南京大学教务系统软件详细设计描述文档

南京大学软件学院

2013 年 10 月

韩旭（121250041）
郭天昊（121250039）
陈俊杉（121250011）
施周利（121250126）

目录

1 引言.....2
1.1 编制目的.....2
1.2 词汇表.....2
1.3 参考资料.....2
2 产品概述.....2
3 体系结构设计描述.....2
4 结构视角.....2
4.1 业务逻辑层的分解.....2
4.1.1 deanImpl 模块 2
(1) 模块描述.....2
(2) 整体结构.....2
(3) 模块内部类的接口规范.....3
(4) 业务逻辑层的动态模型.....5
(5) 业务逻辑层的设计原理.....12
4.1.2 facultyDeanImpl 模块 12
(1) 模块描述.....12
(2) 整体结构.....12
(3) 模块内部类的接口规范.....13
(4) 业务逻辑层的动态模型.....14
(5) 业务逻辑层的设计原理.....20
4.1.3 teacherImpl 模块 20
(1) 模块描述.....20
(2) 整体结构.....20
(3) 模块内部类的接口规范.....21
(4) 业务逻辑层的动态模型.....23
(5) 业务逻辑层的设计原理.....27
4.1.4 studentImpl 模块 27
(1) 模块描述.....27
(2) 整体结构.....27
(3) 模块内部类的接口规范.....28
(4) 业务逻辑层的动态模型.....29
(5) 业务逻辑层的设计原理.....35
5 附录.....35

1 引言

1.1 编制目的

本报告详细完成对教务系统的概要设计，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。

1.2 词汇表

1.3 参考资料

软件开发的技术基础

2 产品概述

参考教务系统用例文档和教务系统需求规格说明文档中对产品的概括描述。教务系统体系结构采用分层风格。

3 体系结构设计描述

参考选课系统体系结构描述文档对体系结构设计的概述。

4 结构视角

4.1 业务逻辑层的分解

业务逻辑层的开发包图参见软件体系结构文档图 4。

4.1.1 deanImpl 模块

(1) 模块描述

deanImpl 模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

deanImpl 模块的职责与接口参见软件体系结构描述文档表 9。

(2) 整体结构

根据体系结构设计，我们将系统分为展示层（presentation）、业务逻辑层（logic）、数据层（data）。每一层之间为了增加灵活性，我们会添加接口。展示层和业务逻辑层之间添加 com.logicService.DeanMethod 接口。业务逻辑层和数据层之间添加 com.dataService.DeanDatabaseMethod 接口。为了隔离业务逻辑职责和逻辑控制职责，我们增加了 DeanMethodController,这样 DeanMethodController 会将对教务员业务的业务处理逻辑委托给 DeanMethodImpl 对象。Dean,Course,Teacher,BasicFrame,Frame 是作为教务员的持久化对象被添加到设计模型中去的。DeanMethodImpl 采用策略模式，将方法分解到 CourseGetter,CoursePublish,CourseUpadta,TeacherGetter,DeanGetter,BasicFrameManagement,FrameManagement,Login,PasswordChange 中去实现，提高了代码的复用性和易修改性。

