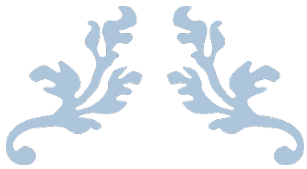


物流管理系统（LMS）
项目评价文档

项目评价文档

2017-1-1

目录

修改记录

修改人	修改时间	修改原因
万年杰	2016/1/1	初始版本

一、项目管理评价

在本次“酒店预订系统”的开发过程当中，项目管理整体较为良好。项目所使用的过程是迭代式开发，每一个阶段都能有部分产品产生，极大地鼓舞了团队士气。但是，实际的过程和当初的设想有所不同，在当初的设想中，从设计阶段开始，每一次的迭代都应该产生可运行的代码；然而在实际开发过程当中，从构造阶段开始才有了可执行的代码。

在过程当中，由于经验不足的原因，需求阶段和体系结构设计做的不是很好，需要在下次的项目当中强化和改进；详细设计阶段设计的较为出色；构造阶段分工并不是很合理，导致每个人的工作量不是很均衡，好在后来将提前完成自己任务的队员帮助尚未完成任务的队员完成任务，才将整个项目的构造阶段顺利完成，但构造阶段总体时间花费较多，改动太大。

二、产品评价

1.产品在项目生命周期中的变化

在项目的需求、体系结构、详细设计阶段，产品主要以文档为主，包括需求阶段的用例文档、需求文档、需求度量文档，设计阶段的体系结构文档和详细设计文档、测试用例文档等，此外包括少量的代码，主要是桩和驱动，单元测试用例等。在项目的构造阶段，代码量发生了巨大的变化，产品基本成型，但还存在不少缺陷，界面也不够美观。在测试阶段，我们修复了找到的缺陷，并美化了界面，同时写了产品测试报告，产品基本达到交付标准，在此阶段我们准备了产品的部署工作。

2.产品返工情况

因为项目经验不足，在这次项目开发过程中，我们出现了不少返工情况。问题主要集中在以下方面：

1、 在项目设计阶段发现对需求的理解不足，需求阶段一些关键需求的具体内容没有能细化，导致设计阶段有些功能无法设计，要回到需求文档落实细化部分需求。
2、 在构造阶段，我们发现设计阶段定义的接口不够稳定，出现过一些修改，由此导致层与层之间的冲突，出现集成失败的情况。需要修改设计阶段定义的接口。

3.开发工具

在项目开发过程中，我们使用 Word 编写文档，用 ProcessOn 画设计图，使用 Github 进行配置管理，用 maven 进行项目构建，使用 Jenkins 进行项目的持续集成，使用 eclipse 进行软件构造，Junit 进行单元测试，使用 Metrixs 进行软件度量。我们使用大量的工具，成功提

高了项目的开发效率和产品质量。

但是在项目开发过程中，我们在 Github 上遇到一些问题，但我们学习使用 Git Shell 解决问题，并学习使用 gitignore 来忽略一些东西，从而使 commit 更有效，让问题保持在可控的范围内。

