高校选课系统

软件详细设计描述文档

**V2.0 正式版**

南京大学软件学院

焖烧小鲍鱼工作组

2013-10-23

**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、体系结构设计概述 ………………………………………………………………3

4、中层设计 …………………………………………………………………………3

4.1客户端展示层模块的静态结构和动态职责 ………………………………5

4.1.1客户端展示层模块局部模块的职责…………………………………5

4.1.2客户端展示层模块局部模块的接口规范……………………………5

4.1.3客户端展示层模块的设计原理………………………………………7

4.2客户端业务逻辑层模块的静态结构和动态职责…………………………7

4.2.1客户端业务逻辑层模块局部模块的职责……………………………7

4.2.2客户端业务逻辑层模块局部模块的接口规范………………………7

4.2.3客户端业务逻辑层模块局部的行为 ……………………………… 20

4.2.4客户端业务逻辑层模块的设计原理 ………………………………26

4.3客户端网络模块的静态结构和动态职责…………………………………26

4.3.1客户端网络模块局部模块的职责 ……………………………… 27

4.4服务器端网络模块的静态结构和动态职责…………………………… 27

4.4.1服务器端网络模块局部模块的职责 ………………………………27

4.4.2服务器端网络模块局部模块的接口规范 …………………………28

4.5服务器端数据层模块的静态结构和动态职责 ………………………… 28

4.5.1服务器端数据层模块局部模块的职责 ……………………………28

4.5.2服务器端数据层模块局部模块的接口规范 ………………………29

4.5.3服务器算数据层模块的设计原理 …………………………………34

5、依赖关系………………………………………………………………………… 34

# 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 | 2013/10/23 | 最初草稿 | V1.0草稿版 |
| 张梦雨 刘劭 王吉 宋程宇 | 2013/11/9 | 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详细设计，达到指导后续软件构造的目的，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**

[Bin2012]骆斌，丁二玉，刘钦.软件工程与计算（卷二）——软件开发的技术基础.北京：机械工程出版社，2012.12;

高校选课系统用例文档、高校选课系统软件需求规格说明和高校选课系统体系结构设计文档。

**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、体系结构设计概述**

参考高校选课系统体系结构设计文档中对体系结构设计的概述。

**4、中层设计**

中层设计的基本思路依托了软件体系的结构视角。

在高校选课系统中，可以分别在客户端和服务器端都添加系统启动模块专门负责系统的初始化启动工作，并使它横跨系统的分层结构，可以直接访问到各层。客户端则具体分为展示层、业务逻辑层和网络模块，而服务器端则包含网络模块和数据层。具体的客户端和服务器端模块视图分别如图1和图2所示。客户端各层和服务器端各层的职责分别如表1和表2所示。



**图1 客户端模块视图**



**图2 服务器端模块视图**

**表1 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2 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。

下面，根据以上对客户端和服务器的分解对各个模块进行分析与设计。

**4.1客户端展示层模块的静态结构和动态职责**

客户端展示层由八个模块组成，分别是mainui, userui, adminui, frameui, planui, infoui, courseui, courseselectionui。mainui模块负责界面的跳转；userui模块包含用户登录和修改密码界面。adminui模块包含用户的增删改查界面。frameui模块包含制定整体框架策略界面。planui模块包含制定本院教学计划界面。infoui模块包含各种信息资料查看界面。courseui模块包含发布课程、管理自己的课程、登记成绩界面。courseselectionui包含选择课程界面、退选课程界面。

**4.1.1客户端展示层模块局部模块的职责**

客户端展示层模块的职责如下表所示。

**表3 客户端展示层模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| mainui | 负责界面的跳转 |
| userui | 包含用户登录和修改密码界面 |
| adminui | 包含用户的增删改查界面 |
| frameui | 包含制定整体框架策略界面 |
| planui | 包含制定本院教学计划界面 |
| infoui | 包含各种信息资料查看界面 |
| courseui | 包含发布课程、管理自己的课程、登记成绩界面 |
| courseselectionui | 包含选择课程界面、退选课程界面 |

**4.1.2客户端展示层模块局部模块的接口规范**

**4.1.2.1 mainui模块的接口规范**

mainui模块的接口规范如下表所示。

**表4 mainui模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的接口（供接口）** | | |
| mainui.init | 语法 | public void init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 根据逻辑进行界面跳转。 |

**4.1.2.2 userui模块的接口规范**

userui模块的接口规范如下表所示。

**表5 userui模块的接口规范**

|  |  |
| --- | --- |
| **需要的接口（需接口）** | |
| **服务名** | **服务** |
| businesslogicservice.UserBLService | 登录界面的业务逻辑（身份验证）接口  用户的修改密码的业务逻辑接口 |

**4.1.2.3 adminui模块的接口规范**

adminui模块的接口规范如下表所示。

**表6 adminui模块的接口规范**

|  |  |
| --- | --- |
| **需要的接口（需接口）** | |
| **服务名** | **服务** |
| businesslogicservice.AdminBLService | 用户的增删改查的业务逻辑接口 |

**4.1.2.4 frame模块的接口规范**

frameui模块的接口规范如下表所示。

**表7 frameui模块的接口规范**

|  |  |
| --- | --- |
| **需要的接口（需接口）** | |
| **服务名** | **服务** |
| businesslogicservice.FrameBLService | 制定整体框架策略的业务逻辑接口 |

**4.1.2.5 planui模块的接口规范**

planui模块的接口规范如下表所示。

**表8 planui模块的接口规范**

|  |  |
| --- | --- |
| **需要的接口（需接口）** | |
| **服务名** | **服务** |
| businesslogicservice.PlanBLService | 制定本院教学计划的业务逻辑接口 |

**4.1.2.6 infoui模块的接口规范**

infoui模块的接口规范如下表所示。

**表9 infoui模块的接口规范**

|  |  |
| --- | --- |
| **需要的接口（需接口）** | |
| **服务名** | **服务** |
| businesslogicservice.InfoBLService | 用户查看各种信息的业务逻辑接口 |

**4.1.2.7 courseui模块的接口规范**

courseui模块的接口规范如下表所示。

**表10 courseui模块的接口规范**

|  |  |
| --- | --- |
| **需要的接口（需接口）** | |
| **服务名** | **服务** |
| businesslogicservice.CourseBLService | 发布课程、管理自己的课程、登记成绩的业务逻辑接口 |

**4.1.2.8 courseselectionui模块的接口规范**

courseseletionui模块的接口规范如下表所示。

**表11 courseseletionui模块的接口规范**

|  |  |
| --- | --- |
| **需要的接口（需接口）** | |
| **服务名** | **服务** |
| businesslogicservice.CourseSelectionBLService | 选择课程的业务逻辑接口 |

**4.1.3 客户端展示层模块局部模块的设计原理**

客户端展示层模块使用了集中式的控制风格。每个界面都通过mainui模块实现跳转。

客户端展示层模块使用了集中式的控制风格。每一个界面的职责从mainui之中分离并通过mainui模块实现跳转。

**4.2客户端业务逻辑层模块的静态结构和动态职责**

客户端业务逻辑层由7个模块组成，分别是userbl, adminbl, infobl, coursebl, planbl, framebl, courseselectionbl。userbl模块负责用户的登录以及修改密码的安全性需求；infobl模块负责所有信息查看，包括教师信息、学生信息、课程信息、框架策略信息和教学计划信息的查看。coursebl模块负责所有课程的制定。planbl负责框架策略的制定、修改。Adminbl模块负责管理员对用户信息的增删改。Framebl负责框架策略的制定、修改。Courseselectionbl负责学生的选课退课。

**4.2.1客户端业务逻辑层模块局部模块的职责**

客户端业务逻辑层层模块的职责如下表所示。

**表12 客户端业务逻辑层模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| userbl | 负责实现对应登录界面的服务 |
| infobl | 负责实现对应查看信息界面所需要的服务 |
| coursebl | 负责实现对应安排课程、填写课程信息界面所需要的服务 |
| planbl | 负责实现对应教学计划所需要的服务 |
| courseselectionbl | 负责实现对应选课和退课所需要的服务 |
| framebl | 负责实现制定\修改框架策略所需要的服务 |
| adminbl | 负责实现用户账户信息的增删改所需要的服务 |

**4.2.2客户端业务逻辑层模块局部模块的接口规范**

**4.2.2.1 userbl模块**

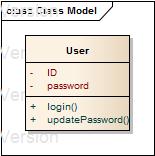
**(1)模块概述**

userbl模块承担用户登录和修改密码的功能需求和安全性的非功能需求，具体可参见需求规格说明文档功能需求和相关非功能需求。

**(2)整体结构**

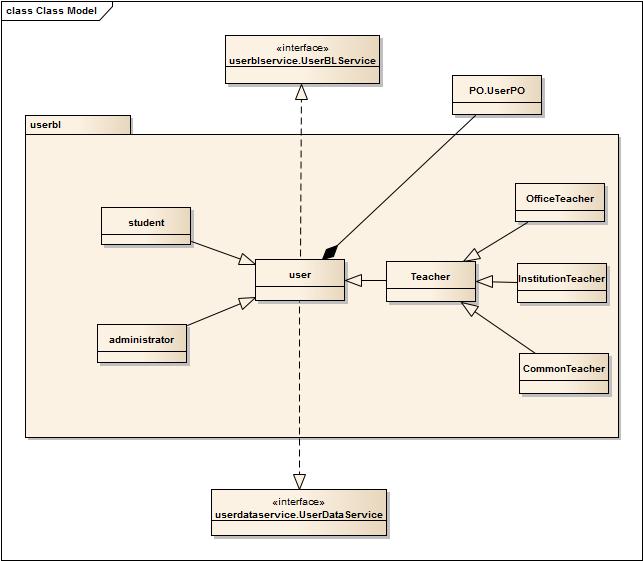
根据体系结构的设计，我们在展示层和业务逻辑层之间添加了userblservice接口。在业务逻辑层和数据层之间添加了userdataservice接口。Userblservice有login和updatePassword方法。因业务逻辑处理较少，根据专家模式没有添加controller类。其余类是User类的子类或子类的子类。

User类的属性如图3所示。

****

**图3 user类的关键属性**

userbl模块的设计如图所示

****

**图4 userbl各个类的设计**

**(3)模块内部类的接口规范**

userbl的接口规范如表13所示

**表13 userbl的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的接口（供接口）** | | |
| User.login | **语法** | public ResultMessage login(String ID,String password) |
| **前置条件** | 用户启动一个登录任务 |
| **后置条件** | 返回登录验证的结果 |
| User.updatePassword | **语法** | public ResultMessage updatePassword(String ID,String password,String newPassword) |
| **前置条件** | 用户启动修改密码任务 |
| **后置条件** | 系统更新用户密码信息 |
| **需要的接口（需接口）** | | |
| **服务名** | **服务** | |
| UserDataService.find | 根据用户ID返回用户信息。 | |
| DataFactory.getUserDatabase | 得到user数据库的引用 | |

