高校选课系统

体系结构设计文档

**V2.0 正式版**

南京大学软件学院

焖烧小鲍鱼工作组

2013-10-17

**目录**

更新历史………………………………………………………………………………3

1、引言……………………………………………………………………………3

1.1编制目的……………………………………………………………………3

1.2词汇表………………………………………………………………………3

1.3参考目录……………………………………………………………………3

2、产品概述…………………………………………………………………………3

3、逻辑视角…………………………………………………………………………3

4、组合视角…………………………………………………………………………5

4.1开发包图……………………………………………………………………5

4.2运行时进程…………………………………………………………………7

4.3物理部署……………………………………………………………………7

5、接口视角…………………………………………………………………………8

5.1模块的职责…………………………………………………………………8

5.2用户界面层的分解…………………………………………………………9

5.2.1用户界面层模块的职责……………………………………………10 5.2.2用户界面层模块的接口规范………………………………………11 5.2.3用户界面层模块设计原理…………………………………………11

5.3业务逻辑层的分解…………………………………………………………11

5.3.1业务逻辑层模块的职责……………………………………………11

5.3.2业务逻辑层模块的接口规范………………………………………12

5.4数据层的分解………………………………………………………………14

5.4.1数据层模块的职责…………………………………………………15

5.4.2数据层模块的接口规范……………………………………………16

5.5客户端启动模块的分解……………………………………………………18

5.5.1客户端启动模块的职责……………………………………………18

5.5.2客户端启动模块的接口规范………………………………………18

5.5.3客户端启动模块设计原理…………………………………………19

5.6服务器端启动模块的分解………………………………………………19

5.6.1服务器端启动模块的职责………………………………………19

5.6.2服务器端启动模块的接口规范……………………………………20

5.6.3服务器端启动模块设计原理………………………………………20

5.7客户端网络模块的分解……………………………………………………20

5.8服务器端网络模块的分解…………………………………………………21

6、信息视角…………………………………………………………………………22

6.1数据持久化对象…………………………………………………………22

6.2数据库表……………………………………………………………………22

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **日期** | **变更原因** | **版本号** |
| 张梦雨刘劭王吉宋程宇 | 2013/10/15 | 最初草稿 | V1.0草稿版 |
| 张梦雨刘劭王吉宋程宇 | 2013/10/17 | 评审后的正式版 | V2.0正式版 |

**1、引言**

**1.1 编制目的**

本报告详细完成对高校选课系统的概要设计，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 |
| bl | 业务逻辑（business logic） | 无 |

**1.3 参考资料**

[IEEE1471-2000];