总体来说，我们使用的工具支持产品的制造、维护和测量。

4.产品规模

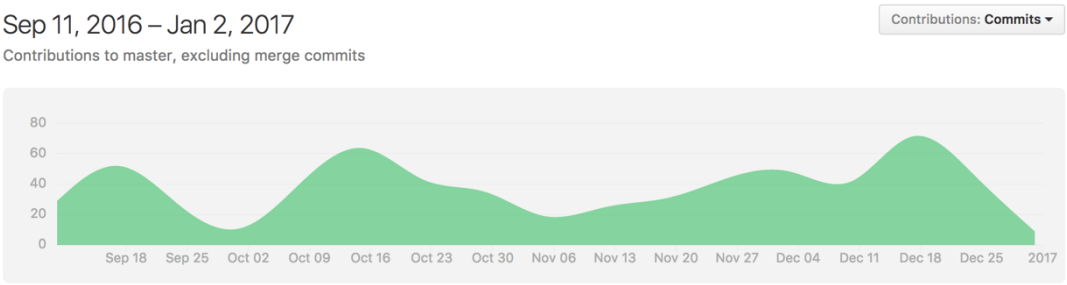
最终产品共有 22907 行代码。

5.产品质量

因为有前期大量的文档支持和采用了良好的设计模式和编码规范，项目的产品质量较之前的项目有很大的提高。但因为开发经验不足和开发能力有限，产品难免存在缺陷。我们在开发过程中已经尽力提高产品的质量，尽我们最大的努力使产品达到交付标准。在测试阶段，我们约有 30 多个记录的产品缺陷，并修复了这些问题。

6.产品度量数据及分析

产品的增长情况



产品在每个里程碑上的测量

里程碑	时间	成果
需求阶段	2016/9/18——2016/10/2	软件需求规格说明书、软件需求度量文档
体系结构设计阶段	2016/10/3——2016/10/17	体系结构设计模型、体系结构设计文档、体系结构原型代码（含桩和驱动）
详细设计阶段	2016/10/18——2016/11/13	详细设计模型、详细设计

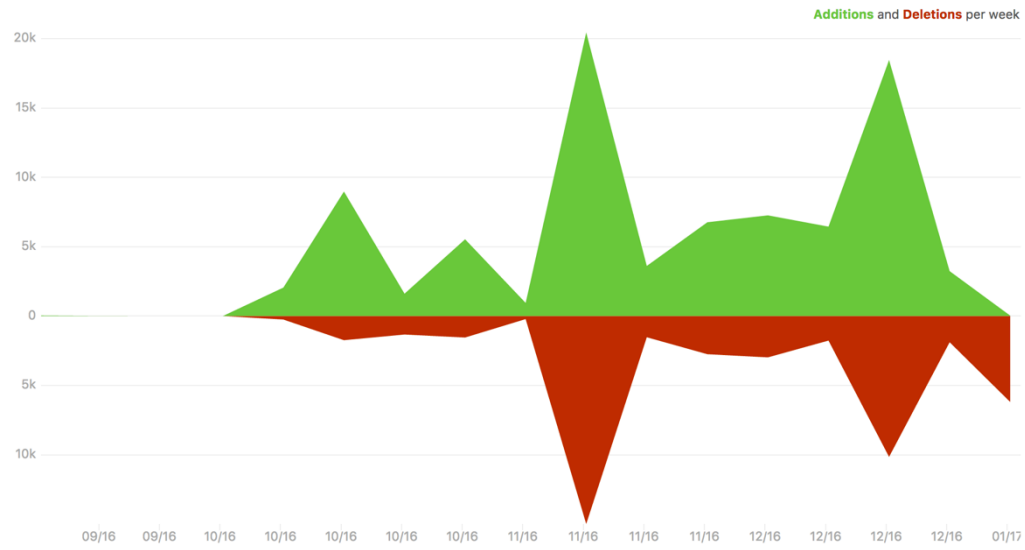
		文档、详细设计的集成测试代码和驱动代码
构造阶段	2016/11/14——2016/12/11	可执行程序（功能实现率 86%）
测试阶段	2016/12/12——2016/12/19	系统测试用例和测试报告文档（功能实现率 95%）

产品复杂度和内容的测量

代码量为 22907 行，共有 373 个类。该产品全部实现了酒店预定系统的基本功能需求，实现了酒店预订操作的一体化。

过程和工具对产品的影响

本项目采取迭代式开发，每次迭代周期为 2 个星期。开发过程的频率如下图。



采用 Jenkins、maven 进行构建和集成，提高了集成的效率。利用 junit 进行测试驱动开发，减少了代码的缺陷。通过 git 进行项目的同步和备份，使小组成员可以随时获得项目的最新进度，也有助于项目开发过程中的回退。