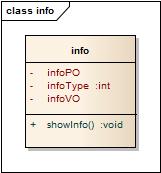
**4.2.2.2 infobl模块**

**(1)模块概述**

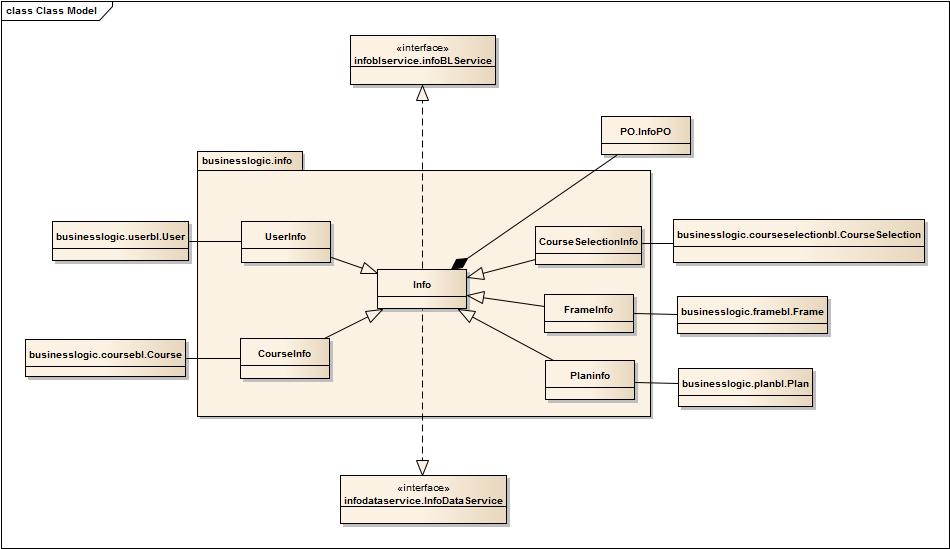
infobl模块负责所有查看信息的功能需求。具体功能需求和非功能需求可参见需求规格说明文档和体系结构设计文档**。**

**(2)整体结构**

info类作为整个模块最重要的类，实现了展示层与业务逻辑层的infoblservice.InfoBLService接口和业务逻辑层与数据层的infoDataService接口。提供了showInfo方法，展示需要查看的各类信息。其他的类为Info类的子类或子类的子类。各个类与其他模块的存在的交互如图5所示。Infobl模块的关键类属性如图6所示。



**图5 info类的关键属性**

****

**图6 infobl模块各个类的设计**

**(3)模块内部类的接口规范**

模块内部类的接口规范如表14所示

**表14 infobl的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的接口（供接口）** | | | |
| Info.showUserInfo | **语法** | | public AdminInfoVO showUserInfo(User user) |
| **前置条件** | | 管理员启动一个查看用户信息任务 |
| **后置条件** | | 返回对应的用户信息 |
| info.showPlanInfo | **语法** | | public PlanInfoVO showPlanInfo(String institute) |
| **前置条件** | | 用户启动一个查看教学计划任务 |
| **后置条件** | | 返回对应的教学计划信息 |
| info.showCourseInfo | **语法** | | public CourseInfoVO showCourseInfo(User user, Student student) |
| **前置条件** | | 用户启动一个查看课程信息任务 |
| **后置条件** | | 返回对应的课程信息 |
| Info.showCourseSelectionInfo | **语法** | | public CourseSelectionInfoVO showCourseSeInfo(User user,CourseSelection courseSelection) |
| **前置条件** | | 用户启动一个查看选课信息的任务 |
| **后置条件** | | 返回对应的选课信息 |
| Info.showFrameInfo | **语法** | | public FrameInfoVO showFrameInfo(InfoVO vo) |
| **前置条件** | | 用户启动一个查看框架策略任务 |
| **后置条件** | | 返回对应的框架策略信息 |
| Info.showCourseStaticInfo | **语法** | | PublicCourseStatisticInfoVO showCourseStaticInfo(Student student) |
| **前置条件** | | 用户启动一个查看某学生的课程统计信息 |
| **后置条件** | | 返回对应的课程统计信息 |
| Info.showStudentStatisticInfo | **语法** | | public StudentStatisticInfoVO showStudentStatisticInfo() |
| **前置条件** | | 用户启动一个查看学生的准入准出数据 |
| **后置条件** | | 返回对应的学生准入准出数据 |
| **需要的接口（需接口）** | | | |
| 服务名 | | 服务 | |
| InfoDataService.findUserInfo | | 返回User信息 | |
| InfoDataService.finsPlan | | 返回Plan信息 | |
| InfoDataService.findCourse | | 返回Course信息 | |
| InfoDataService.findCourseSelection | | 返回CourseSelection信息 | |
| InfoDataService.findFrame | | 返回Frame信息 | |
| InfoDataService.find | | 初始化查找 | |
| DatabaseFactory.getInfoDatabase | | 得到Info数据库的服务的引用 | |

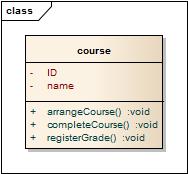
**4.2.2.3 coursebl模块**

**(1)模块概述**

coursebl模块负责实现安排课程，填写课程信息，登记课程的功能需求，具体功能需求和非功能需求请参见需求规格说明文档和体系结构设计文档。

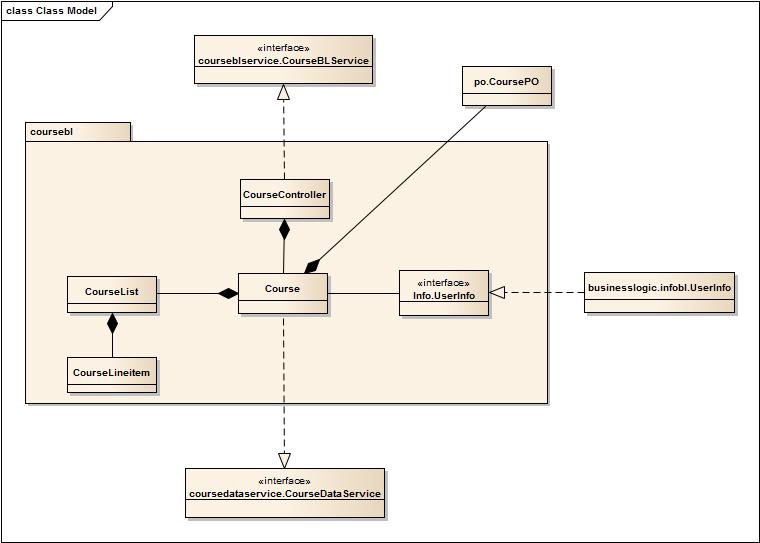
**(2)整体结构**

coursebl模块实现了展示层与业务逻辑层之间的courseblservice接口和业务逻辑层与数据层之间的coursedataservice接口。提供了arrangeCourse，completeCourse，RegisterGrade方法，与info.UserInfo存在依赖关系。courseList是CourseLineItem的容器类，courseLineItem是数据库中课程信息的对应一条字段。关键类图的属性如图7所示。

****

**图7 course类的属性**

coursebl模块各个类的设计如图8所示

****

**图8 coursebl模块各个类的设计结构图**

**(3)coursebl模块的内部接口规范**

coursebl的接口规范如表15所示

**表15 coursebl的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的接口** | | | |
| Course.arrangeCourse | **语法** | | public ResultMessage arrangeCourse(CourseList courseList) |
| **前置条件** | | 院系教务老师启动一个安排课程任务 |
| **后置条件** | | 生成一条课程信息，返回安排课程成功与否信息 |
| Course.registerGrade | **语法** | | Public ResultMessage registerGrade(RegisterGradeVO registerGradeVO, UserVO userVO) |
| **前置条件** | | 任课教师启动一个登记课程成绩任务 |
| **后置条件** | | 系统记录输入的成绩 |
| Course.completeCourse | **语法** | | public ResultMessage completeCourse(CompleteCourseVO completeCourseVO,  UserVO userVO) |
| **前置条件** | | 任课教师启动一个填写课程信息的任务 |
| **后置条件** | | 系统记录课程信息，返回操作结果 |
| Course.updateCourse | **语法** | | public ResultMessage updateCourse(CompleteCourseVO completeCourseVO, UserVO userVO) |
| **前置条件** | | 院系教务老师、任课教师启动一个修改课程信息的任务 |
| **后置条件** | | 系统记录更新后的课程数据，返回操作结果 |
| **需要的接口（需接口）** | | | |
| **服务名** | | **服务** | |
| CourseDataService.insert | | 在数据库中插入coursePO对象 | |
| CourseDataService.delete | | 在数据库中删除coursePO对象 | |
| CourseDataService.update | | 在数据库中修改coursePO对象 | |
| DatabaseFactory.getCourseDatabase | | 得到Course数据库的引用 | |
| Info.showCourseInfo | | 得到course的信息 | |

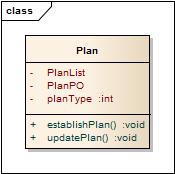
**4.2.2.4 planbl模块**

**(1)模块概述**

planbl模块负责实现制定教学计划和修改教学计划的功能需求，具体请参见需求规格说明文档和体系结构设计文档

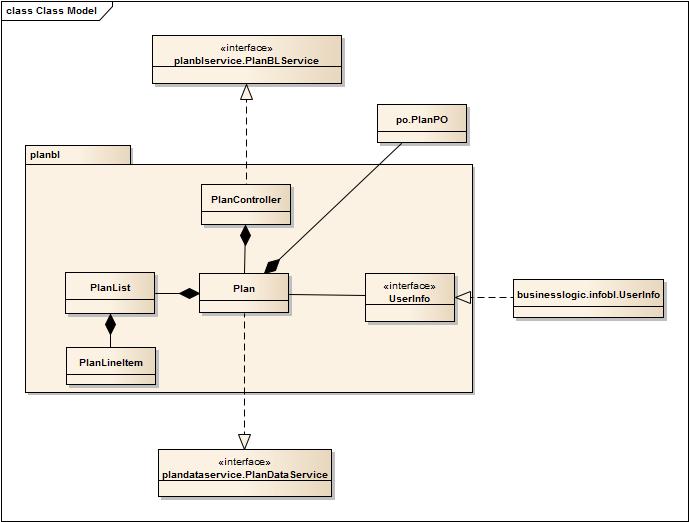
**(2)整体结构**

planbl模块实现了展示层与业务逻辑层的接口planblservice和业务逻辑层与数据层的接口plandataservice。提供了establishPlan和updatePlan方法。Plancontroller和plan将业务的实现和控制逻辑分开。Planlist是planLineItem的容器类，planLineItem是数据库中对应的一条plan记录。Plan关键类的属性如图9所示

