系统分析个人总结

左云浩 17100021048 17计科3班

1. 个人工作

1.1作为组长

作为组长，我首先了解我的组员们的个人能力，从而协调分工问题。通过我们第一次见面讨论，集合大家的意见后，我对于整个项目功能架构有个初步的想法。然后组织写出了需求文档，项目开发文档，这样方便大家对于项目的理解不会出现过多的分歧。为了让大家之间的工作更好的衔接，我提出了三层架构的方式去开发：

①先分出三个人去解决数据访问的问题，这三个人负责的数据尽量关联性不大，然后一个人根据需求文档去制作出qt图形静态界面。

②数据问题解决了，然后就利用数据访问的方法去实现所需的接口函数，接口函数开发过程是也是整个项目最复杂的部分，虽然几乎没有一个接口函数是我写的，但是每一个接口函数都是我提出的要求，我要求了传入参数是什么，传出参数是什么，然后让负责数据的组员去实现。他们测试之后发在github上面，我再进行大量数据的详细测试并了解代码的具体实现（方便后期debug工作），出现问题就让负责人重新修改，满足要求为止。

1.2作为组员

等到大部分接口函数都已经实现了，我就利用这些接口函数在qt开发平台去开发我们小组的具体项目。我首先根据用户使用逻辑去修改组员开发的qt静态界面，再根据具体功能利用接口函数去实现。在qt功能实现方面，我负责了

用户界面——我的图书馆（个人信息，修改个人信息，借阅历史，当前借阅）

用户界面——书目检索（书名，书号，ISBN）

管理员界面——图书管理（图书借阅日志，图书增删，图书借阅归还）

管理员界面——用户管理（添加用户，删除用户，修改用户）

这期间还是出现不少问题，进行debug过程中，同时也修改了很多接口函数代码（部分我让负责人修改，部分我自己修改）。在第一次测试前两天，我跟张根同学，在学院机房待了8个小时，对未能解决的bug和项目整体测试进行完善。第一次测试后，我对整个项目的输入输出进行规范化限制。

报告部分，我负责写图书管理系统系统报告，图书管理系统技术报告。

1. OO思想

根据本次管理系统开发经验和老师上课内容，我简单谈一下我对于OO思想的理解。OO（Object Oriented）思想包括OOA、OOD、OOP。

2.1 OOA（object oriented Analysis）

在系统项目开发的准备阶段，我们要进行系统相关问题分析，我们就要按照面向对象的思想进行分析，对项目素材进行分类整理分析。面向对象分析的时候有一些方法和原则。

①抽象：现实中问题和事物以及事物之间的关系是非常复杂的，我们没有必要去了解它的一切，我们只需要从事物中抽象出来一些我们所需的特征，忽略不必要的细节，从而对有相同特征事物进行分类，比如公交车，汽车，货车都是车。这时候我们也可以限定一些数据属性，进行数据抽象，对他们进行分类，比如这些车长度，重量不一样。从而利用属性和操作将一个事物抽象成一个系统单位，也就是对象。

②封装：封装是抽象出来的对象进行包装成一个类，我们可以根据所需来设置访问者的权限，利用public，protected，private来修饰对象的属性和方法，也可以是对象本身。

③继承：继承是类与类之间的关系的一种，他们之间有相同的部分，也有不同的部分。举个例子来说，就是如果一个类A继承自另一个类B，就把这个A称为"B的子类"，而把B称为"A的父类"。继承可以使子类获得父类所有属性和方法，同时也可以拥有属于自己的属性和方法，主要是为了节省代码量和更好的理解类与类之间的关系，一个类可以被多个子类继承。比如一个类是车，那么它的子类可以是轿车，公交车，货车，电动汽车。继承可以使整个系统模型更加清晰。

④多态：就是A类继承了B类，那么A类就具有了本身和B类的属性和方法，这个状态就是多态性，但是对于一个属性或者方法，不同的对象可能不一样，比如猫科动物是一个基类，其派生类猫的叫声是喵喵，狮子的叫声是吼，豹子的叫声也跟其他派生类不同。

利用以上的思想方法我们可以对一个事物进行抽象分析出成我们所需的素材，这个方法就是面向对象分析。

2.2 OOD（Object Oriented Design）

面向对象设计是面向对象分析接下来的工作，是一种软件设计方法。分析后，并不是所有的类和属性都要用到，设计就是根据功能需求进行筛选，来决定所需的类，累的属性和方法，也要明确继承具体继承那些共同点，也就是依赖关系，这个依赖还包括接口函数之间的关系，接口函数把数据端与客户端进行连接。通过以上细化，分类，明确接口函数传递消息，数据之后，我们就要画出具体的类图和时序图，使开发人员明白项目的整体架构，方便开发人员进行开发（代码编写）。