三、团队评价

1. 团队总体评价

团队总体评价良好。管理当中常见的问题有组内人员空闲时间不统一，导致难以确定开

会和集体开发的时间；当出现争执时不能够冷静地通过讨论优缺点来提出解决方案，容易情绪化。主要优点体现在定好了会议时间以后，大家都能较为准时地参加；大家都能以团队项目为重，能够意识到“团队的失败就是每个人的失败”，当出现问题的时候都能够停下手中的工作来修复项目问题；但在布置任务之后常常不能按时完成，这是态度上需要改进的地方。

在团队过程当中，风险集中出现在构造阶段，由于对项目代码量的没有准确的把握和最后期限的限制，导致组内对于能否按时完成构造产生了一种怀疑的态度，好在随着过程的进行，组员们渐渐意识到项目能够按时完成，有了充足的信心。相信这样的问题能随着经验的上升慢慢改进。

在团队管理当中做出的最重要的决定应该是构造阶段的分工决定。由于构造阶段较为重要，分工方式又有很多，但是考虑到我们的体系结构是分层式的设计，拥有并行开发的特点；又考虑到界面层要学习很多并不掌握的 UI 知识，还要承担美工的任务；最后，考虑到我们做这个项目仍然处于一个学习的阶段，我们的决定是让每一个组员尽量多的参与项目的每个过程，所以万年杰、曾虎双、陆茹茹承担了界面层的所有任务，和逻辑层的大部分任务，田原承担了逻辑层的部分任务和数据层的所有任务。

2.个人自我评价

万年杰——在这次项目实践的过程当中，我第一次作为队长带领这个团队，说实话，压力还是很大。这些压力主要表现在每一次分工的过程当中，要充分考虑组员的技术优缺点，作业量的不同，最为合理，最为公平地分配任务，这也就要求你首先对自己的组员有一个比较全面的了解，但这是我们第一次合作，相互的了解并不多，这也就导致之后一些问题的出现。在看法不一致的情况下，我都希望能通过沟通和分析达成一致，但是难免有谁也没有办法说服谁的情况，这时候只能听组长(我)的安排，有时候组员会很抵制，但是之后还是会完成任务。

在这次的项目实践中，我主要承担了 UI 层和订单逻辑的工作，有时也要去看看数据层的代码，因此，我也学习了数据库的相关语句操作，包括建表、增删改查等。在 UI 的构建过程中学会了用 JavaFX 来构建 UI 的实现，以及了解了 FXML 和 Controller 的对应机制，JavaFX 的反射机制，CSS 的映射机制。为了 Github 不发生很多冲突从而正常运行，我学会了如何利用 git shell，如何编写.gitignore 文件。在工具使用方面，我学会了如何用 Junit 进行单元测试，如何用用 Metrixs 进行软件度量。

总之，我们组最终还是实现了全部需求，产生了可执行的文件，组内关系融洽，建立起了很好的友谊，这也是这次大作业除了技术方面我获得的最重要的东西吧。 学到了很多，也感受到了很多，更多的是看到了和别人的差距，日后还需努力，以此共勉。

田原——这次大作业是我们第一次以近似工程化的方法进行团队合作开发，得益于我们良好的交流机制，我们在讨论过程中从未发生过争吵的情况，大家都能欣然接受别人提出来的建议。

项目实践过程中，我主要负责的是服务器端的数据库工作和网络连接的部分，以及自己模块的逻辑部分。我学会了一些基本的 JDBC 语言，也熟练掌握了如何用命令行的方式操作数据库，关于 java 的 RMI 的搭建也有了更深一步的理解，特别是注册表的一些问题，以及 Mac 与 Windows 系统进行互 ping 时的一些注意点。这次项目实践中，由于前期我就建了两个工程，所以 PO 包与 DAO 层的接口包不可避免的分散在 client 与 server 两边，造成了修改时可能只改了一端的情况，两端代码不同步，这样会导致 RMI 通信出问题，这是前期项目搭建时的失误。还有在具体编写代码时，也牢牢记住了防御式编程的思想，使用前都会进行有