****

**图9 plan类的关键属性**

planbl模块各个类的设计如图10所示

****

**图10 planbl类各个类的设计**

**(3)模块内部类的接口规范**

planbl模块的接口规范如表16所示

**表16 planbl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的接口（供接口）** | | | |
| Plan.establishPlan | **语法** | | public void establishPlan(PlanList planList) |
| **前置条件** | | 院系教务老师启动一个制定教学计划的任务 |
| **后置条件** | | 系统记录教学计划 |
| Plan.updatePlan | **语法** | | public void updatePlan(PlanVO planVO) |
| **前置条件** | | 院系教务老师启动一个修改教学计划的任务 |
| **后置条件** | | 系统记录修改后的教学计划 |
| **需要的接口（需接口）** | | | |
| **服务名** | | **服务** | |
| PlanDataService.establish | | 向数据库插入planPO对象 | |
| PlanDataService.update | | 更新数据库的planPO对象 | |
| Info.showPlanInfo | | 获得plan的信息 | |
| DatabaseFactory.getPlanDatabase | | 获得Plan数据库的引用 | |

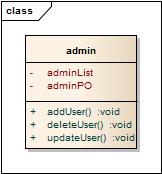
**4.2.2.5 adminbl模块**

**(1)模块概述**

adminbl模块实现管理员对用户账户的增删改查的功能需求，具体的功能需求和非功能需求请参见需求规格说明文档和体结构设计文档。

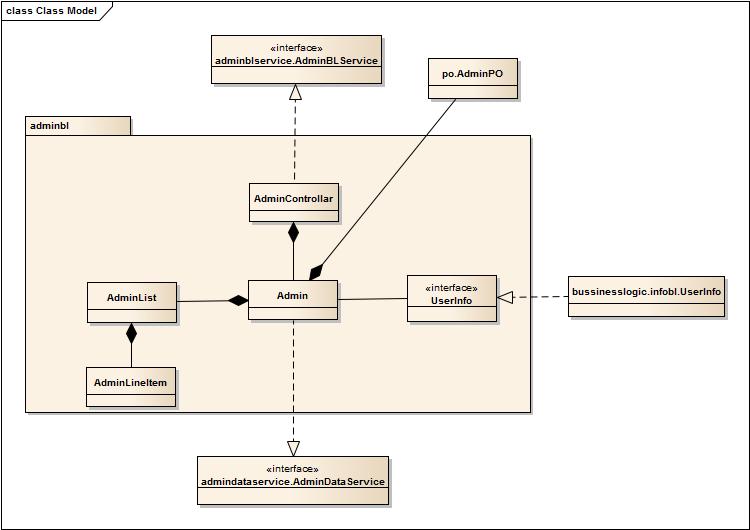
**(2)整体结构**

Adminbl模块实现了展示层与业务逻辑层之间的adminblservice结构以及业务逻辑层和数据层之间的admindataservice接口。提供了addUser,deleteUser, updateUser方法。Admincontroller和admin类将逻辑控制和业务实现分离，adminList是adminLineItem的容器类，adminLineItem对应着数据库中的用户表信息字段。Admin关键类的属性如图11所示

****

**图11 admin类的关键属性**

adminbl模块各个类的设计如图12所示

****

**图12 adminbl模块各个类的设计**

**(3)模块内部类的接口规范**

adminbl的接口规范如表17所示

**表17 adminbl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的接口（供接口）** | | | |
| Admin.addUser | **语法** | public ResultMessage addUser(AdminList adminList) | |
| **前置条件** | 管理员启动一个添加用户任务 | |
| **后置条件** | 返回添加用户的操作结果 | |
| Admin.delUser | **语法** | public ResultMessage deleteUser(AdminVO adminVO) | |
| **前置条件** | 管理员启动一个删除用户的任务 | |
| **后置条件** | 返回删除用户的结果 | |
| Admin.updateUser | **语法** | public ResultMessage(AdminVO adminVO) | |
| **前置条件** | 管理员启动一个修改用户信息的任务 | |
| **后置条件** | 系统更新用户信息 | |
| **需要的接口（需接口）** | | | |
| **服务名** | | | 服务 |
| AdminDataService.insert | | | 向数据库中加入用户信息 |
| AdminDataService.delete | | | 删除数据库中的用户信息 |
| AdminDataService.update | | | 更新数据库中的用户信息 |
| DatabaseFactory.getAdminDatabase | | | 获得Admin数据库的引用 |

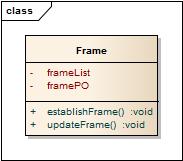
**4.2.2.6 framebl模块**

**(1)模块概述**

framebl模块实现了制定和修改框架策略的功能需求，具体功能需求和非功能需求请参见需求规格说明文档和体系结构设计文档。

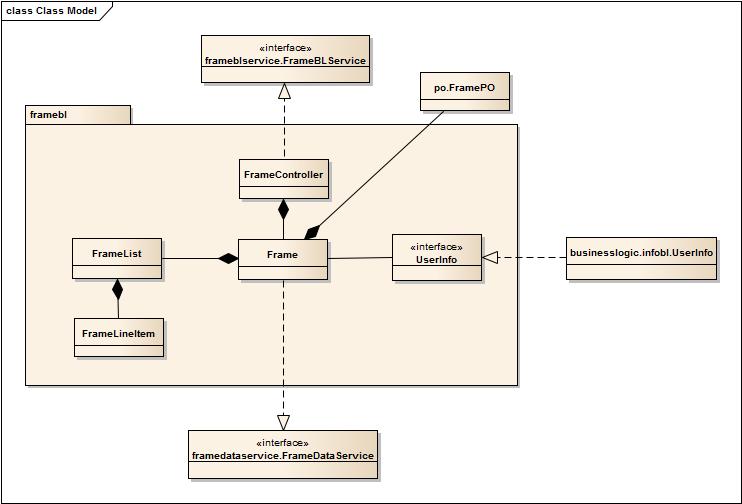
**(2)整体结构**

framebl模块实现了展示层与业务逻辑层之间的frameblservice接口以及业务逻辑层和数据层之间的framedataservice接口。提供了establishFrame ,updateFrame方法。Admincontroller和admin类将逻辑控制和业务实现分离frameList是frameLineItem的容器类，负责框架策略的数据统计功能。frameLineItem对应着数据库中的框架策略表信息字段。framebl关键类的属性如图13所示

****

**图13 framebl关键类属性**

framebl模块各个类的设计如图14所示

****

**图14 framebl各个类的设计**

**(3)模块内部类的接口规范**

framebl模块的接口规范表18所示

**表18 framebl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的接口（供接口）** | | | |
| Frame.establishFrame | **语法** | | public void establishFrame(FrameList frameList) |
| **前置条件** | | 教务处老师启动一个制定框架策略的任务 |
| **后置条件** | | 系统记录 |
| Frame.updateFrame | **语法** | | public void updateFrame(FrameVO frameVO) |
| **前置条件** | | 教务老师启动一个修改框架策略的任务 |
| **后置条件** | | 系统记录修改后的框架策略 |
| **需要的接口（需接口）** | | | |
| **服务名** | | **服务** | |
| FrameDataService.establish | | 向数据库插入一条FramePO对象 | |
| FrameDataService.update | | 更新数据库中的FramePO对象 | |
| Info.showFrameInfo | | 获得frame的信息 | |
| DatabaseFactory.getAdminDatabase | | 获得Frame数据库的引用 | |

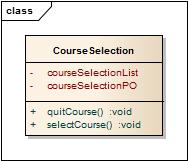
**4.2.2.7courseselectionbl模块**

**(1)模块概述**

courseselectionbl模块是系统的关键模块。实现了学生选课和退课的功能需求。具体功能需求和非功能需求请参见需求规格说明文档和体系结构说明文档。

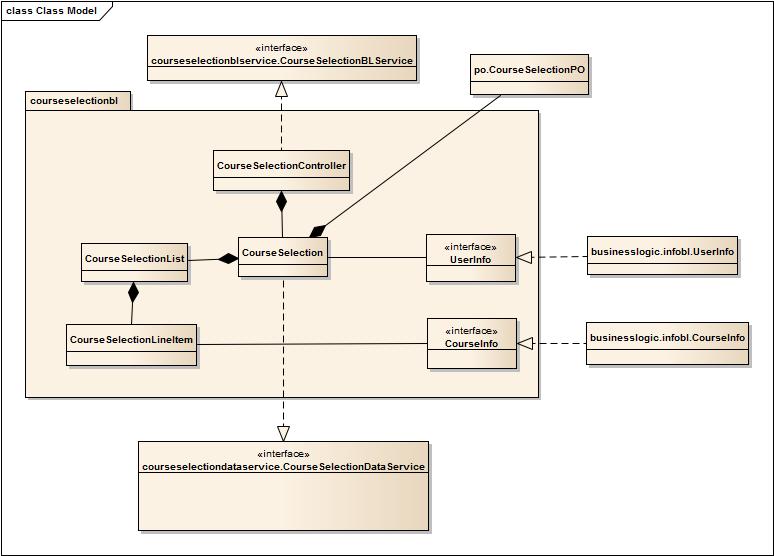
**(2)整体结构**

courseselectionbl模块实现了展示层与业务逻辑层之间的courseselectionblservice接口以及业务逻辑层和数据层之间的courseselectiondataservice接口。提供了selectCourse ,quitCourse方法。courseselectioncontroller和courseselection类将逻辑控制和业务实现分离courseselectionList是courseselectionLineItem的容器类，负责选课的数据统计功能。courseselectionLineItem对应着数据库中的选课信息表字段。courseselectionbl关键类的属性如图15所示