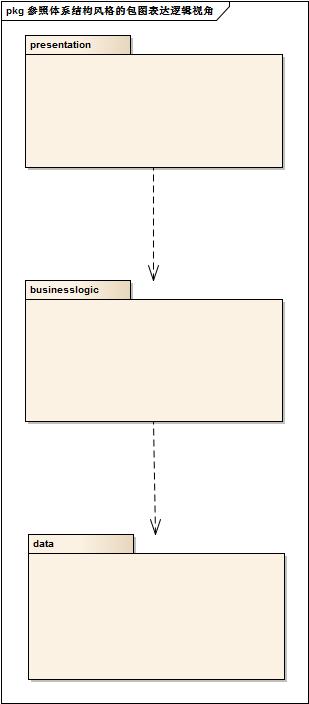
高校选课系统用例文档和高校选课系统软件需求规格说明。

**2、产品概述**

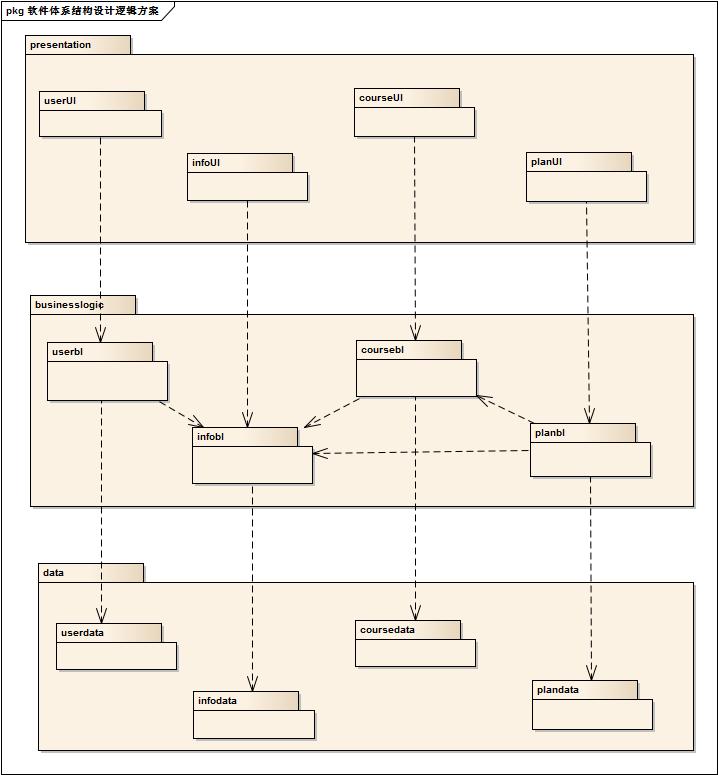
参考高校选课系统用例文档和高校选课系统软件需求规格说明中对产品的概括描述。

**3、逻辑视角**

在高校选课系统中，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）能够很好地展示整个高层抽象。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1和图2所示。