deanImpl 模块的设计如图 1 所示。

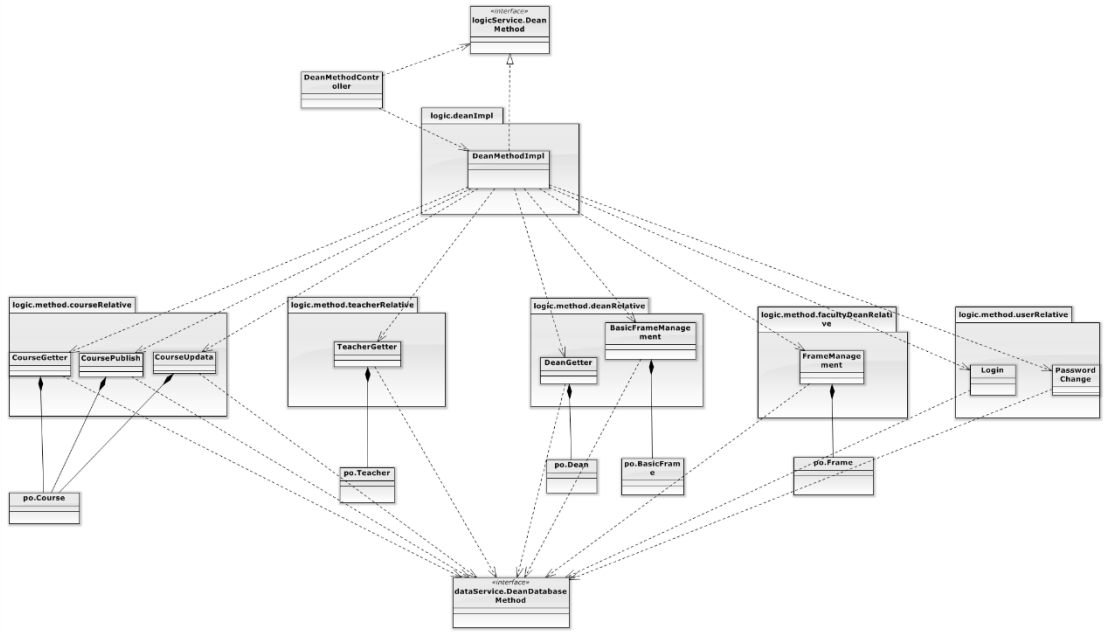

图 1 deanImpl 模块各个类的设计

deanImpl 模块各个类的职责如表 1 所示。

表 1 deanImpl 模块各个类的职责

模块	职责
DeanMethodController	负责实现教务员所需要服务于 DeanMethodImpl 的解耦
DeanMethod	教务员所旭服务的方法接口
DeanMethodImpl	教务员所需服务具体方法的实现
DeanDatabaseMethod	提供数据库服务方法的接口

（剩余相关类的职责见表 13）

(3) 模块内部类的接口规范

DeanMethodController 的接口规范如表 2 所示。

表 2DeanMethodController 的接口规范

提供的服务		
DeanMethodController. getDeanMethod	语法	public static DeanMethod getDeanMethod()
	前置条件	无
	后置条件	返回一个已 DeanMethod 为接口的对象

DeanMethodImpl 的接口规范如表 3 所示。

表 3 DeanMethodImpl 的接口规范

提供的服务（供接口）		
Dean.login	语法	public interface DeanMethod extends Remote
	前置条件	用户输入有效的账号和密码。
	后置条件	根据账号查找是否存在相应的用户，根据账号验证密码是否正确，如果通过验证，则登录成功，返回 True 否

		则，返回 False
Dean.getSelf	语法	public Dean getSelf(String ID) throws RemoteException;
	前置条件	学校教务员已经成功登录
	后置条件	返回该院系教务员信息
Dean. changePassword	语法	public boolean changePassword(String ID, String originalPassword,String password)throws RemoteException;
	前置条件	学校教务员已经登录。且用户输入有效的账号，原密码和现密码。
	后置条件	院系教务员修改密码。根据账号查找是否存在相应的用户，根据账号验证旧密码是否正确，如果通过验证，则将密码修改为现密码，且返回 True，否则，返回 False
Dean. formulateFrame	语法	public boolean formulateFrame(BasicFrame bf)throws RemoteException;
	前置条件	学校教务员已经登录，且用户根据内容进行了有效的输入
	后置条件	建立整体框架，如果创建成功，则返回 True，否则返回 False
Dean. modifyFrame	语法	public boolean modifyFrame(BasicFrame bf)throws RemoteException;
	前置条件	学校教务员已经登录，且用户根据内容进行了有效的输入
	后置条件	学校教务员修改整体框架，如果修改成功，则返回 True，否则返回 False
Dean. lookUpBasicFrame	语法	public BasicFrame lookUpBasicFrame()throws RemoteException;
	前置条件	学校教务员已经登录
	后置条件	学校教务员请求查看整体框架，如果整体框架已创建，则返回该框架。反之，则系统提示错误
Dean.lookUpFrame	语法	public Frame lookUpFrame(String facultyID)throws RemoteException;
	前置条件	学校教务员已经登录，且输入了有效的院系 ID
	后置条件	教务员查看院系教学计划，根据 ID 找到对应的院系，并返回该院系计划，否则系统提示 ID 输入有误
Dean.getAllTeacher	语法	public List<Teacher> getAllTeacher()throws RemoteException;
	前置条件	学校教务员已经成功登录
	后置条件	学校教务员查看全校老师列表，返回一个装有信息的列表
Dean.getFacultyTeacher	语法	public List<Teacher> getFacultyTeacher(String facultyID)throws RemoteException;
	前置条件	学校教务员已经登录，且输入了有效的院系 ID
	后置条件	学校教务员查看院系所有老师，根据 ID 找到对应的院