****

**图15 courseselectionbl关键类属性**

courseselectionbl模块各个类的设计如图16所示

****

**图16 courseselectionbl各个类的设计**

**(3)模块内部类的接口规范**

courseselectionbl模块内部的接口规范如表19所示

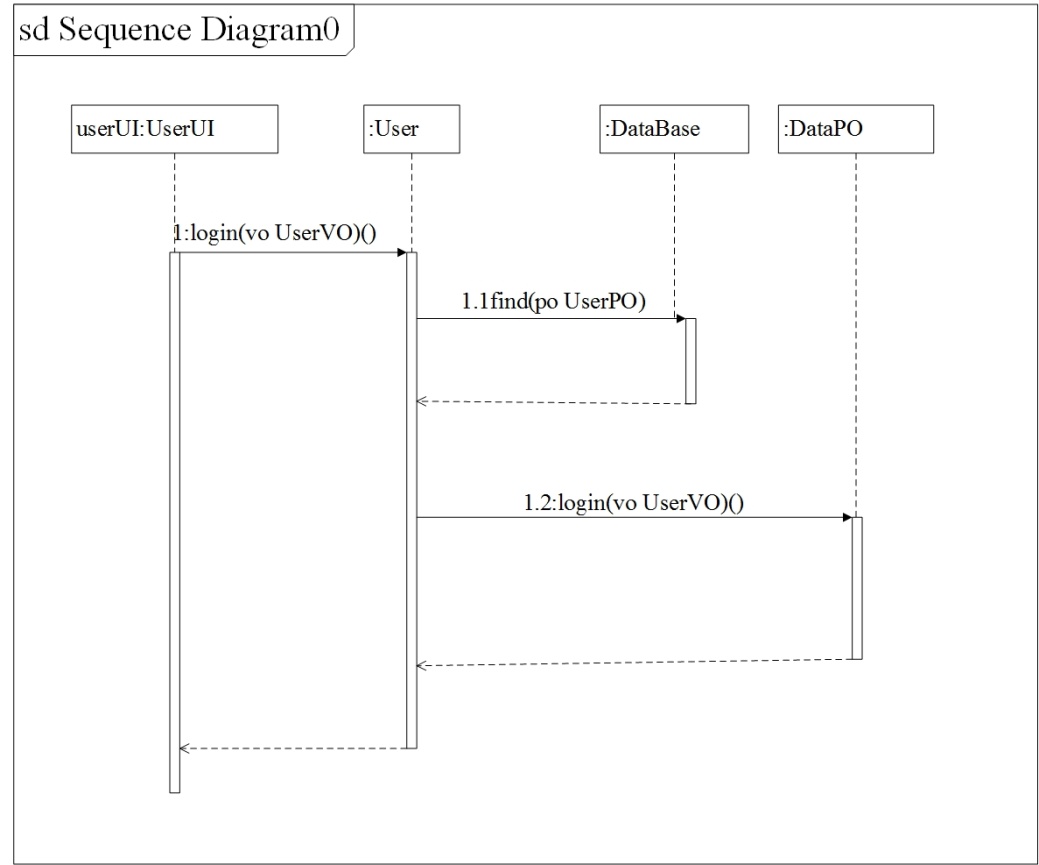
**表19 courseselectionbl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的接口（供接口）** | | | |
| CourseSelection.selectCourse | **语法** | | public ResultMessage selectCourse(CourseSelectionList courseSelectionList,UserVO userVO ) |
| **前置条件** | | 学生启动一个选课任务 |
| **后置条件** | | 生成选课记录，返回选课结果 |
| CourseSelection.quitCourse | **语法** | | public ResultMessage quitCourse(CourseSelectionList courseSelectionList,UserVO userVO ) |
| **前置条件** | | 学生启动一个退选课程任务 |
| **后置条件** | | 删除一条选课记录，返回退课结果 |
| **需要的接口（需接口）** | | | |
| **服务名** | | **服务** | |
| CourseSelectionDataService.insert | | 向数据库中插入CourseSelection对象 | |
| CourseSelectionDataService.delete | | 删除数据库中的CourseSelection对象 | |
| CourseSelectionDataService.update | | 更新数据库中的CourseSelection对象 | |
| Info.getCourseSelectionInfo | | 获得选课信息 | |
| DatabaseFactory.getCourseSeDatabase | | 获得CourseSelection数据库的引用 | |

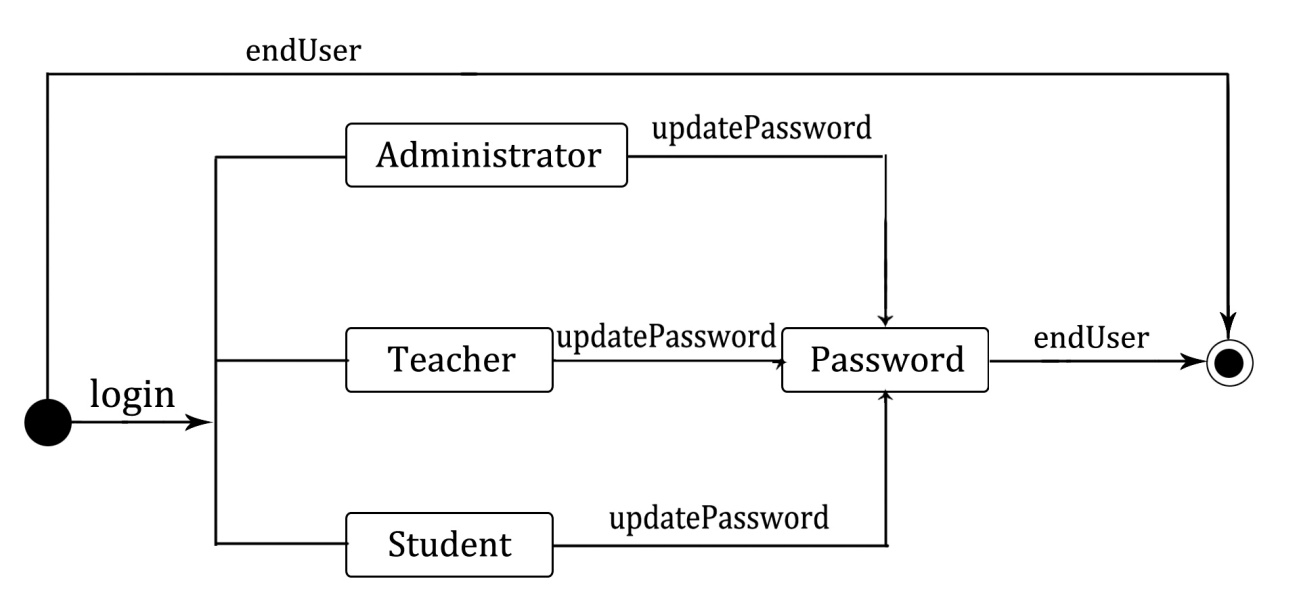
**4.2.3客户端业务逻辑层模块局部的行为**

**4.2.3.1 userbl模块**

如图15所示的顺序图表明了在选课系统中，当用户输入用户ID和password后，登录业务逻辑处理的相关对象之间的协作。

****

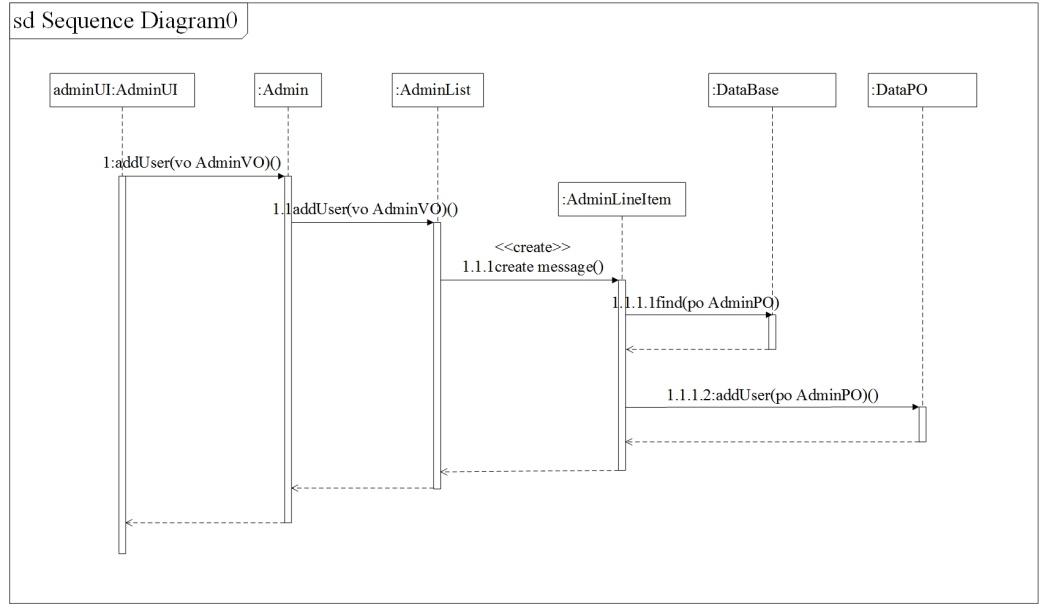
**图17 userbl模块的顺序图**

如图18所示，User对象状态图描述了User对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。随着login方法被 UI调用，User进入Administrator、Student或Teacher状态；之后通过修改密码进入Password状态。****