**图1 参照体系结构风格的包图表达逻辑视角**

****

**图2 软件体系结构逻辑设计方案**

**4、组合视角**

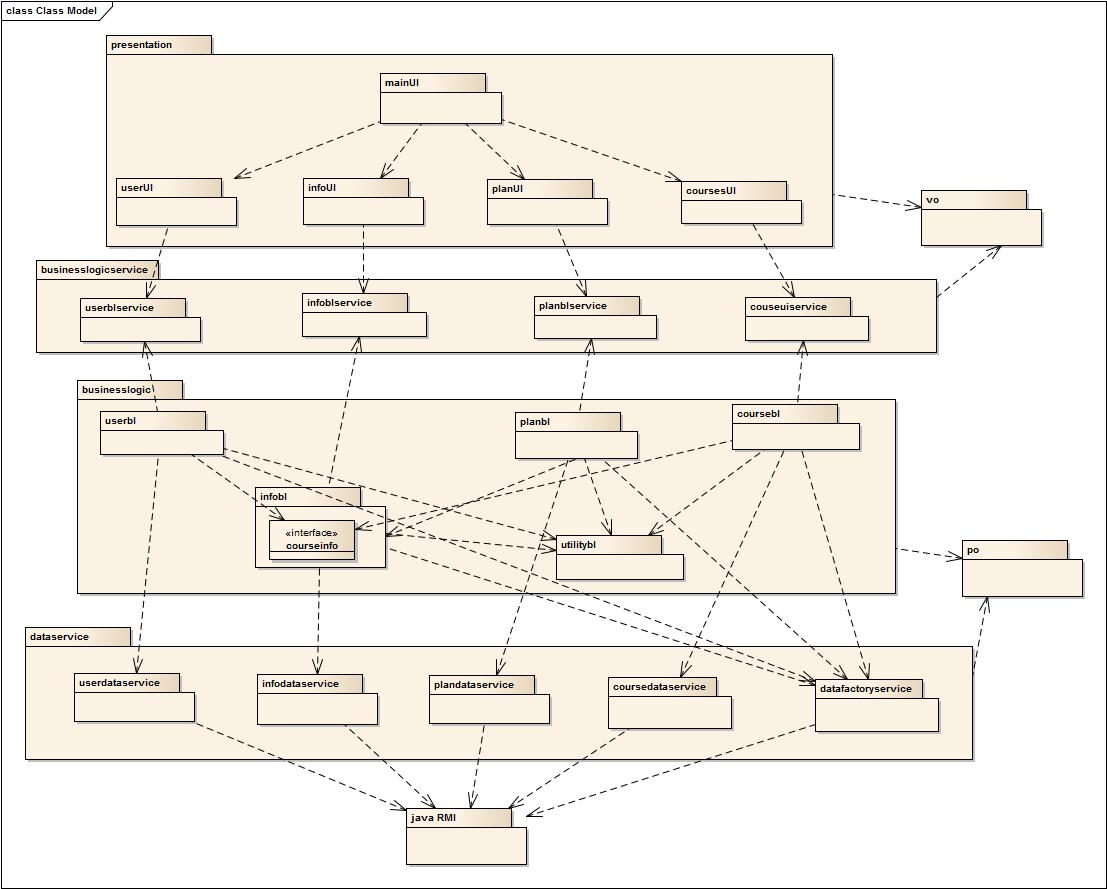
**4.1开发包图**

高校选课系统的最终开发包设计如表1所示。

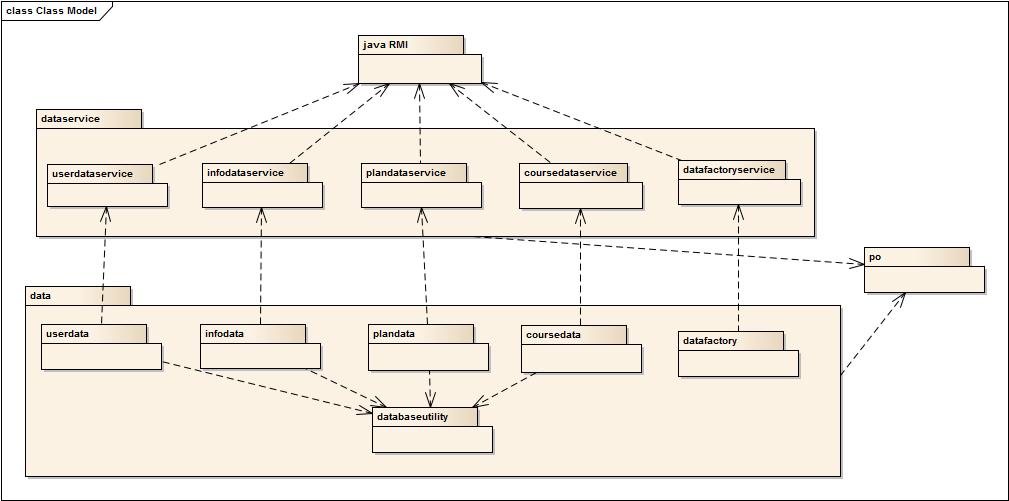
**表1 高校选课系统的最终开发包设计**

|  |  |
| --- | --- |
| **开发（物理）包** | **依赖的其他开发包** |
| mainui | userui, infoui, planui, courseui, vo |
| infoui | infoblservice, 界面类库包, vo |
| infoblservice |  |
| infobl | infoblservice, infodataservice, po, promotionbl, userbl |
| infodataservice | Java RMI, po |
| infodata | infodataseutility, po, infodataservice |
| planui | planblservice, 界面类库包 |
| planblservice |  |
| planbl | planblservice, plandataservice, po, infobl |
| plandataservice | Java RMI, po |
| plandata | Java RMI, po, databaseutility |
| courseui | courseblservice, 界面类库包 |
| courseblservice |  |
| coursebl | courseblservice, coursedataservice, vo |
| coursedataservice | Java RMI, po |
| coursedata | Java RMI, po, databaseutility |
| userui | userblservice, 界面类库包 |
| userblservice |  |
| userbl | UserInterface, UserDataClient, UserPO |
| userdataservice | Java RMI, po |
| userdata | RMI, po, databaseutility |
| vo |  |
| po |  |
| utilitybl |  |
| 界面类库包 |  |
| Java RMI |  |
| databaseutility | JDBC |

高校选课系统客户端开发包图如图3所示，服务器端开发包图如图4所示。

****

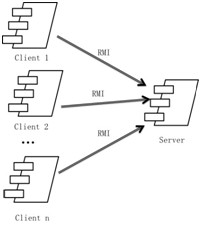
**图3 高校选课系统客户端开发包图**



**图4 高校选课系统服务器端开发包图**

**4.2运行时进程**

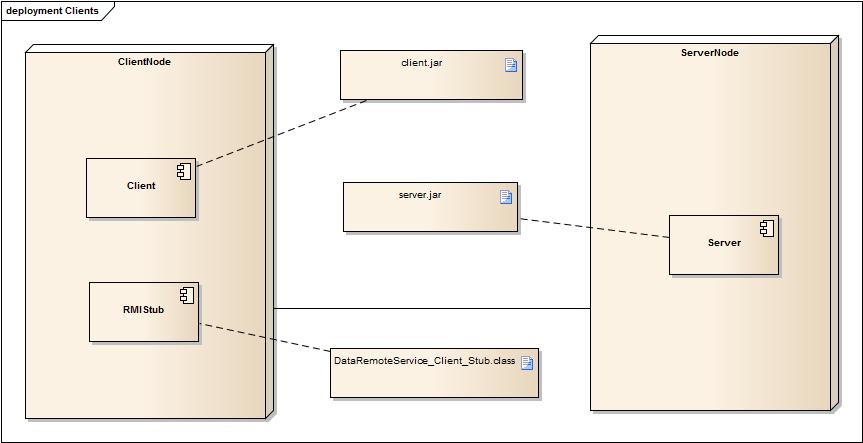
在高校选课系统中，会有多个客户端进程和一个服务器端进程，其进程图如图5所示。结合部署图，客户端进程是在客户端机器上运行，服务器端进程在服务器端机器上运行。



**图5 进程图**

**4.3物理部署**

高校选课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JDK6.0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6所示。

****

**图6 部署图**

**5、接口视角**

**5.1 模块的职责**

客户端模块和服务器模块视图分别如图7和图8所示。客户端各层和服务器端各层的职责分别如表2和表3所示。



**图7 客户端模块视图**



**图8 服务器端模块视图**

**表2 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 | 基于窗口的高校选课客户端用户界面 |
| 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3 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，启动用户界面 |
| 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 |