效性，是不是为 null 的判断。这次项目中也加深了我对 junit 使用的理解，以及在结合服务器端进行测试时，RMI 连接部分要放在 beforeclass 注释的方法中，切不可启动客户端后直接测试，因为客户端的界面还没有开发，客户端瞬间就 terminate 了。remotehelper 就不存在了。更加加深了我对异常处理的理解，知道何时该抛出怎样的异常，何时该处理异常才能增加程序的鲁棒性。

最后，也吸取了一些教训，比如说某些大家模块里都要出现的变量名，应该提前统一好，免得后期修改造成大量麻烦。详细设计阶段定接口时，应当考虑的更加完备一些，避免到代码构建时再修改接口，这样也会影响并行开发的效率。还有就是小组的沟通还是不够频繁，导致一些理解上的偏差，最终导致某个人的工作要重新做。

曾虎双——在这次大作业的项目实践过程中，我在需求设计等阶段和组员们合作完成了文档的编写，在构造阶段主要负责逻辑层和界面层 roombl 和 strategybl 这两个模块的编写。在这个过程中，这是我的第一次团队项目实践，我在其中收获不少。

首先是学会了需求文档、体系结构文档等各种文档的编写，在构造阶段对分层的体系结构风格、模块化等有了更深一步的了解，在编写逻辑层代码过程中我对各个模块的实现，模块之间的协作有了深入了解，学会用接口解决循环依赖问题，操作上基本实现了 vo 和 po 的相互转换，学会了防御式编程，运用异常处理机制，抽象工厂模式等。在编写界面层代码过程中学会了用 javaFX 来构建 UI，学会了用 scene builder 来构建 FXML 界面，运用 Controller 和 FXML 的对应机制实现界面逻辑，这些知识对于我的学习是一大提高。

其次，我在团队合作中学会了团队交流的方式。懂得倾听接受他人的意见，懂得在意见不一致的时候协调统一，达成一致。在不断的交流过程中我也收获到和组员们的友情。
同时，我也发现了很多自身的不足。比如在编写代码过程中只是为了实现功能，经常忽视内聚耦合、代码重复等问题，编程能力还有待改进。日后我还要多加努力，不断提高自我，争取下一次的团队实践中有更大进步。

陆茹茹——在这次大作业的过程中，我学到了很多东西。首先，我学会了怎么去写用例、体系结构和详细设计等一系列文档。也懂得了这些文档的用处，虽然有时候写文档是件很麻烦的事情，但是我体会到文档的编写对我们后期的工作有着很大的影响。从用例的划分到后面体系结构的设计，都影响了作业的整体设计。而我在体系设计环节，考虑的因素太少，从这导致设计的不够全面，以至于在详细设计阶段还回过头改了体系结构，万幸变动不算太大。这是由于我考虑不周从而导致的麻烦。在文档之后，我负责了逻辑层 user 模块以及界面层 user 模块和部分 hotel 模块代码，在这个过程中，我体会到很多自己的不足，逻辑层代码有的有代码重复现象，而且异常处理做的也不够好，而且没有好好运用 resultMessage，这使得后来界面层人机交互做的不够完善。而界面层的工作，我学习到了有关 javafx 和 html 以及 css 的运用的相关知识，这是这次大作业的一大收获。尽管我们的界面做的很简单，但我的确学习到了不少东西。比较遗憾的是，我比较喜欢界面方面的工作，这次界面做的不够好算是很大的遗憾。希望接下来的时间，我能够学习到更多的关于界面层的知识，能在下一次团队合作中做好界面相关工作。而在这次的团队合作中，我懂得了团队合作的方法，但是有时我还是可能因为意见不合跟某个队友争论起来，我觉得自己在这些方面仍然需要注意。但总体来说，我们的团队合作还算融洽。总结来说，感觉这次的大作业，我的工作还算是相对较少的，但是有的时候我完成的进度反而比其他人慢，而且有的完成的也不算好，这是我需要反思的地方，希望之后自己能够完善自己，继续努力，在下一次的团队合作中发挥好自己的作用。

