  | |
|  |  |  |  |  | |

|  |  |  |  |
| --- | --- | --- | --- |
| 功能 | 缺陷 | 分析 | 缺陷情况 |
| 信用充值 | 内容为负 | 传入的金额数未检查 | 已解决，若为负则返回FAIL，并在界面层再检查一次增加一层保险 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

总体分析：

与规格较为吻合

### 2.2.3 Hotel模块

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 基本要求 | 测试情况 | 测试通过 | |
| 是 | 否 | |
| 获取城市 | 客户界面初始化搜索酒店时给出所有城市和对应城市的商圈 | 功能实现 | 是 |  | |
| 添加城市和商圈 | 初始化数据库中的数据，由于城市和商圈一对多的关系，且数据库中无对应表显示，故手动代码添加 | 功能实现 | 是 |  | |
| 获取酒店 | 根据SearchVO中传入的城市商圈和checkintime checkouttime，获取满足条件的所有酒店的集合（基础的酒店信息BasicHotelVO），若不存在则返回NULL | 功能实现 | 是 |  | |
| 获取预定过的酒店 | 根据传入的用户的信息，获取所有该用户预订过的酒店的id的集合，若不存在则返回null | 功能实现 | 是 |  | |
| 更新客房信息 | 根据传入的MaintainRoomInfoVO，修改相应的房间信息并返回SUCCESS，若失败返回FAIL | 功能实现 | 是 |  | |
| 获取酒店详细信息 | 该接口为搜索酒店第二步，根据传入的userid和hotelid返回所有和该用户和酒店有关的comment与order，若不存在则返回NULL | 功能实现 | 是 |  | |
| 线下入住执行 | 根据传入的VO修改相信的房间信息，成功返回SUCCESS，失败返回FAIL | 功能实现 | 是 |  | |
| 线上提前退房 | 根据传入的VO修改相信的房间信息，成功返回SUCCESS，失败返回FAIL | 功能实现 | 是 |  | |
| 获取房间信息 | 酒店工作人员在执行客房管理前所需的信息，返回酒店当天的房间信息 | 功能实现 | 是 |  | |
| 维护酒店信息 | 根据传入的VO修改相应的酒店信息，成功返回SUCCESS失败返回FAIL | 功能实现 | 是 |  | |
| 添加酒店 | 根据传入的VO向数据库中添加相应的酒店，成功返回对应酒店的信息 | 功能实现 | 是 |  | |
|  |  |  |  |  | |

|  |  |  |  |
| --- | --- | --- | --- |
| 功能 | 缺陷 | 分析 | 缺陷情况 |
| 获取城市 | 一个城市对应的商圈仅为一个 | 执行的SQL语句写错或者本来就是一个 | 已解决，执行的SQL语句有错 |
| 添加城市和商圈 | 重复主键 | 一对多设置inverse=false，应让一的一方管理，并先插入一方，且不应设置主键 | 已解决，更改相应的关系文件与插入语句的顺序 |
| 添加城市和商圈 | 效率低下 | 单向一对多，是先插入一，多再更新多，效率低下 | 未解决，功能完整，且考虑到城市和商圈仅初始化一次 |
| 获取酒店 | SQL语句错误 | 根据错误信息可知是对绑定的属性设置了空值，由于SearchVO可能未传入checkintime | 已解决，在SQL语句拼接加入对日期null的判断 |
| 获取酒店 | 无法获取到相应数据 | 检查过逻辑层后怀疑数据库中的数据有问题 | 已解决，数据库中数据有问题 |
| 设置房间信息 | dozer无法匹配 | 传入参数未对相应属性进行null的检查 | 已解决，增加对null的检查 |
| 获取酒店详细信息 | EOF EXCEPTION | 由于传入的Image序列化会出现很大的错误，故改写了writeObject，但是这时若Image为null则报此错 | 已解决，在write中增加对image类型属性的检查 |
| 获取房间信息 | 获取房间太多 | 执行的SQL语句有问题 | 已解决，修改后正确 |
| 维护酒店信息 | EOF EXCEPTION | 同获取酒店详细信息 | 已解决 |
| 增加酒店 | 数据库中关联数据未插入 | 逻辑层忘记增加 | 已解决，增加相应逻辑 |

