软件系统分析与设计结课报告

报告人：许健

学号：20110031020

专业年级：20级计算机科学与技术

1. 软件工程开发的认识

我将从软件工程的开发前期，中期，后期 这三个方面

进行描述

1. 前期：

个人认为软件开发的前期包含需求分析与软件设计两个方面，而这两个部分是软件开发过程非常重要的部分，因为会直接决定后续流程的方向。

1）需求分析:

在开发软件之前，首先要理解用户的需求，根据需求实现相应的功能。

分析需求首先要收集需求，一般情况下用户提供需求，但在此项目中没有用户的情况下我们可以先把自己想象为用户，如果自己是读者自己是系统的管理员需要什么功能，还可以通过上网搜寻资料或者进入别人的图书管理系统去丰富需求的内容。

根据我们收集到的需求可以勾画出用户使用场景，从此场景中提取所要实现的功能。当然需求并不是一次就能够完善的，可能在后来的开发过程中不断有新的需求会加进来，同时还可能因为需求的变更删减一些需求。

2）软件系统结构设计:

根据上一个步骤的需求，首先将所要做的系统模块化，方便分工，比如我们可以把图书管理系统分为用户与管理员系统，密码登录与认证系统，文件存储与加载系统，先将大的框架描绘出来，然后我们再考虑每个模块用什么思想来实现（比如面向对象模型），每个模块内部数据有那些，需要给它输入以及它能够输出什么数据，模块与模块之间是否有数据的交互，能否让模块之间的耦合度降低，有相似的功能实现是否应该代码复用减少工作量，为了提高工作性能是否应该使用一些诸如hash表b+树之类的算法与数据结构，还有更具体更细节的问题要在开发中期即写代码的过程中要体现，但这些问题尽量要在软件系统结构设计这个过程中想到。

3）用户界面设计:

为了方便用户使用系统，给以更好的用户体验，一个良好的用户界面设计是必不可少的。首先与用户进行交互的页面是怎么样的？是命令行黑框，还是色彩丰富的页面？而页面怎么布局，页面是否美观，一个页面会有哪些内容，页面中的哪些地方是用户可以使用的接口，哪些地方可以体现用户使用接口的提示和结果，页面之间的调用关系是怎么样等很多问题都需要去考虑。

作为一个初学者我们可以使用qt，也可以使用html+css+js的网页方式，我们可以从简单做起，逐步丰富完善，如果我们自己的审美设计并不好，还可以借鉴优秀的前端模板，但借鉴完之后要将它内化，使其成为自己在将来可以信手拈来的页面。

4）软件开发的标准与规范

由于软件开发需要多人分工协作完成，就需要一些标准与规范。首先每个人都应该了解自己所要实现的东西，自己会调用什么接口，自己向外提供什么接口，如果一些数据结构与算法并不熟悉，还要进行学习加以熟悉，避免理解错误导致出错。

个人在写代码之前还要到公共代码库中看一下代码，因为可能会有一些公共变量的声明，或者你会调用到别人所写的接口，同时因为自己的代码自己所写的接口会被别人用到，所以代码尽量要简单，变量与函数名要清晰，注释要写好。

II)开发中期:

个人感觉这一部分主要是代码的书写

1. 里程碑

个人所分配的任务不是一蹴而就，很短时间内就能完成的，所以我们要规划好项目进度，根据团队的进度，个人的时间去规划好一段时间内应该完成什么任务，设立一个里程碑

2）结对编程

当个人遇到暂时无法解决的困难时结对编程就是比较有效的，通过两人或者多人的讨论能够更快的解决这个问题。当个人在比较疲倦的情况下，又写了大量的代码，在这种情况下代码大概率存在错误，有的可能是语法编译错误，有的可能是逻辑错误，但当事人往往无法察觉，此时换一个头脑比较清醒的人来检查代码就能及早地纠正错误，节省时间。

3）单元测试

在完成代码的部分模块后，要进行单元测试。首先保证语法过关，没有编译错误，之后复看自己的代码逻辑，看是否实现逻辑正确，自己是否考虑周全，没有遗漏情况，再之后自己使用用例，调用别人写好的接口或者自己实现一个简单的接口进行测试。