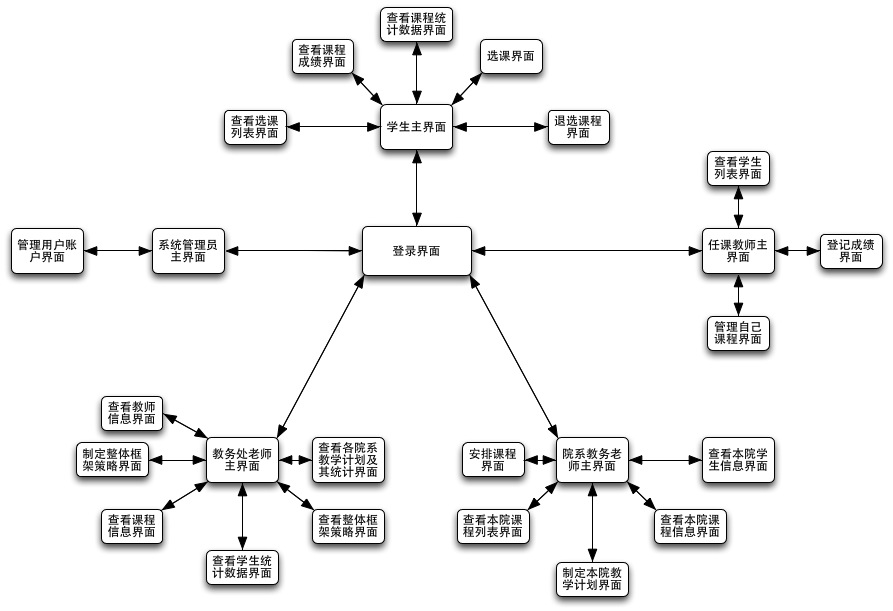
每一层只是使用下方直接接触的层。层与层之间仅仅是通过接口的调用来完成的。层之间调用的接口如表4所示。

**表4 层之间调用的接口**

|  |  |  |
| --- | --- | --- |
| **接口** | **服务调用方** | **服务提供方** |
| UserBLService  InfoBLService  PlanBLService  CourseBLService | 客户端展示层 | 客户端业务逻辑层 |
| UserDataService  InfoDataService  PlanDataService  CourseDataService  DatabaseFactory | 客户端业务逻辑层 | 服务器端数据层 |

**5.2用户界面层的分解**

根据需求，系统存在26个用户界面：登录界面、学生主界面、任课教师主界面、院系教务老师主界面、教务处老师主界面、系统管理员主界面、查看选课列表界面、查看课程成绩界面、查看课程统计数据界面、选课界面、退选课程界面、查看学生列表界面、登记成绩界面、管理自己的课程界面、安排课程界面、查看本院课程列表界面、指定本院教学计划界面、查看本院课程信息界面、查看本院学生信息界面、查看教师信息界面、制定整体框架策略界面、查看课程信息界面、查看学生统计数据界面、查看整体框架策略界面、查看各院系教学计划及其统计界面、管理用户账户界面。界面跳转如图10所示。

****

**图10 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11所示。

****

**图11 用户界面类**

**5.2.1用户界面模块的职责**

如表5所示为用户界面层模块的职责

**表5 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跳转 |

**5.2.2用户界面模块的接口规范**

用户界面层模块的接口规范如表6所示。

**表6 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7所示。

**表7用户界面层模块需要的服务接口**

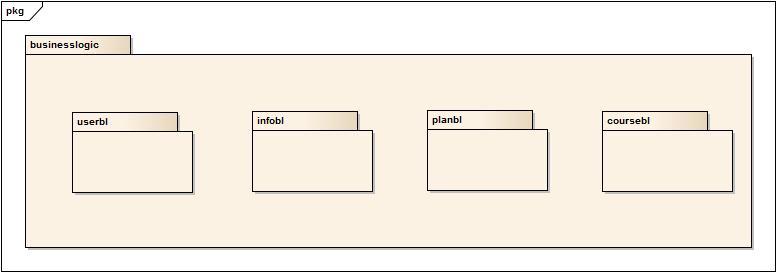
|  |  |
| --- | --- |
| **服务名** | **服务** |
| businesslogicservice.UserBLService | 登录界面的业逻辑（身份验证）接口  用户的增删改查的业务逻辑接口 |
| businesslogicservice.InfoBLService | 查看界面的业务逻辑接口 |
| businesslogicservice.PlanBLService | 框架策略的制定和修改的业务逻辑接口  教学计划的制定和修改的业务逻辑接口 |
| businesslogicservice.CourseBLService | 课程操作（选课、退课、安排课程）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。例如，userbl模块负责处理登录界面的业务逻辑，coursebl负责处理选择与课程有关的界面的业务逻辑。业务逻辑层的设计如图12所示。