总体分析：

与规格有部分不同，主要是规格中对需求的理解出现了偏差，在实际编码过程中意识到如此实现不可，于是更改了相应的逻辑。

### 2.2.4 Member模块

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 基本要求 | 测试情况 | 测试通过 | |
| 是 | 否 | |
| 注册VIP | 根据传入的userid与相应注册VIP的信息来注册VIP，若成功返回SUCCESS,已经是会员返回FAIL\_ALREADYVIP，无此用户返回FAIL\_WRONGID,信用值未达要求返回FAIL\_CREDITNOTENOUGH | 功能实现 | 是 |  | |
| 取消会员 | 根据传入的userId取消相应的会员 | 功能实现 | 是 |  | |
| 用户管理-客户 | 根据传入的id/username返回相应用户管理的信息，若成功则在相应的VO中返回信息为成功，失败（无此用户），则该信息为WRONG\_ID | 功能实现 | 是 |  | |
| 用户管理-酒店工作人员 | 根据传入的MaintainHotelWorkerVO返回相应更改的成功与否 | 功能实现 | 是 |  | |
| 用户管理-网站营销人员 | 根据传入的MaintainWEBSalerVO返回相应更改的成功与否 | 功能实现 | 是 |  | |
| 用户管理-删除用户 | 根据传入的id删除相应的用户 | 功能实现 | 是 |  | |
|  |  |  |  |  | |

|  |  |  |  |
| --- | --- | --- | --- |
| 功能 | 缺陷 | 分析 | 缺陷情况 |
| 删除用户 | 未对用户做判断即删除 | 好像没什么报错，但是传回界面层所显示的信息有误 | 增加返回信息 |
|  |  |  |  |

总体分析：

规格中对VO的理解不到位，新的理解，一个表单即为一个VO，故约定了许多表单

### 2.2.5 Order模块

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 基本要求 | 测试情况 | 测试通过 | |
| 是 | 否 | |
| 创建订单 | 根据newOrderVO创建新订单，成功放回SUCCESS，失败则返回FAIL | 功能实现 | 是 |  | |
| 订单异常 | 根据id对订单执行异常，成功返回SUCCESS，失败返回FAIL | 功能实现 | 是 |  | |
| 订单执行 | 根据id对订单执行，成功返回SUCCESS，失败返回FAIL | 功能实现 | 是 |  | |
| 订单补执行 | 根据id对订单补执行，成功返回SUCCESS，失败返回FAIL | 功能实现 | 是 |  | |
| 订单撤销-用户 | 根据id对订单执行撤销，成功返回SUCCESS，失败返回FAIL | 功能实现 | 是 |  | |
| 订单撤销-网站营销人员 | 根据id对订单执行撤销，成功返回SUCCESS，失败返回FAIL | 功能实现 | 是 |  | |
| 获取用户订单 | 用户查看订单初始化，若有返回订单的集合，若无返回NULL | 功能实现 | 是 |  | |
| 获取酒店订单 | 酒店工作人员查看订单初始化，若有返回订单的集合，若无返回NULL | 功能实现 | 是 |  | |
| 获取网站营销人员订单 | 网站营销人员查看订单初始化，若有返回所有异常订单的集合，无返回NULL | 功能实现 | 是 |  | |
|  |  |  |  |  | |