**图18 userbl模块的状态图**

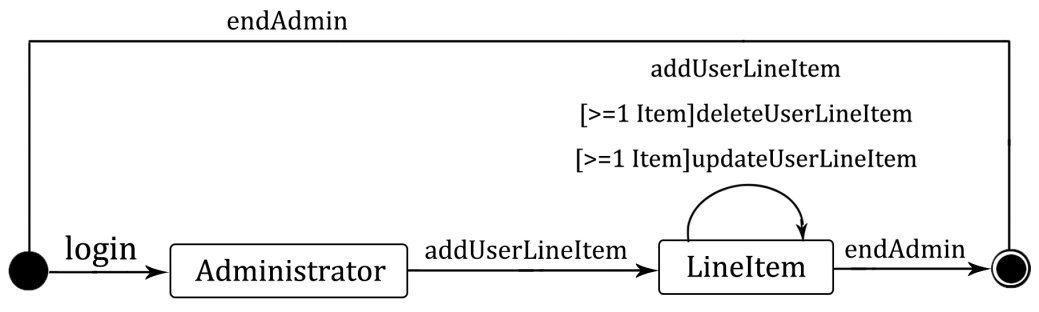
**4.2.3.2 adminbl模块**

如图17所示的顺序图表明了在选课系统中，当管理员输入要增加的用户信息后，添加用户信息业务逻辑处理的相关对象之间的协作。

****

**图19 adminbl添加用户信息的顺序图**

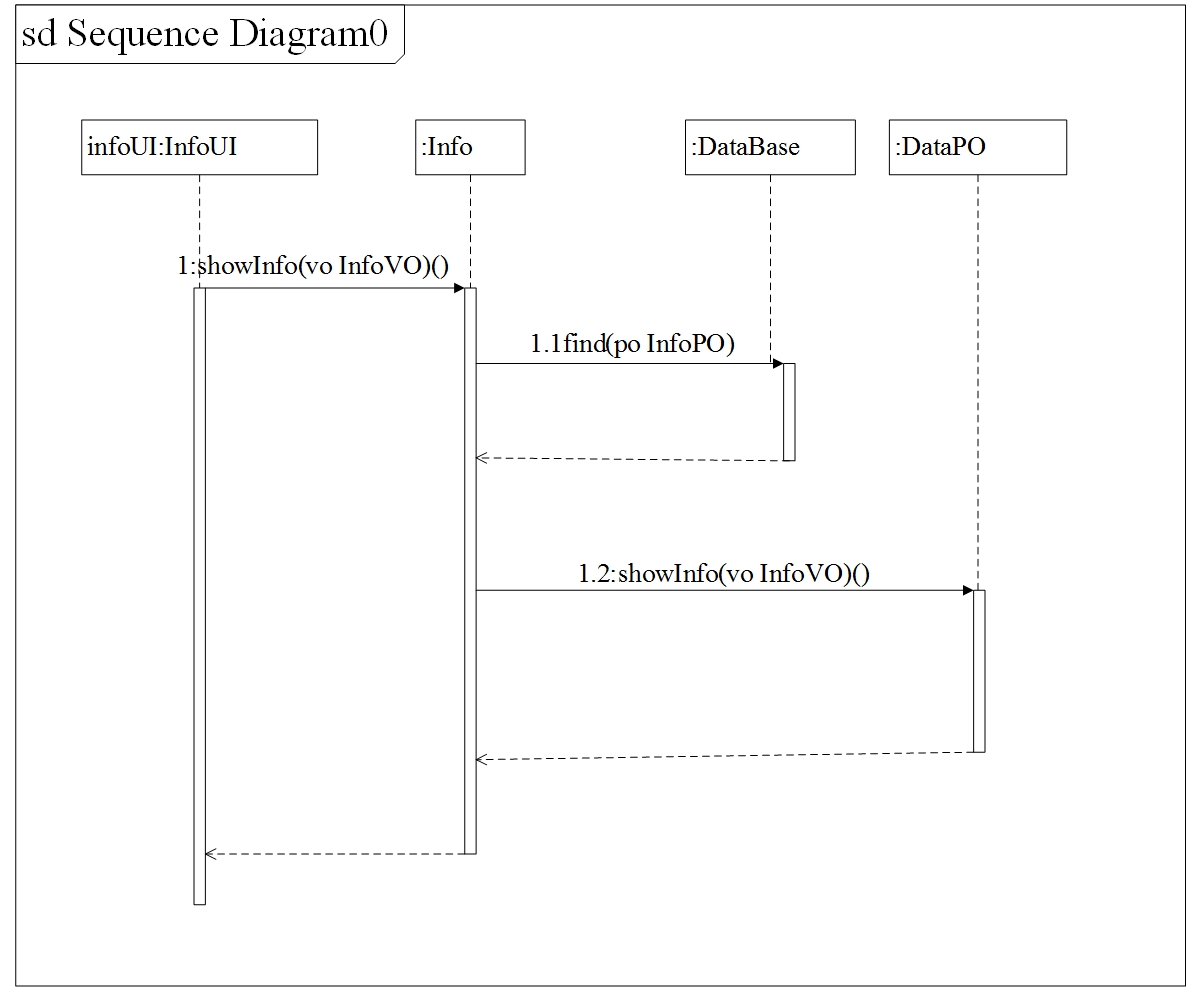
如图20所示，Admin对象状态图描述了Admin对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。随着login方法被 UI调用，Admin进入Administrator 状态；之后通过添加用户进入LineItem状态。

****

**图20 adminbl模块添加用户信息的状态图**

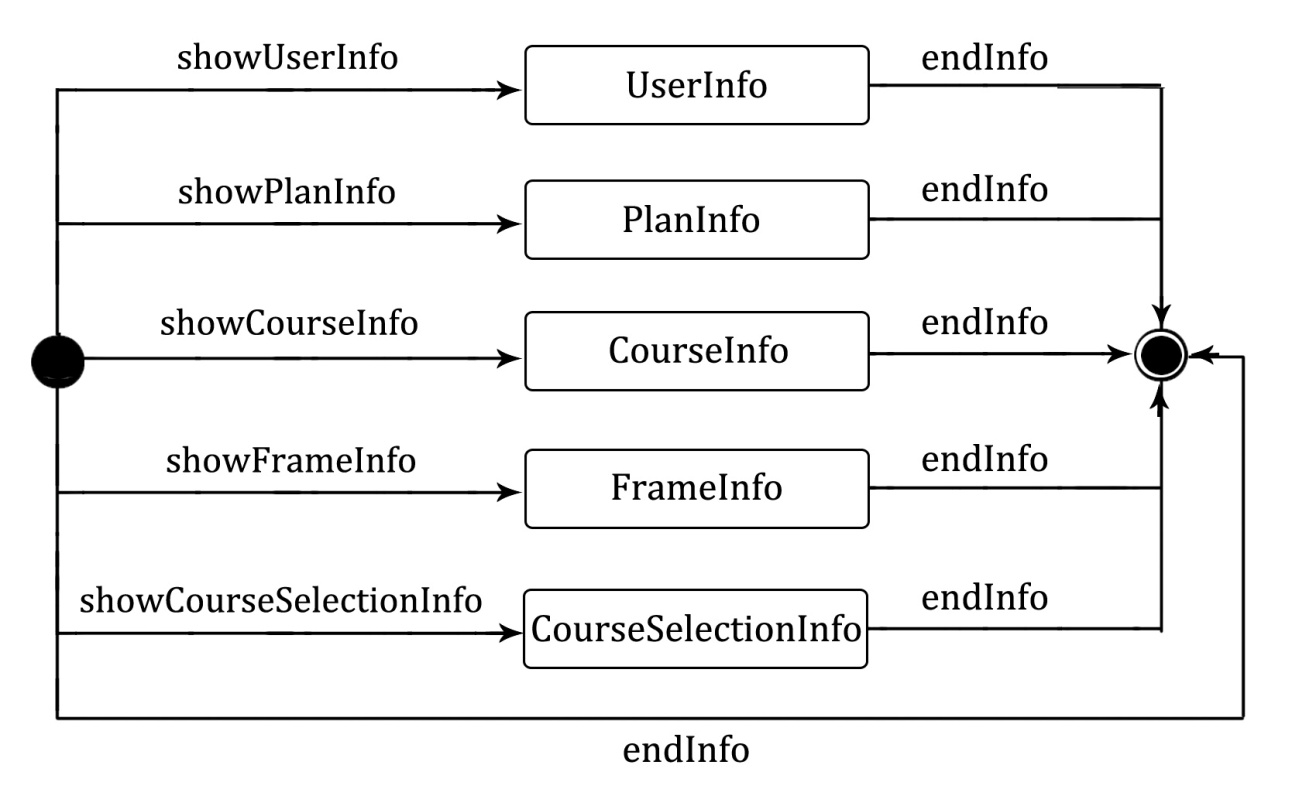
**4.2.3.3 infobl模块**

如图21所示的顺序图表明了在选课系统中，当用户选择查看信息功能后，信息业务逻辑处理的相关对象之间的协作。

****

**图21 infobl模块的顺序图**

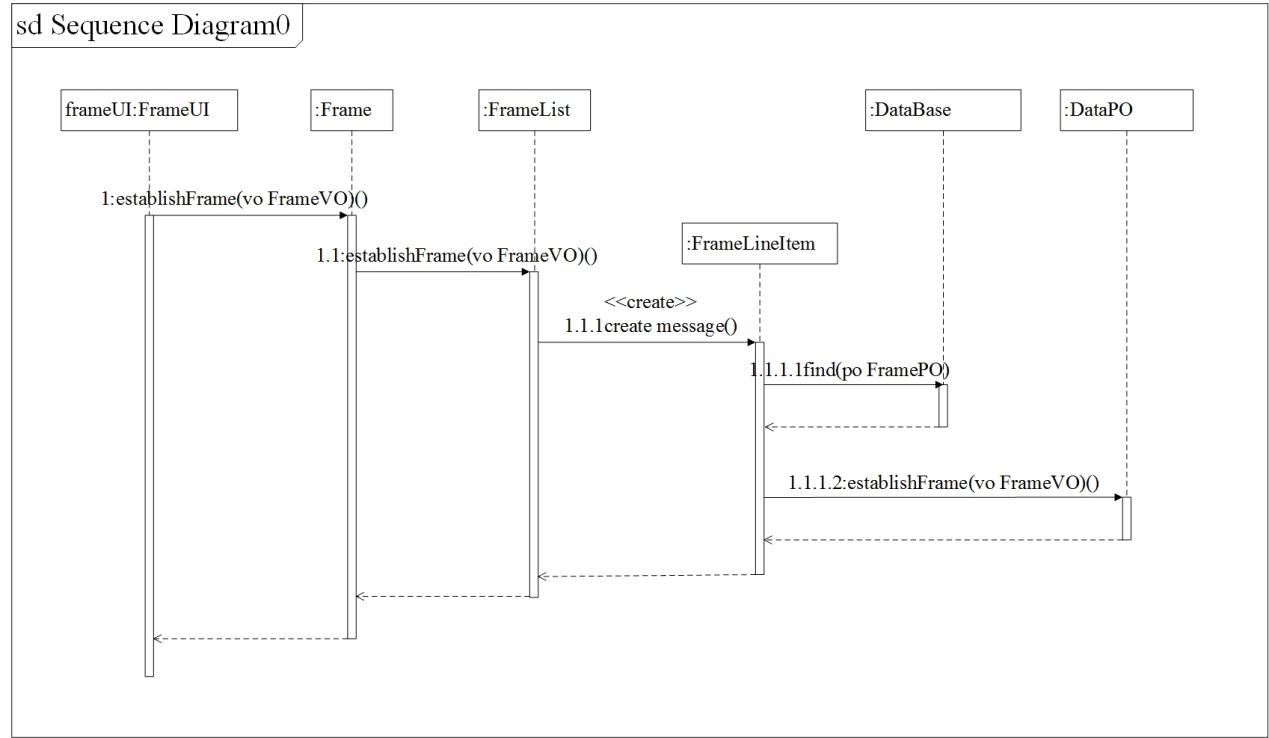
如图22所示，Info对象状态图描述了Info对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。showUserInfo方法被 UI调用，Info进入UserInfo状态；showPlanInfo方法被 UI调用，Info进入PlanInfo状态；showCourseInfo方法被 UI调用，Info进入CourseInfo状态；showFrameInfo方法被 UI调用，Info进入FrameInfo状态；showCourseSelectionInfo方法被 UI调用，Info进入CourseSelectionInfo状态。