**图12 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8所示。

**表8 业务逻辑层模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| userbl | 负责实现对应于登录界面所需要的服务 |
| infobl | 负责实现所有查看界面所需要的服务 |
| planbl | 负责实现对应于制定框架策略、制定院系教学计划界面所需要的服务 |
| coursebl | 负责实现对应于关于课程操作界面所需要的服务 |

**5.3.2业务逻辑层模块的接口规范**

**表9 userbl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| User.login | 语法 | public ResultMessage login(long ID,String password) | |
| 前置条件 | password符合输入规则 | |
| 后置条件 | 查找是否存在相同的User，根据输入的password返回登录验证的结果 | |
| User.addUser | 语法 | public void addUser(UserVO vo) | |
| 前置条件 | 无 | |
| 后置条件 | 向系统中输入新增的用户账户信息 | |
| User.deleteUser | 语法 | public void deleteUser(UserVO vo) | |
| 前置条件 | 无 | |
| 后置条件 | 系统查找对应的用户账户信息，并删除 | |
| User.updateUser | 语法 | public void updateUser(UserVO vo) | |
| 前置条件 | 无 | |
| 后置条件 | 系统根据输入的用户信息和用户编号更新用户信息 | |
| User.findUser | 语法 | public UserPO findUser(long UserID) | |
| 前置条件 | 无 | |
| 后置条件 | 根据输入的ID查找用户，并返回用户信息 | |
| **需要的服务（需接口）** | | | |
| **服务名** | | | **服务** |
| DatabaseFactory.getUserDatabase | | | 得到User数据库服务的引用 |
| UserDataService.insert(UserPO po) | | | 在数据库中插入UserPO对象 |
| UserDataService.check(UserPO po) | | | 得到根据输入的ID和password验证的结果 |
| UserDataService.delete(UserPO po) | | | 在数据库中删除一个po对象 |
| userDataService.update(UserPO po) | | | 在数据库中更新一个po对象 |
| Info.showUserInfo(long ID) | | | 得到根据输入的ID返回的用户信息 |

**表10 infobl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| Info.showCourseInfo | 语法 | public courseVO showCourseInfo(long ID) | |
| 前置条件 | 无 | |
| 后置条件 | 查找是否存在相同的courseID,返回对应的课程信息 | |
| Info.showUserInfo | 语法 | public userVO showUserInfo(long userID) | |
| 前置条件 | 无 | |
| 后置条件 | 查找是否存在相同的userID,返回对应的用户信息 | |
| Info.showPlanInfo | 语法 | public planVO showPlanInfo(long planID) | |
| 前置条件 | 无 | |
| 后置条件 | 查找是否存在相同的planID,返回对应的框架策略或者教学计划信息。 | |
| **需要的服务（需接口）** | | | |
| **服务名** | | | **服务** |
| DatabaseFactory.getUserDatabase | | | 得到Info数据库的引用 |
| InfoDataService.getInfo(long ID) | | | 得到根据输入的ID返回的信息 |

**表11 planbl模块的接口规范**

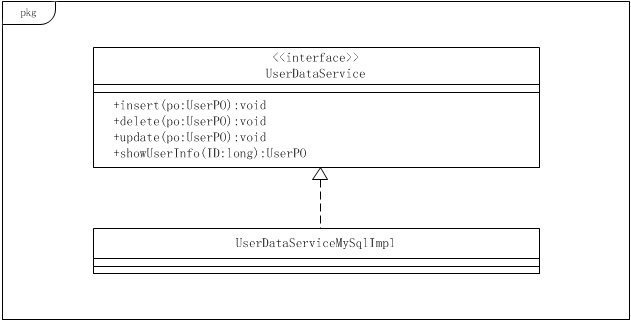
|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| Plan.addFrame | 语法 | public void addFrame(PlanPO po) | |
| 前置条件 | user经验证可以制定框架策略 | |
| 后置条件 | 向系统输入新的框架策略 | |
| Plan.changeFrame | 语法 | public void changeFrame(PlanPO po) | |
| 前置条件 | User经验证能够修改框架策略 | |
| 后置条件 | 向系统输入修改后的框架策略 | |
| Plan.addPlan | 语法 | public void addPlan(PlanPO po) | |
| 前置条件 | user经验证能够制定教学计划 | |
| 后置条件 | 向系统输入教学计划 | |
| Plan.changePlan | 语法 | public void changePlan(PlanPO po) | |
| 前置条件 | user经验证能够制定教学计划 | |
| 后置条件 | 向系统输入修改后的教学计划 | |
| **需要的服务（需接口）** | | | |
| **服务名** | | | **服务** |
| DatabaseFactory.getPlanDatabase | | | 得到Plan数据库的引用 |
| PlanDataservice.insert(PlanPO po) | | | 向Plan数据库中插入planPO对象 |
| PlanDataservice.change(PlanPO po) | | | 向数据库中输入修改过的planPO对象 |