III)开发后期：

项目即将结束的阶段

1)对接与测试

无论是后端模块之间的对接，还是前后端之间的对接，可能会出现意向不到的错误，小到头文件的复用，公共变量的多重定义，大到某个数据结构的实现有错误，都有可能出现，这些都需要花费时间，所以在完成自己的模块之后也要积极参与整体的对接与测试。

2）项目完善

由于初始设计时考虑不周或者是随着测试用例的强度增加，项目或多或少都会出现不足，比如某个地方对用户的提示不够充分，没有对输入的合法性进行判断，在没有进行搜索书籍时应该出现一些书籍却没有出现，这些都需要后期的完善。

1. 面向对象模型的特征以及在图书管理系统中的应用
2. 抽象:

抽象通过一个对象的区别于其他对象的特征将其提取出来，在图书管理系统中，用户包括姓名，学号，院系，借还记录等特征，而书籍包括书名，作者，ISBN号等特征，用户的特征与书籍的特征不同，用户就可以被抽象成一个类。而这样抽象的结果我认为是让代码逻辑性更强。

1. 封装:

抽象出的对象被封装为一个类，并且会有很多针对于这个类的操作。比如图书管理系统中用户类会有借书，还书等操作，将这些操作和用户类封装在一起，一个是逻辑上合情合理，第二个是用户类只向外提供这些操作的接口，而不提供实现，更加安全

3)模块化

我认为模块化还是在类的封装中体现，通过类的封装实现模块化是得被抽象对象的内部属性和对外操作结构上更加清晰，同时也方便开发

4）层次

关于分层可以由上到下分，在系统中前端和后端会有数据交流，那么在后端中就会有一个专门负责接收前端数据并向前端发送数据的层面，可以称之为控制层（即Controller层），它会调用业务层的提供的接口；而业务层（Service）是什么呢？比如用户类的还书操作就可以看作是业务层里的一个内容，而业务层又会调用持久层的接口；持久层是直接对数据库进行操作，在此次项目里数据库可以简单的看作是B+树+文件存储加载系统，那么用户类的还书操作就会调用持久层里的还书记录的删除操作，持久层的操作一般都带有“增删改查”。

5)对象标识:

每个对象在整个用户周期里都会有一个唯一的且不与别的对象共享的对象标识。在图书管理系统中，我们把每本图书，每个用户都看作是唯一的，这样做的好处是使得操作可以更加细化。

6）分类:

我认为分类是将不同的对象区分开来，以便分别进行抽象封装

7）持久性:

在我们的计算机中，如果我们突然给计算机断电，那么内存中的数据就会消失，但硬盘中的数据仍然会保存，所以要在我们的图书管理系统中实现这样，那么只需要我们的每一步操作后都马上保存到相应的dat文件中即可。

8）并发性

举一个比较小的例子，我们使用for循环打印100次’hello world’，正常情况下应该是一次又一次进行打印，但我们可不可以给此代码分配4个线程，每个线程打印25次，使得运行时间变为原来的1/4呢?当然是可以的，但我们要注意会有多个子任务要借助同一个公共变量操作，并且每次子任务操作都会影响到公共变量的情况。无论如何，并发性都将极大的提高运行速度。比如在图书的批量删除中，若使用多线程将会大大提高运行速度。

1. 自我总结与体会

I）在本次项目中我调用hash表和MD5加密算法来完成

密码登录认证系统的编写，了解了MD5加密算法，进一步掌握hash表的结构。

II)在本次项目中我做的不好的地方是：过于犹豫，没有主动要求分配任务，因为我的部分因为比较简单很早就已完成，其实我应该更主动地为其他组员分担一些，事后看来这样做会有两个好处，一个是加快小组的进度，一个是将会更加熟悉整个项目的代码，这样在最后项目测试修改时就可以提供更多的帮助。

III)在本次项目中我的进步是:

1. 熟悉了一个完整软件开发的过程。
2. 对项目管理工具如gitee更加了解
3. 一个比较高效的方法是线下一起编程开发，这样有什么问题解决就会比较及时，而且在一起讨论个人进步的速度会更快。