3.团队会议记录

在前期的合作过程中，我们组主要由万年杰负责会议记录，后来发现这种统一记录的方式效率过于低下，于是后来我们采用了各自记各自的“分布式会议记录法”，但总体的问题仍然由万年杰记录，发现效果确实不错。

F. A. F第一次例会

2016/9/14
7:00 – 9:30
地点暂定

1. 讨论**软件架构模式**（MVC 还是其他），请提前思考。
2. 讨论**用例文档**分工，定DDL，请提前阅读老师给的需求文档，仔细思考需求用例，便于届时讨论。
3. 讨论整个项目分工，软件架构师，UI设计师，如何很好地分配任务。
4. 其他问题讨论。

F. A. F第一次例会Review

2016/9/14

1. **软件架构模式** —— 分层，MVC，根据用例再确定。
2. **用例文档**分工：（按用例图从上到下）
   （DDL：2016/9/21 23:59:59）
   a) 万年杰 1-6
   b) 田原 7-12
   c) 曾虎双 13-17
   d) 陆茹茹 18-21
3. 其他分工，（UI、架构）等，中秋后陆茹茹到场再讨论。
4. 关于用例图的讨论结果
   a) 一个用例对应一个用例描述
   b) 格式参考书上的用例描述
   c) 今晚会在群里发上届的大作业作为参考

F.A.F第三次会议Preview

2016/9/22
7:00 – 9:30
南京大学图书馆

1. 修改并确定用例图。
2. 讨论并审核系统顺序图，概念类图，进一步完善用例说明文档。
3. 统一所有交叉概念类的定义，讨论全局的概念类图。
4. 商定全局的状态图分工。
5. 商定以上事物的DDL。
6. 其他问题讨论。

F. A. F第二次会议

2016/9/20
16:00 –
四食堂

1. UC3
    a) 生成订单时是否必须选定房型后，即：列表中是否可以生成订单
    b) 信用值<= 0不能生成订单
    c) 折扣是否可以叠加
2. UC4
    a) 若需要扣除信用值，则先提示用户将会扣除信用值
3. UC6
    a) 注册会员，是否需要验证
    b) 是否有“注册”这个用例
    c) 同一个人是否可以注册两种会员
4. UC9
    a) 酒店详情还应包括评论
5. UC12：录入可用客房，是什么意思
6. UC13
    a) 扩展流程1-3a，信用值及可预定等
    b) 正常流程1.2，工作人员只需输入房间号，无需在列表中寻在
7. UC14
    a) 订单执行有没有必要作为一个用例存在
    b) 或者是合并到
    c) 增加搜索功能
8. “入住信息”应与“订单信息”分开——线下和线上的交互
9. UC 21
    a) 网站管理人员不可以修改酒店信息
    b) 网站管理人员添加酒店时只需初始化“酒店名称”信息
10. 每个用例的系统顺序图和概念类图 DDL： 9.22nd 18:30

F. A. F第三次会议

2016/9/22
7:00 – 9:30
图书馆

1. 酒店简要信息:
    a) 酒店名称
    b) 所属商圈
    c) 酒店地址
    d) 酒店星级
    e) 酒店评分
2. 酒店详细信息:
    a) 酒店简要信息
    b) 酒店简介
    c) 设施服务
    d) 客房类型
    e) 原始价格
    f) 空房类型及数量
    g) 评论
3. 订单简要信息:
    a) 客户ID
    b) 订单ID
    c) 酒店名称
    d) 酒店地址
    e) 开始时间
    f) 退房时间
    g) 客房类型
    h) 客房数量
    i) 总价
    j) 订单状态
4. 订单详细信息:
    a) 订单简要信息
    b) 订单生成时间
    c) 最晚订单执行时间
    d) 预计入住人数
    e) 有无儿童
    f) 是否享受折扣
    g) 评价状态
5. 已撤销订单 里有 撤销时间
6. UC2
    a) 更新入住信息
    b) 更新可用客房