**表12 coursebl模块的接口规范**

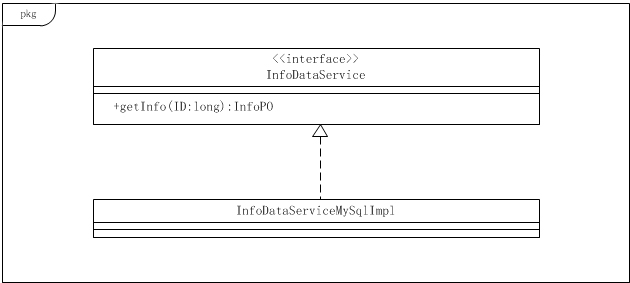
|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| Course.selectCourse | 语法 | public void selectCourse(CourseVOvo,UserVOvo) | |
| 前置条件 | user经验证有选课的权利 | |
| 后置条件 | 向系统输入选课记录 | |
| Course.quitCourse | 语法 | public void quitCourse(CourseVOvo,UserVOvo) | |
| 前置条件 | user经验证有退选课程的权利 | |
| 后置条件 | 删除一条选课记录 | |
| Course.arrangeCourse | 语法 | public void arrangeCourse(CourseVO po,UserVO vo) | |
| 前置条件 | 无 | |
| 后置条件 | 向数据库输入coursePO对象 | |
| **需要的服务（需接口）** | | | |
| **服务名** | | | **服务** |
| DatabaseFactory.getCourseDatabase | | | 获得Course数据库的引用 |
| CourseDatabaseService.insert（coursePO po） | | | 向数据库插入coursePO对象 |
| courseDatabaseService.delete(coursePO po) | | | 在数据库中删除对应的cousePO对象 |

**5.4数据层的分解**

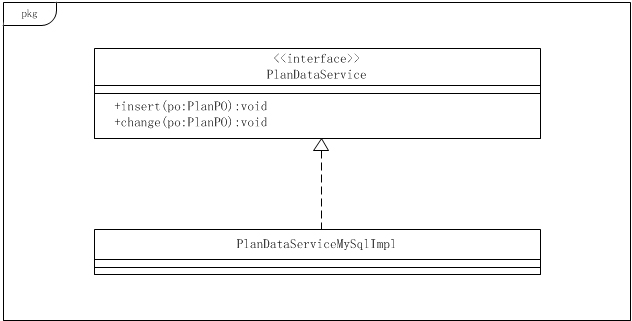
数据层主要给业务逻辑层提供数据访问服务。包括对于持久化数据的增、删、改、查。这些服务由DataService接口提供。在本系统中我们采用数据库形式保存持久化数据。具体如图13—16所示。



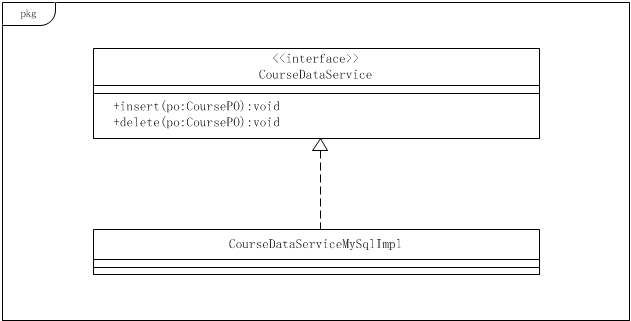
**图13 数据层模块（UserDataService）的描述**



**图14 数据层模块（InfoDataService）的描述**



**图15 数据层模块（PlanDataService）的描述**

****

**图16 数据层模块（CourseDataService）的描述**

**5.4.1数据层模块的职责**

数据层模块的职责如表13－16所示。

**表13 数据层模块（UserDataService）的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| UserDataService | 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 |
| UserDataServiceMySqulImpl | 基于MySql数据库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 |

**表14 数据层模块（InfoDataService）的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| InfoDataService | 持久化数据库的接口，提供查找服务 |
| InfoDataServiceMySqulImpl | 基于MySql数据库的持久化数据库的接口，提供查找服务 |

**表15 数据层模块（PlanDataService）的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| PlanDataService | 持久化数据库的接口，提供集体载入、集体保存、增、改服务 |
| PlanDataServiceMySqulImpl | 基于MySql数据库的持久化数据库的接口，提供集体载入、集体保存、增、改服务 |

**表16 数据层模块（CourseDataService）的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| CourseDataService | 持久化数据库的接口，提供集体载入、集体保存、增、删、服务 |
| CourseDataServiceMySqulImpl | 基于MySql数据库的持久化数据库的接口，提供集体载入、集体保存、增、删服务 |