****

**图22 info模块的状态图**

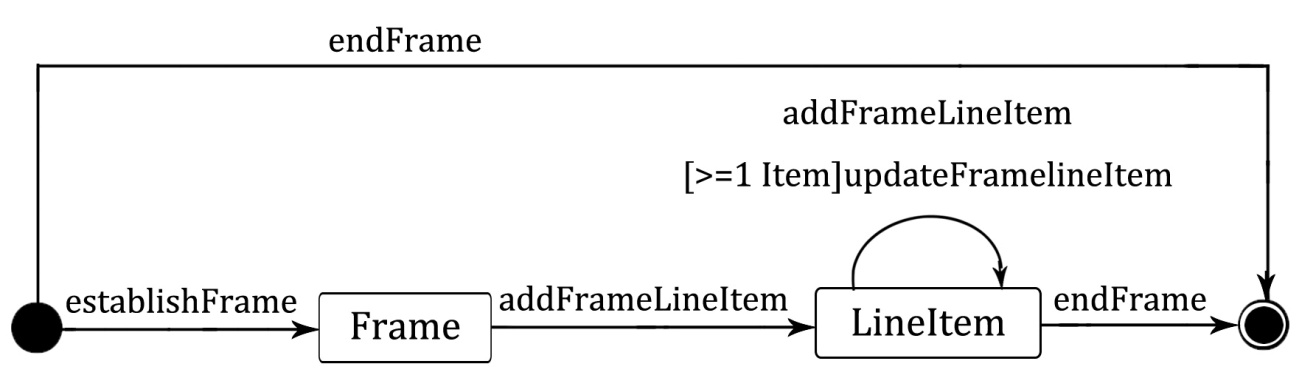
**4.2.3.4 framebl模块**

如图23所示的顺序图表明了在选课系统中，当教务处老师新建框架策略后，框架策略业务逻辑处理的相关对象之间的协作。

****

**图23 frame模块新建框架策略的顺序图**

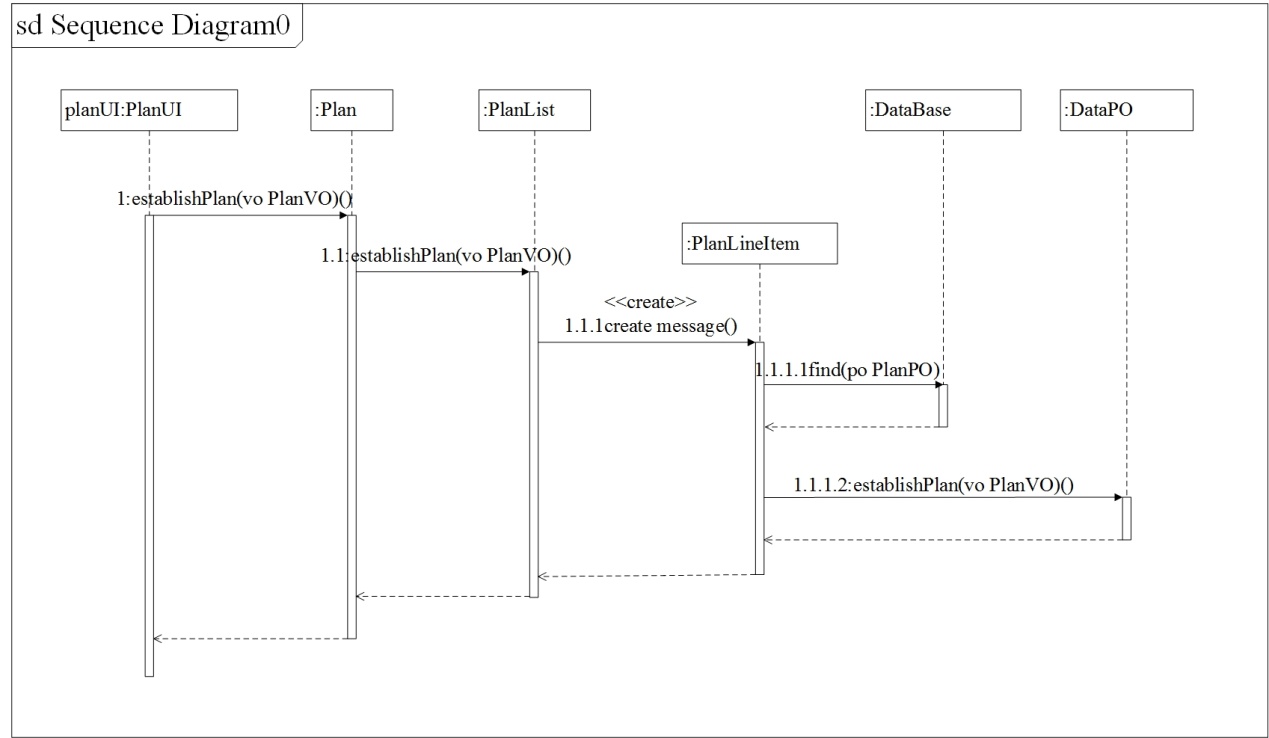
如图24所示，Frame对象状态图描述了Frame对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。随着eatablishFrame方法被 UI调用，Frame进入Frame状态；之后通过添加界面进入LineItem状态。



**图24 framebl模块新建框架策略的状态图**

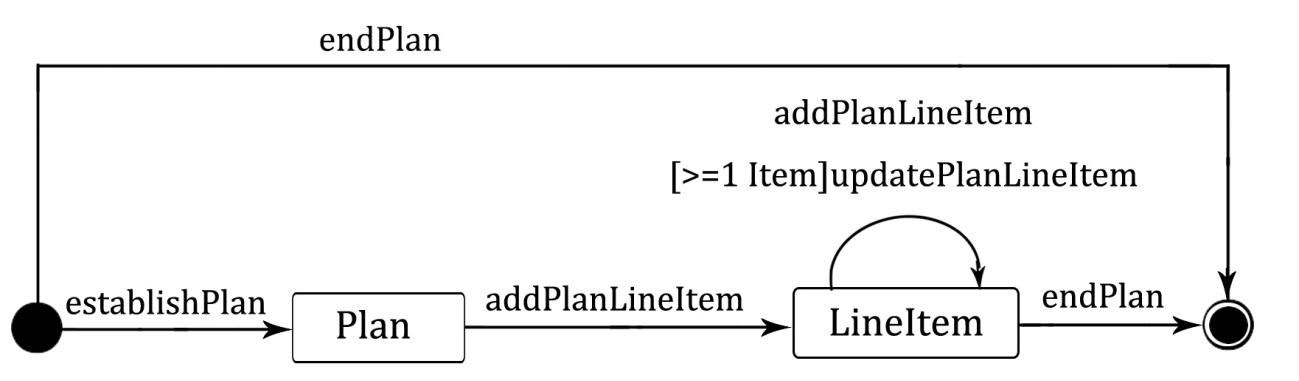
**4.2.3.5 planbl模块**

如图25所示的顺序图表明了在选课系统中，当院系教务老师新建教学计划时，教学计划业务逻辑处理的相关对象之间的协作。

****

**图25 planbl模块新建教学计划顺序图**

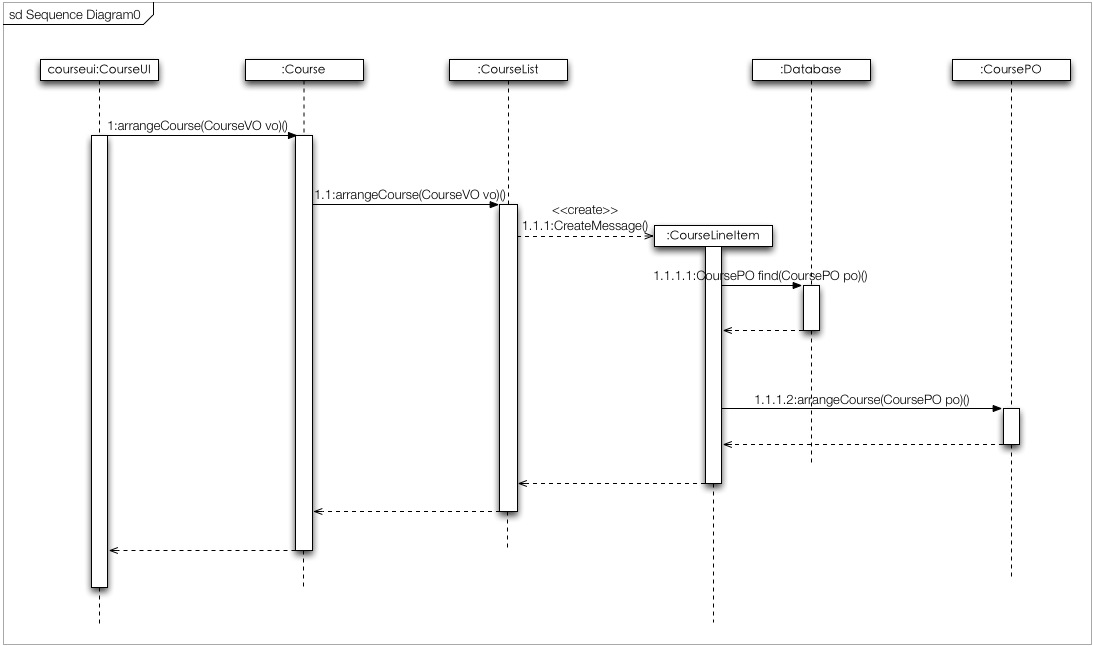
如图26所示，Plan对象状态图描述了Plan对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。随着establishPlan方法被 UI调用，Plan进入Plan状态；之后通过添加计划进入LineItem状态。