7. 可用客房列表
    a) 房间号
    b) 房间类型
    c) 数量
    d) 原始价格
8. 评价
    a) 不须注明订单信息
    b) 酒店不须知道评价订单
    c) UC10修改
9. CCD1、2、3、4、5、
10. 异常、未执行、已执行、撤销
11. 客户简要信息：
    a) 名称
    b) 密码
    c) 联系方式
    d) 信用
12. 酒店工作人员
    a) **工号**
    b) **密码**
    c) **酒店名称**
13. **问老师的问题**
    a) **酒店工作人员：浏览酒店客房（要不要增加用例）**
    b) **酒店房间要不要房间号**

F. A. F第五次会议

2016/9/27
4：00-6：00
四食堂

1. 测试用例套件以用例为单位，尽量用最少的场景覆盖最多的功能。覆盖率要求达到90%。
2. 度量数据以用例为单位，计算方法见书。
3. DDL：在9.28th 19:00 前提交所有系统级需求。尽量提交度量数据。不强制要求。
4. 每个人列出需求规格文档4.2后***非功能需求***内容中与自己有关的部分，便于届时整合。
5. 9.28th 在教室开会，主要内容为评审、整合需求规格文档4.2后的部分和度量数据。
6. 9.29th 在图书馆开会，主要内容为评审、整合之前工作，讨论测试用例套件的开发。
7. 系统级需求前的内容由万年杰撰写。

2016年10月26日 下午7:09

1. 初始化订单：(***BL***)检查UserCredit，若>0, 返回一个VO，填好了酒店名称，生成了订单号等初始信息。否则返回null
2. (**UI**)要求客户按照如下顺序填写订单
    - 入住时间 --> 退房时间 --> 房型 --> 房间数量 --> ...
3. (**UI**)除入住退房时间（必须在现在或将来的某一时刻）外，其他各项
    - 都由前面的条件决定，限制用户选择
    - 并在填好前面所有项时将该项置为可填
    - 在填好2中四个值时即可显示订单价格了
4. (**UI**)所需所有信息填写完毕方可点击“生成订单”，(***BL***)再次检查是否所有信息都可以被满足，若可以，在数据库中添加相应订单信息并返回ResultMessage，否则返回ResultMessage

2016年10月23日 下午4:58

1. 类图中的是否需要添加属性和方法名，还是像书上一样不用写。
2. 在详细设计中可能会出现接口规范的修改、原型代码的修改，此时与相关人员沟通，注意修改历史的修改。
3. UI层有无供接口

分工：界面原型图确定到个人
划定DDL：逻辑层基本框架

万年杰（OrderUI）：
1. 登录和注册（1.1.1）
2. 客户主界面（1.2.1）
3. 查看自己的订单（1.2.3）
4. 查看酒店的订单（1.3.4）
5. 生成订单界面（1.2.6）
6. 执行订单界面（1.3.7）
7. 管理异常订单（1.4.2）

陆茹茹（UserUI）：
1. 维护个人基本信息（1.2.2）
2. 信用充值（1.4.1）
3. 网站管理人员主界面
4. 管理用户（1.5.1）
5. 管理酒店（1.5.2）

陆茹茹（HotelUI）：
1. 查看预定过的酒店（1.2.4）
2. 评价酒店界面（1.2.5）
3. 酒店工作人员主界面（1.3.1）
4. 维护酒店基本信息（1.3.2）
5. 更新可用客房界面

曾虎双（StrategyUI）：
1. 管理酒店促销策略界面（1.3.3）
2. 网站营销人员主界面
3. 制定营销策略（1.4.3）

曾虎双（RoomUI）：
1. 浏览空房界面（1.3.5）
2. 办理入住界面（1.3.8）
3. 办理退房界面（1.3.9）