		系，并返回该院系所有老师列表，否则系统提示 ID 输入有误
Dean.getAllCourse	语法	public List<Course> getAllCourse()throws RemoteException;
	前置条件	学校教务员已经成功登录
	后置条件	学校教务员查看全校课程列表，返回一个装有信息的列表
Dean.getFacultyCourse	语法	public List<Course> getFacultyCourse(String facultyID)throws RemoteException;
	前置条件	学校教务员已经登录，且输入了有效的院系 ID
	后置条件	学校教务员查看院系所有老师，根据 ID 找到对应的院系，并返回该院系所有老师列表，否则系统提示 ID 输入有误
Dean.publishCourse	语法	public boolean publishCourse(Course c)throws RemoteException;
	前置条件	学校教务员已登录，正确输入课程信息
	后置条件	发布课程成功，系统返回 true，否则提示发布失败，返回 false
Dean.modifyCourse	语法	public boolean modifyCourse(Course c)throws RemoteException;
	前置条件	学校教务员已登录，正确查找已发布课程
	后置条件	修改成功，系统返回 true，否则返回 false

需要的服务（需接口）

服务名	服务
DatabaseMethod.getConnection	得到数据库的服务引用
DatabaseMethod.search(String tableName,List clueName,List clue,List aimName)	在数据库中的数据进行搜索
DatabaseMethod.update(String tableName,String clueName,String clue,String aimName,String aim)	更新数据库中信息
DatabaseMethod.insert(String tableName,List<String> clueName,List clue)	向数据库中插入条目
DatabaseMethod.delete(String tableName,List<String> clueName,List<String> clue)	删除数据库中的条目