**图26 planbl模块新建教学计划的状态图**

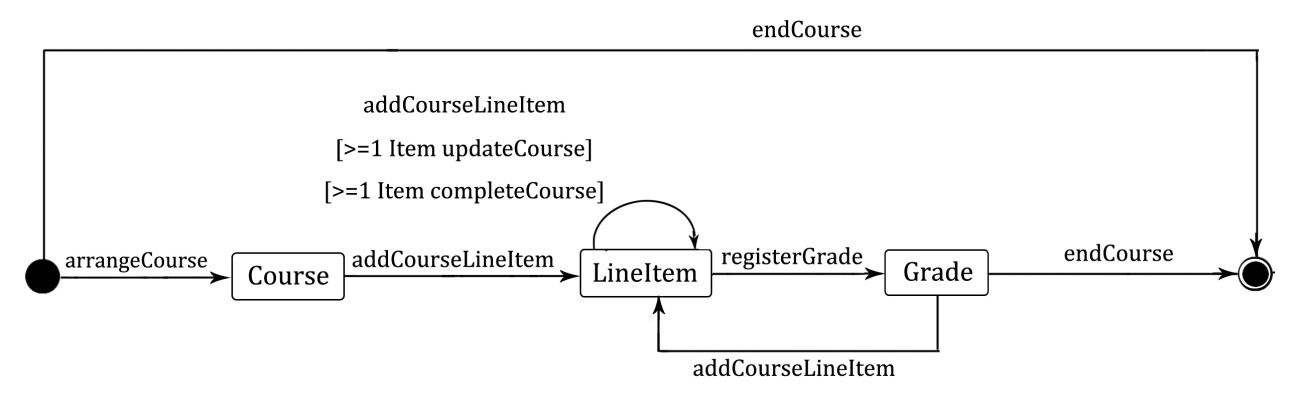
**4.2.3.6 coursebl模块**

如图27所示的顺序图表明了在选课系统中，当院系教务老师发布课程后，course业务逻辑处理的相关对象之间的协作。

****

**图27 coursebl模块安排课程顺序图**

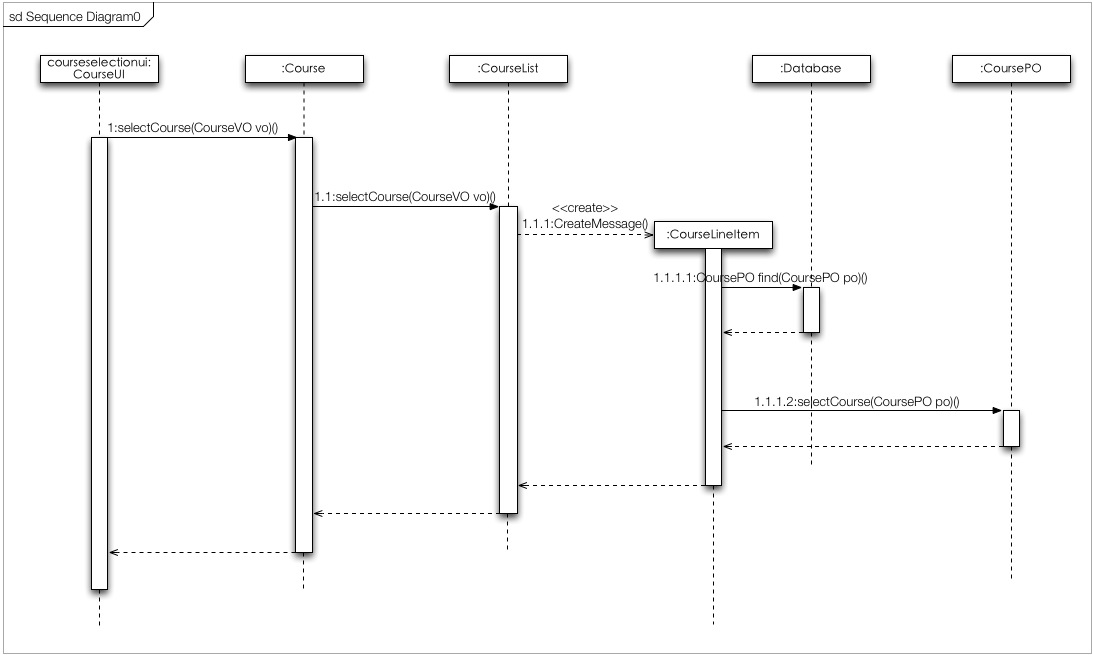
如图28所示，Course对象状态图描述了Course对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。随着arrangeCourse方法被 UI调用，Course进入Course状态；之后通过添加课程进入LineItem状态。随后，随着selectCourse方法被UI调用，Course进入CourseSelection状态；随着registerGrade方法被UI调用，Course进入Grade状态。

****

**图28 coursebl模块安排课程状态图**

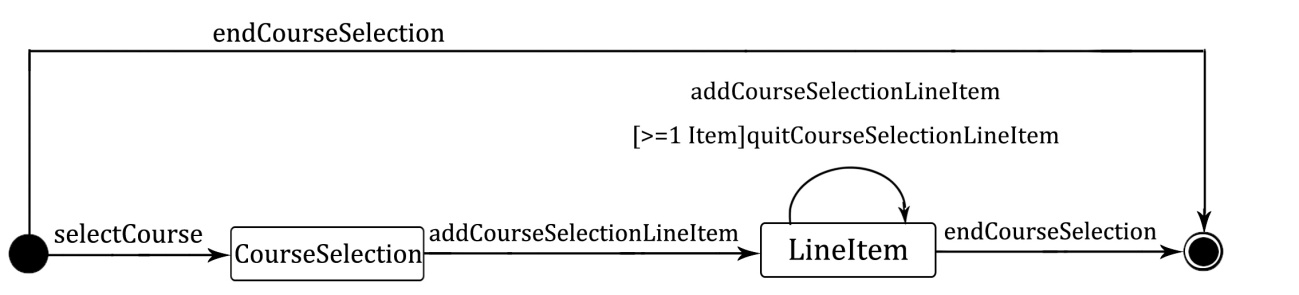
**4.2.3.7 courseselectionbl模块**

如图29所示的顺序图表明了在选课系统中，当学生选课后，选课业务逻辑处理的相关对象之间的协作。

****

**图29 courseselectionbl模块选课顺序图**

如图30所示，CourseSelection对象状态图描述了CourseSelection对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。随着selectCourse方法被 UI调用，CourseSelection进入CourseSelection状态；之后通过添加选课记录进入LineItem状态。



**图30 courseselectionbl模块选课状态图**

**4.2.4 客户端业务逻辑层模块的设计原理**

利用委托式控制风格，每个模块需要访问的业务逻辑有各自的控制器委托给不同的领域对象。这些对象分别承担一定的职责，做出一定的决策，从而实现大的职责。

**4.3客户端网络模块的静态结构和动态职责**

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31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31客户端数据通信层模块的描述**

**4.3.1客户端网络模块局部模块的职责**

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20所示。

**表20 网络通信层模块的职责**

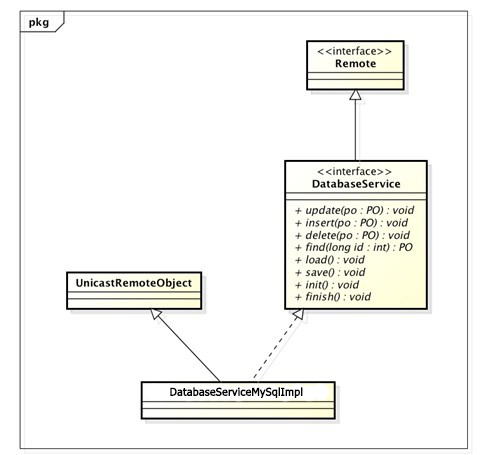
|  |  |
| --- | --- |
| **模块** | **职责** |
| IDataRemoteService\_Stub | 给业务逻辑层提供服务接口 |
| Java RMI 类库 | 实现RMI机制 |

**4.4服务器端网络模块的静态结构和动态职责**

**4.4.1服务器端网络模块局部模块的职责**

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32 服务器数据层实现RMI的类的描述**

**4.4.2服务器端网络模块局部模块的接口规范**

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21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 |

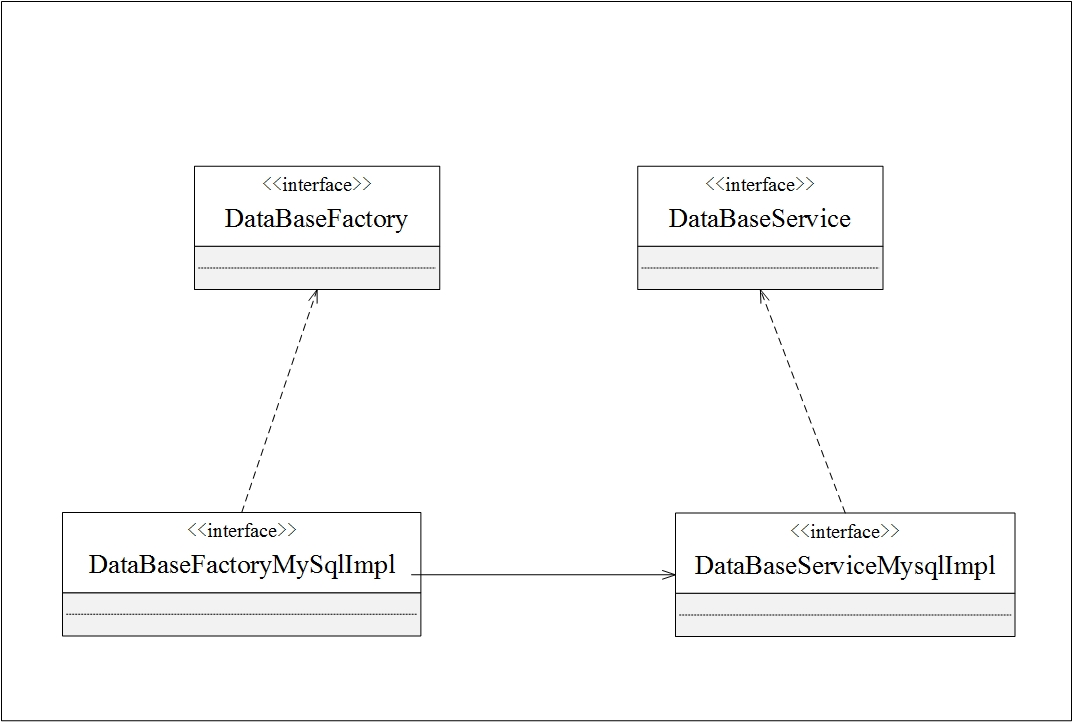
**表22服务器网络模块的RMI接口返回值的规范**

|  |  |
| --- | --- |
| **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 |

**4.5服务器端数据层模块的静态结构和动态职责**

**4.5.1服务器端数据层模块局部模块的职责**

服务器端数据层主要负责数据的持久化及数据访问接口。对于持久化数据的集体载入、集体保存和增、删、改、查。这些服务由DataService接口提供。本系统（高校选课系统）程序的持久化数据的保存以数据库的形式实现。具体如图33所示。



**图33 数据层模块的描述**

如表20所示为数据层模块的职责。

**表23 数据层模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| DatabaseFactory | 持久化数据工厂，提供获得不同数据的工厂方法 |
| DatabaseService | 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。 |
| DatabaseFactoryMySqllmpl | 基于MySql数据库的持久化数据库工厂的实现 |
| DatabaseServiceMySqlImpl | 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。 |