2.3 OOP（Object Oriented Programming）

这里是项目具体编写和实现的部分。面向对象编程是一个系统项目是由一个独立的子程序所组合而成的，这些子程序也完全可以运用在其他软件工程当中。从而面向对象编程使代码就具有了重用性，灵活性和扩展性。比如我们现在开发了一个图书管理系统，也完全可以很快修改成服装仓库管理系统，也就使得软件开发公司开发出来一个系统，完全可以经过修改后交付给不同的甲方，节约了开发成本。这样每一个子程序（对象）都可以接收休息，处理数据和发送消息。面向对象编程就是模块化，结构化编程。其具有以下特性：

①组件：数据和方法一起在运行着的计算机程序中形成的单元，组件是模块化和结构化的基础。

②封装性：确保组件不会以不可预期的方式改变其它组件的内部状态；只有在那些提供了内部状态改变方法的组件中，才可以访问其内部状态。每类组件都提供了一个与其它组件联系的接口，并规定了其它组件进行调用的方法。具体实现利用了public，protected，private来修饰组件的数据和方法。

③继承性：允许在现存的组件基础上创建子类组件，这统一并增强了多态性和封装性。典型地来说就是用类来对组件进行分组，而且还可以定义新类为现存的类的扩展，这样就可以将类组织成树形或网状结构，这体现了动作的通用性。

1. 软件工程与传统工程的区别

传统工程主要是通过结构化设计，项目被划分成许多树状结构模块，主要的模块就是这颗大树的根，其他模块比如工具模块，低级功能模块就是叶子和小树枝。模块之间具有上下级的关系，互相调用。处于顶端的模块就越接近问题领域，抽象层次也就越高。位置越低就越解决细节部分，接近解决方案领域。上下层模块之间具有很强的依赖关系，与问题领域相关的抽象要依赖于与问题领域无关的细节。当实现细节发生变化时，抽象也会受到影响。那么我们如果想要复用一个抽象的话，就必须把它依赖的细节也一起复制过去。

而软件工程中的OO思想，就与之相反。不再是抽象依赖细节，细节高度依赖抽象。由此，模块的复用性就很强，我们想要用一个类就直接复制过去有啥不一样的地方可以生成派生类。

1. 课程体会

主要是我在项目开发中作为组长的体会。我觉得作为一个开发小组的组长最重要的不是代码能力，但是也一定要对技术问题有足够的了解，不然组员给你代码，你也不容易发现问题，提出解决方法。最重要的是对于整个项目开发有个整体的思路，了解项目开发的流程，由于我刚开始缺乏开发经验，组织写完需求文档，就直接让大家写功能模块，不仅仅是数据类型难以统一，各个类之间的关系也难以理清楚。国庆期间，我及时意识到自己的错误，赶紧组织大家先完善类图和关系流程图，解决好数据问题，再考虑其他的部分。

其次重要的是能够合理的分工，学校小组不像公司开发团队，一入职就差不多已经决定你要做什么工作，我也不了解他们具体能做什么。由此，我就画了一下整个工作的流程图，让大家说一说自己能够担任什么部分的工作，然后我再根据具体情况分工。我觉得对于一个难度不是很高的项目来说，开发团队不是人越多越好，人数一定要适当，人员过多的话，为了让每一个人都能够有工作做，必须把任务分散开，但是分散开，又会出现大家最后工作之间的衔接出现困难。或者出现工作分配不均的情况，毕竟我也不想让某个人因为我没有分配足够的工作，而导致他最终成绩不理想，这对他来说是不公平的。可能是我的分配工作的能力不行，或者小组成员过多，我觉得我的工作分配不能很好地照顾到而每一个人。我觉得这样的一个项目5个人最好。由此我非常抱歉，我没有做到工作量分配均匀，在这里表达歉意。

但是在项目结束的是我觉得我们项目完成地非常出色（我在小组展示的时候由于时间有限也没有很好展示我们小组的成果，也由于对于个人能力不够自信也就没有做小组报告），在实际开发中自然会出现各种各样的问题考虑的不够周到，由此作为组长我必须成为那个考虑周到的人，数据访问实现后，我根据功能列举了许多所需的接口函数，然后让组员去实现，当然后续也根据需求也添加和删除了其他接口函数。然后我只花了2天就把接口函数装配到qt上面。项目做完了，其他人都很惊讶，我们什么时候把项目写完了。我觉得这已经说明我们小组还算挺成功的，在开发过程中我们很好地运用了OO思想。

最后，我非常喜欢这门课，因为这门课我感受不到水分，觉得我们小组，其他小组都很努力在学习，在不断突破自己，在进步。老师也给我们讲了很多系统开发分析思想，不仅仅是对于这门课，对于我们学习其他课程也很有帮助。在这里非常感谢我的老师，感谢我们小组成员，感谢第一小组测试组。