|  |  |  |  |
| --- | --- | --- | --- |
| 功能 | 缺陷 | 分析 | 缺陷情况 |
| 订单异常 | 一是服务端可能经过更新，导致在此时间的订单无法异常，二是要多开线程对服务端效率有影响 | 将异常工作移交至数据库 | 已解决，在数据库中增加计划任务 |
| 订单操作 | 级联数据未更新 | 相应的方法忘记写相应的逻辑 | 已解决，增加相应的逻辑 |
| 五项订单状态的操作 | 冗余代码，相似代码 | 都是更改订单状态的操作，用表驱动的方式精炼了代码 | 已解决，用表驱动的方式精炼的代码，抽离了相应方法的步骤 |
| 获取订单 | 动态代理加载出错 | 从数据库中获取数据后，对关联数据的属性的懒加载设置为真 | 已解决，获取数据后，用Hibernate.initialize方法相应初始化 |
| 订单操作 | 逻辑错误 | 非逻辑错误，实际上是缓存未更新 | 已解决，在任何涉及到数据更改的地方，设置缓存的更新 |
|  |  |  |  |

总结分析：

主要是效率和代码上的一些优化问题，而在订单操作的地方，实际上借鉴hibernate一级缓存的思想，采用脏检查的方式会更好，但是实际上这样会多进行很多数据库操作，实际上效率也不高，总体来说最后一项缺陷还是悬而未决。

### 2.2.6 Strategy模块

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 基本要求 | 测试情况 | 测试通过 | |
| 是 | 否 | |
| 创建策略 | 根据传入的vo创建相应类型的订单，成功返回SUCCESS，失败返回FAIL | 功能实现 | 是 |  | |
| 删除策略 | 根据传入的strategyid删除相应的策略信息 | 功能实现 | 是 |  | |
| 获取酒店策略 | 根据传入的hotelid获取该hotel的所有strategy的集合，若无返回NULL | 功能实现 | 是 |  | |
| 获取订单满足的策略 | 根据初始订单传入的相应数据，返回满足该当前订单的所有策略 | 功能实现 | 是 |  | |
| 获取策略类型 | 初始化创建策略，返回当前可创建的所有策略 | 功能实现 | 是 |  | |
| 获取网站策略 | 获取所有网站策略 | 功能实现 | 是 |  | |
|  |  |  |  |  | |

|  |  |  |  |
| --- | --- | --- | --- |
| 功能 | 缺陷 | 分析 | 缺陷情况 |
| 创建策略 | 创建策略不知道策略类型的id | 从文件获取的策略类型无id，客户端创建无法返回 | 已解决，把从文件中获取改为在数据库获取 |
| 创建策略 | class not found | 反射机制加载类路径出错 | 已解决，更改为正确的类 |
| 获取策略 | Dozer mapping failure | 映射文件未写好 | 已解决，写好映射文件 |
|  |  |  |  |

总体分析：

之前规格新增策略想到的是策略模式，实际上应用起来还是策略模式和反射机制动态加载类更能解耦合

### 2.2.7 User模块

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 基本要求 | 测试情况 | 测试通过 | |
| 是 | 否 | |
| 登录 | 输入正确的用户名和密码登录成功，输入错误登录失败，输出长度不足或为空也给予相应的信息 | 功能实现 | 是 |  | |
| 登出 | 登出操作 | 功能实现 | 是 |  | |
| 注册用户 | 注册新用户，若不存在同用户，创建成功返回SUCCESS，否则返回FAIL | 功能实现 | 是 |  | |
|  |  |  |  |  | |

此模块较为简单，暂无缺陷

总体分析：

整体符合规格。

## 2.3 决策理由

对所做的测试，若是整体逻辑无错，只是边角忘了检查等容易改的错，更改后若正确则通过

整体逻辑有错，则不通过

## 2.4总结及建议

部分总结已写入2.2中

对于当前所考虑到的环境，已经做出了相应的解决方法，但是高并发情况下订单缓存可能出现相应问题，且由于传统的入住规定，在每晚凌晨0：0，数据库压力可能会大大加大，这时候客户的操作可能会出现相应问题，还未处理。

对数据库的并发性，未考虑脏读，重复读，坏读等情况，虽然有了解数据库锁以及框架通过事务等级和乐观锁机制的解决方法，但是并未加入，重点还是在考虑整个系统的主逻辑。

在测试解决缺陷的过程中学习到了许多