**4.5.2服务器端数据层模块局部模块的接口规范**

服务器端数据层模块的接口规范如表24所示。

**表21 数据层模块的接口规范**

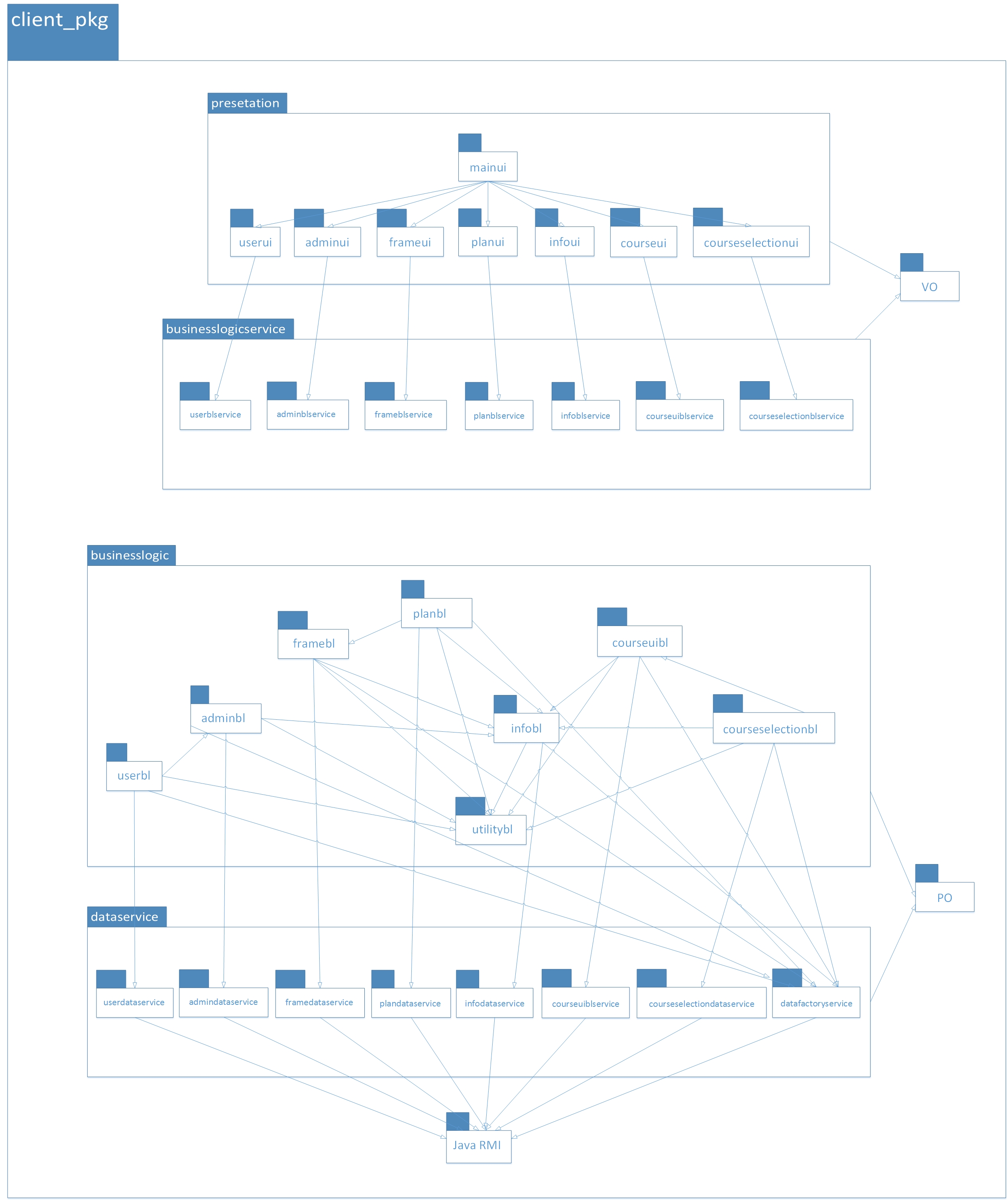
|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| DatabaseFactory.getUserDatabase | 语法 | Public DatabaseService getUserDatabase()throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 得到User数据库对象 |
| UserDataService.find | 语法 | public UserPO find(String ID) throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的UserPO结果 |
| DatabaseFactory.getInfoDatabase | 语法 | Public DatabaseService getInfoDatabase()throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 得到Info数据库对象 |
| InfoDataService.findCourse | 语法 | Public CoursePO findCourse(String userID,String courseID)throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 返回CoursePO信息 |
| InfoDataService.findCourseSelection | 语法 | Public CourseSelectionPO findCourseSelectionInfo(String ueserID)throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 返回CourseSelectionPO信息 |
| InfoDataService.findPlan | 语法 | Public PlanPO findPlan(String institute)throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 返回Plan信息 |
| InfoDataService.find | 语法 | Public PO find()throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 返回PO信息 |
| InfoDataService.findUser | 语法 | Public UserPO findUser(String userID)throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 返回Frame信息 |
| DatabaseFactory.getCourseDatabase | 语法 | Public DatabaseService getCourseDatabase()throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 得到Course数据库对象 |
| CourseDataService.insert | 语法 | public ResultMessage insert(CoursePO po) throws RemoteException; |
| 前置条件 | 同样ID 的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| CourseDataService.delete | 语法 | public ResultMessage delete(CoursePO po) throws RemoteException; |
| 前置条件 | 在Mapper中存在同样ID 的po |
| 后置条件 | 删除一个po |
| CourseDataService.update | 语法 | public ResultMessage update(CoursePO coursepo) throws RemoteException; |
| 前置条件 | 在Mapper中存在同样ID 的po |
| 后置条件 | 更新一个po |
| DatabaseFactory.getPlanDatabase | 语法 | Public DatabaseService getPlanDatabase()throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 得到Plan数据库对象 |
| PlanDataService.establish | 语法 | public ResultMessage establish(PlanPO planPO) throws RemoteException; |
| 前置条件 | 同样ID 的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| PlanDataService.update | 语法 | public ResultMessage update(PlanPO planPO) throws RemoteException; |
| 前置条件 | 在Mapper中存在同样ID 的po |
| 后置条件 | 更新一个po |
| DatabaseFactory.getAdminDatabase | 语法 | Public DatabaseService getAdminDatabase()throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 得到Admin数据库对象 |
| AdminDataService.insert | 语法 | public ResultMessage insert(UserPO userPO) throws RemoteException; |
| 前置条件 | 同样ID 的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| AdminDataService.delete | 语法 | public ResultMessage delete(UserPO userPO) throws RemoteException; |
| 前置条件 | 在Mapper中存在同样ID 的po |
| 后置条件 | 删除一个po |
| AdminDataService.update | 语法 | public ResultMessage update(UserPO userPO) throws RemoteException; |
| 前置条件 | 在Mapper中存在同样ID 的po |
| 后置条件 | 更新一个po |
| DatabaseFactory.getFrameDatabase | 语法 | Public DatabaseService gettFrameDatabase()throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 得到Frame数据库对象 |
| FrameDataService.establish | 语法 | public ResultMessage establish(FramePO framePO) throws RemoteException; |
| 前置条件 | 同样ID 的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| FrameDataService.update | 语法 | public ResultMessage update(FramePO framePO) throws RemoteException; |
| 前置条件 | 在Mapper中存在同样ID 的po |
| 后置条件 | 更新一个po |
| DatabaseFactory.getCourseSeDatabase | 语法 | Public DatabaseService getCourseSeDatabase ()throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 得到CourseSeletion数据库对象 |
| CourseSeletionDataService.insert | 语法 | public ResultMessage insert(CourseSeletionPO po) throws RemoteException; |
| 前置条件 | 同样ID 的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| CourseSeletionDataService.update | 语法 | public ResultMessage update(CourseSeletionPO po) throws RemoteException; |
| 前置条件 | 在Mapper中存在同样ID 的po |
|  | 更新一个po |
| CourseSeletionDataService.delete | 语法 | public ResultMessage delete(CourseSeletionPO po) throws RemoteException; |
| 前置条件 | 在Mapper中存在同样ID 的po |
| 后置条件 | 删除一个po |
| DataService.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 |
| DataService.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。 |

**4.5.3 服务器端数据层模块的设计原理**

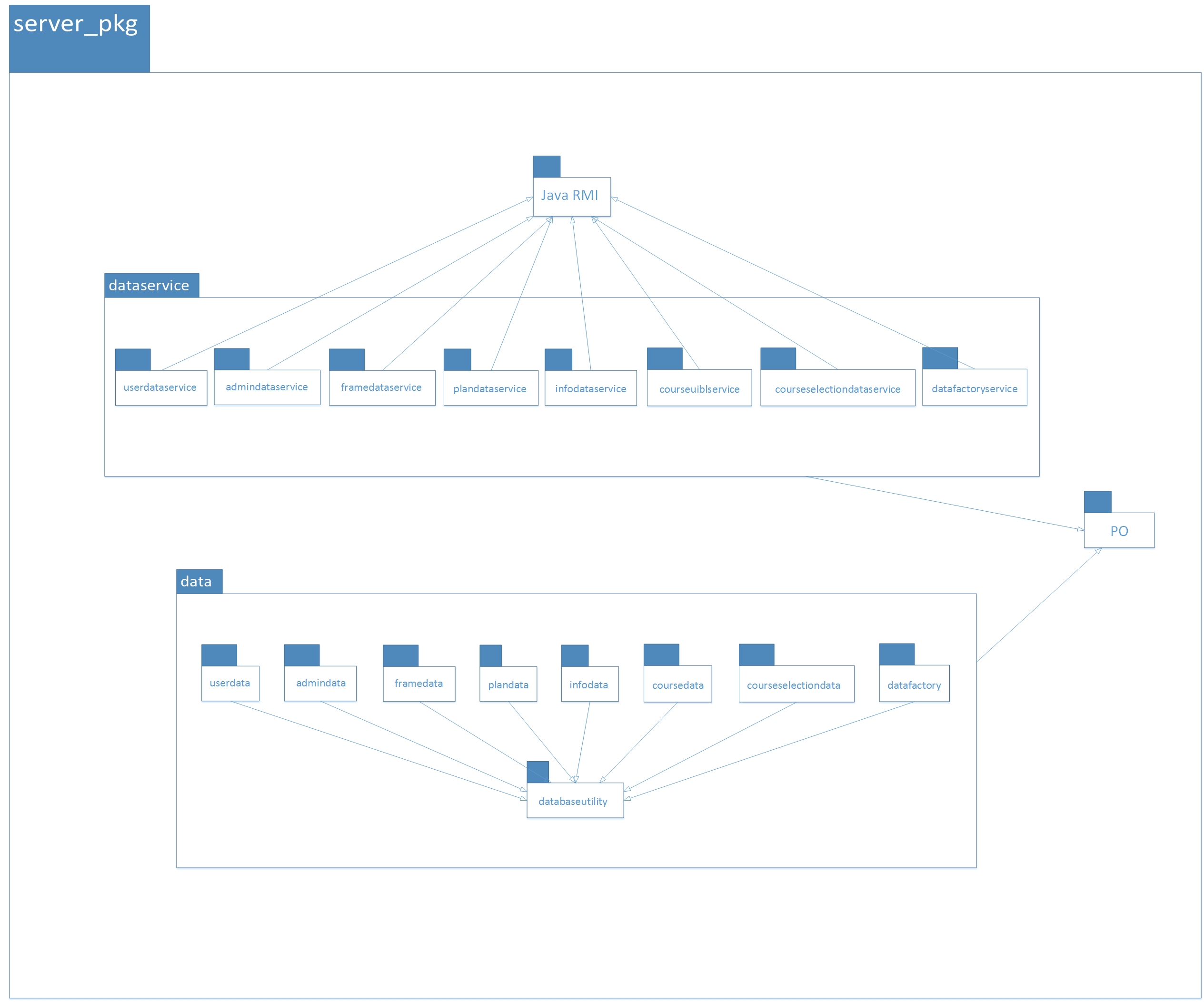
服务器端数据层模块采用了抽象工厂的模式，这种模式对数据库的实现提供了很好的可修改性。

**5、依赖关系**

图34和图35是客户端和服务器端各自的包之间的依赖关系。



**图34 客户端开发包图**

****

**图35 服务器端开发包图**