**5.4.2数据层模块的接口规范**

数据库模块的接口规范如表17－20所示。

**表17 数据层模块（UserDataService）的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| UserDataService.insert | 语法 | public void insert(UserPO po) throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在Mapper中增加一个po记录 |
| UserDataService.delete | 语法 | public void delete(UserPO po) throws RemoteException; |
| 前置条件 | 在Mapper中存在某一个ID 的po需要删除 |
| 后置条件 | 删除一个po |
| UserDataService.update | 语法 | public void update(UserPO po) throws RemoteException; |
| 前置条件 | 在Mapper中存在某一个ID 的po需要更新 |
| 后置条件 | 更新一个po |
| UserDataService.showUserInfo | 语法 | public UserPO showUserInfo (long ID) throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的UserPO的结果 |
| UserDataService.init | 语法 | public void init() throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 初始化持久化数据库 |
| UserDataService.finish | 语法 | public void finish()throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库的使用。 |

**表18 数据层模块（InfoDataService）的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| InfoDataService.getInfo | 语法 | public InfoPO getInfo(long ID) throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的InfoPO的结果 |
| InfoDataService.init | 语法 | public void init() throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 初始化持久化数据库 |
| InfoDataService.finish | 语法 | public void finish()throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库的使用。 |

**表19 数据层模块（PlanDataService）的接口规范**

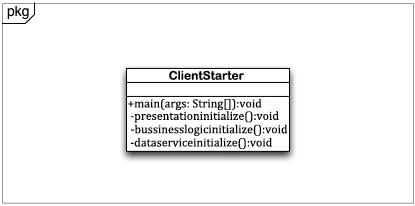
|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| PlanDataService.insert | 语法 | public void insert(PlanPO po) throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在Mapper中增加一个po记录 |
| PlanDataService.change | 语法 | public void change(PlanPO po) throws RemoteException; |
| 前置条件 | 在Mapper中存在某一个ID 的po需要修改 |
| 后置条件 | 修改一个po |
| PlanDataService.init | 语法 | public void init() throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 初始化持久化数据库 |
| PlanDataService.finish | 语法 | public void finish()throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库的使用。 |

**表20 数据层模块（CourseDataService）的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| CourseDataService.insert | 语法 | public void insert(CoursePO po) throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在Mapper中增加一个po记录 |
| CourseDataService.delete | 语法 | public void delete(CoursePO po) throws RemoteException; |
| 前置条件 | 在Mapper中存在某一个ID 的po需要删除 |
| 后置条件 | 删除一个po |
| CourseDataService.init | 语法 | public void init() throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 初始化持久化数据库 |
| CourseDataService.finish | 语法 | public void finish()throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库的使用。 |

**5.5客户端启动模块的分解**

客户端启动模块逻辑上是一个横跨客户端各层的模块，为客户端各层提供初始化启动服务。客户端启动模块主要包含主程序类ClientStarter。主程序类是整个程序的入口，负责整个程序的展示层初始化、业务逻辑层初始化和网络初始化。其结构如图17所示。



**图17 客户端启动模块结构图**

**5.5.1 客户端启动模块的职责**

如表21所示为客户端启动模块的职责。

**表21 客户端启动模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| ClientStarter | 程序入口。负责展示层初始化、业务逻辑层初始化和网络初始化。 |

**5.5.2客户端启动模块的接口规范**

客户端启动模块的接口规范如表22所示。

**表22 客户端启动模块的接口规范**

|  |  |  |
| --- | --- | --- |
| ClientStarter | 语法 | main（args:String[]） |
| 前置条件 | 无 |
| 后置条件 | 根据配置初始化好相应的网络连接并进行展示层和业务逻辑层初始化。 |

客户端启动模块需要的服务接口如表23所示。

**表23 客户端启动模块的服务接口**

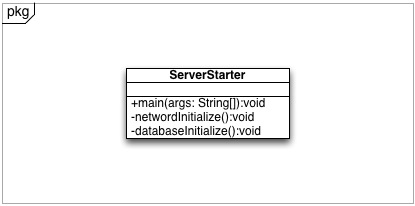
|  |  |
| --- | --- |
| **服务名** | **服务** |
| java.rmi.LocateRegistry.createRegistry() | 启动RMI注册服务 |
| java.rmi.Naming.rebind() | 注册RMI服务 |

**5.5.3客户端启动模块设计原理**

启动模块是一个独立于分层体系之外的模块，可以直接访问各层的接口，完成系统启动的需求。

**5.6服务器端启动模块的分解**

服务器端启动模块逻辑上是一个横跨服务器端各层的模块，为服务器端各层提供初始化启动服务。服务器端启动模块主要包含主程序类SeverStarter。主程序类是整个程序的入口，负责整个程序的网络初始化和数据库初始化。其结构如图18所示。

****

**图18 服务器端启动模块结构图**

**5.6.1 服务器端启动模块的职责**

如表24所示为服务器端启动模块的职责。

**表24 服务器端启动模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| ServerStarter | 程序入口。负责网络初始化和数据库初始化。 |

**5.6.2 服务器端启动模块的接口规范**

服务器端启动模块的接口规范如表25所示。

**表25 服务器端启动模块的接口规范**

|  |  |  |
| --- | --- | --- |
| ServerStarter | 语法 | main（args:String[]） |
| 前置条件 | 无 |
| 后置条件 | 根据配置初始化好相应的网络连接和进行数据库初始化。 |

服务器端启动模块需要的服务接口如表26所示。

**表26 服务器端启动模块的服务接口**

|  |  |
| --- | --- |
| **服务名** | **服务** |
| java.rmi.LocateRegistry.createRegistry() | 启动RMI注册服务 |
| java.rmi.Naming.rebind() | 注册RMI服务 |

**5.6.3服务器端启动模块设计原理**

启动模块是一个独立于分层体系之外的模块，可以直接访问各层的接口，完成系统启动的需求。

**5.7客户端网络模块的分解**

由于数据是存储在服务器上的，客户端访问数据必须通过网络通信层，具体是由客户端的业务逻辑层和服务器端的数据层通过网络通信层进行通信。如图19所示。

具体网络通信层的实现是通过远程方法调用RMI来实现的。



**图19客户端数据通信层模块的描述**

其实网络通信层的实际运作代码是由RMI API所完成的，在客户端我们只需要部署RMI服务的Stub文件和Java RMI类库即可。如表27所示。

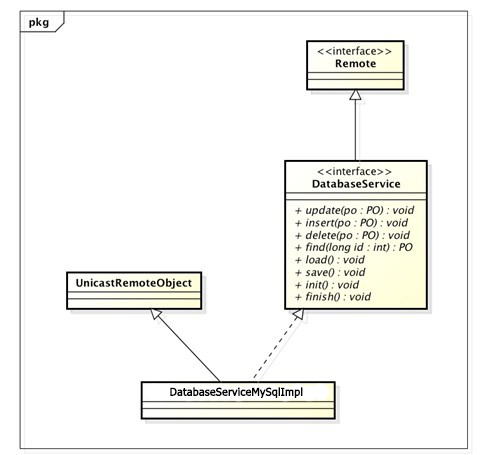
**表27网络通信层模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| IDataRemoteService\_Stub | 给业务逻辑层提供服务接口 |
| Java RMI 类库 | 实现RMI机制 |

**5.8服务器端网络模块的分解**

由于数据是存储在服务器上的，客户端访问数据必须通过网络通信层，具体是由客户端的业务逻辑层和服务器端的数据层通过网络通信层进行通信。

具体服务器端模块中只需要包含Java RMI类库即可。但是其数据层实现RMI的类需要符合以下条件：



**图20 服务器数据层实现RMI的类的描述**

DatabaseService 是被业务逻辑层调用的接口。为了实现RMI， DatabaseService需要继承 Remote接口，而数据层中的实现这个接口的类DatabaseServiceSerializableImpl都需要继承UnicastRemoteObject类。

以下是服务器网络模块的RMI详细的规范：

**表28服务器网络模块的RMI接口规范**

|  |  |
| --- | --- |
| **类/接口** | **规范** |
| 远程服务接口 | extends Remote，方法 throws RemoteException |
| 远程服务的实现类 | extendsUnicastRemoteObject |

**表29服务器网络模块的RMI接口返回值的规范**

|  |  |
| --- | --- |
| **类/接口** | **规范** |
| 作为远程引用传递 | 该类需要extends Remote，方法 throws RemoteException |
| 通过数据库进行复制 | Implements MySql |

**6、信息视角**

**6.1 数据持久化对象**

系统的PO类就是对应的相应的实体类，在此只做简单的介绍。

* UserPO类包含用户的用户名、密码属性。
* InfoPO类包含课程的各项信息、教师的各项信息、学生的各项信息以及相关的统计数据。
* PlanPO类包含整体框架策略和院系教学计划的各项信息。
* CoursePO类保存选课和退选时的数据，包含课程信息和学号，同时还包含了可选课程的各项信息。

持久化用户对象UserPO的定义如图21所示。

public class UserPO{

int id;

String name;

String password;

UserRole role;

public UserPO(int I, String n, String p,UserRole r){

id = i;

name = n;

password = p;

role = r;

}

public String getName(){

return name;

}

public int getID(){

return id;

}

public String getPassword(){

return password;

}

Public UserRole getRole(){

Return role;

}

}

**图持久化用户对象UserPO的定义**

**图21 持久化用户对象UserPO的定义**

**6.2数据库表**

数据库中包含User表、Info表、Course表、Plan表