(4) 业务逻辑层的动态模型

图 2 表明了教务系统中，教务员输入 ID 和密码以登录的顺序图。

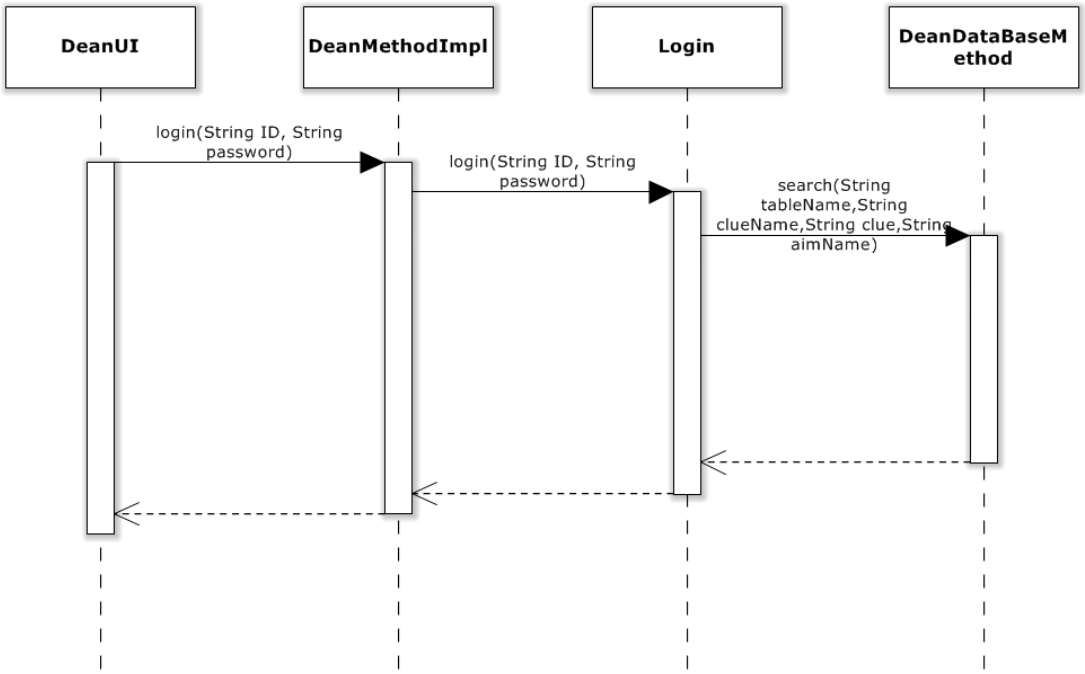

图 2

图 3 表明了教务系统中，教务员获得自身信息的顺序图。

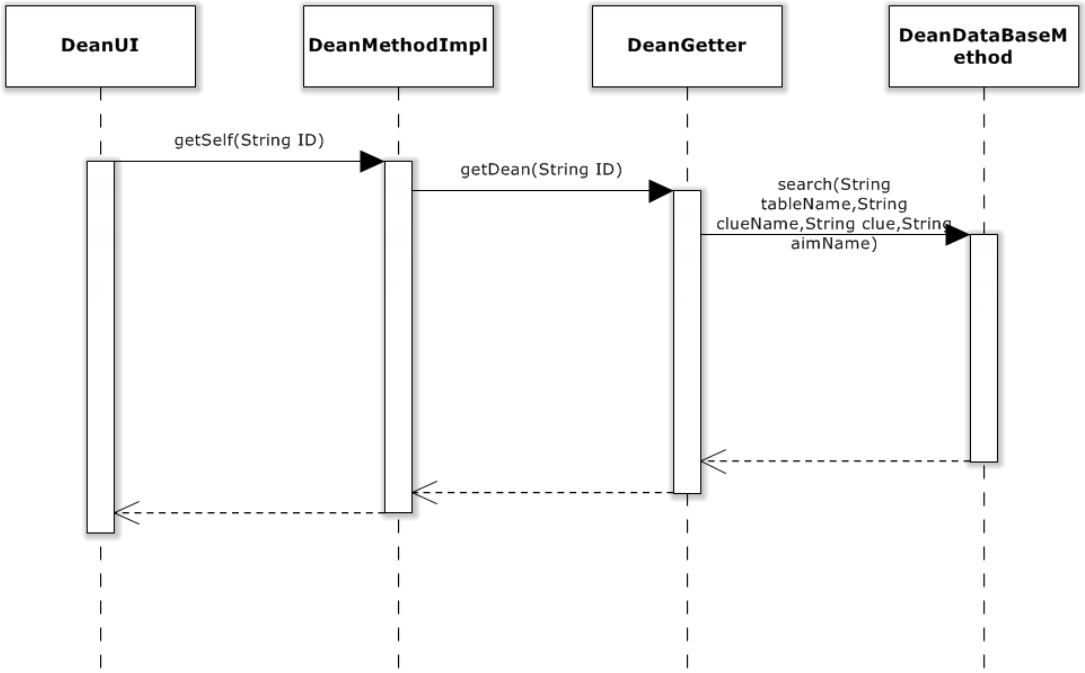

图 3

图 4 表明了教务系统中，教务员输入 ID、原密码和新密码以修改密码的顺序图。

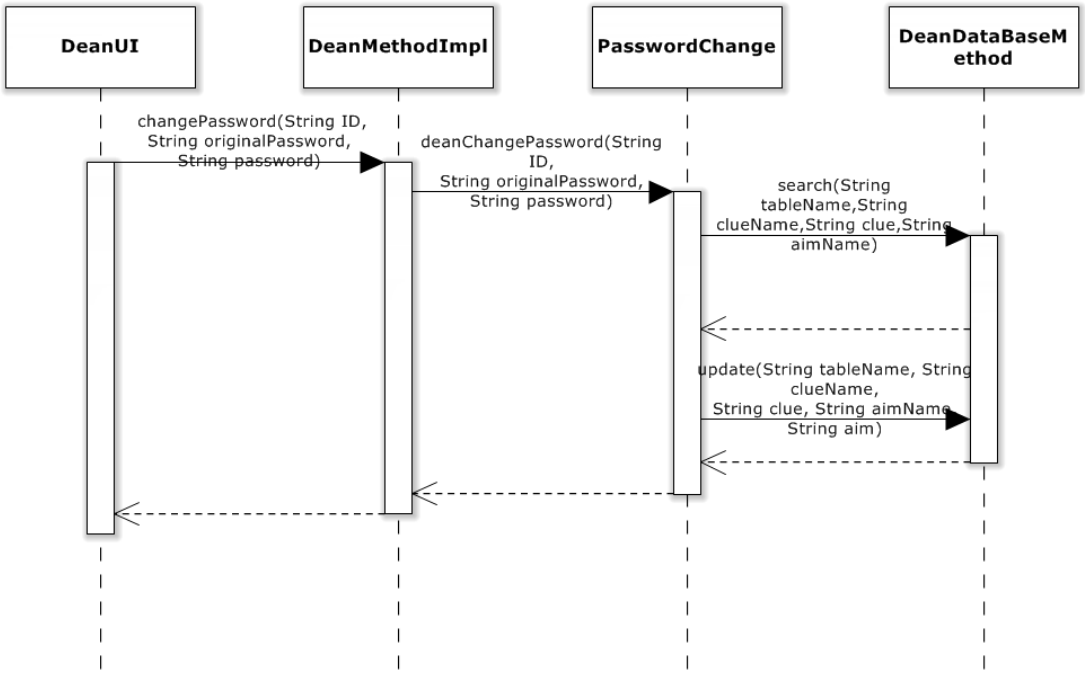

图 4

图 5 表明了教务系统中，教务员制定基本框架策略时各个对象之间的协作。

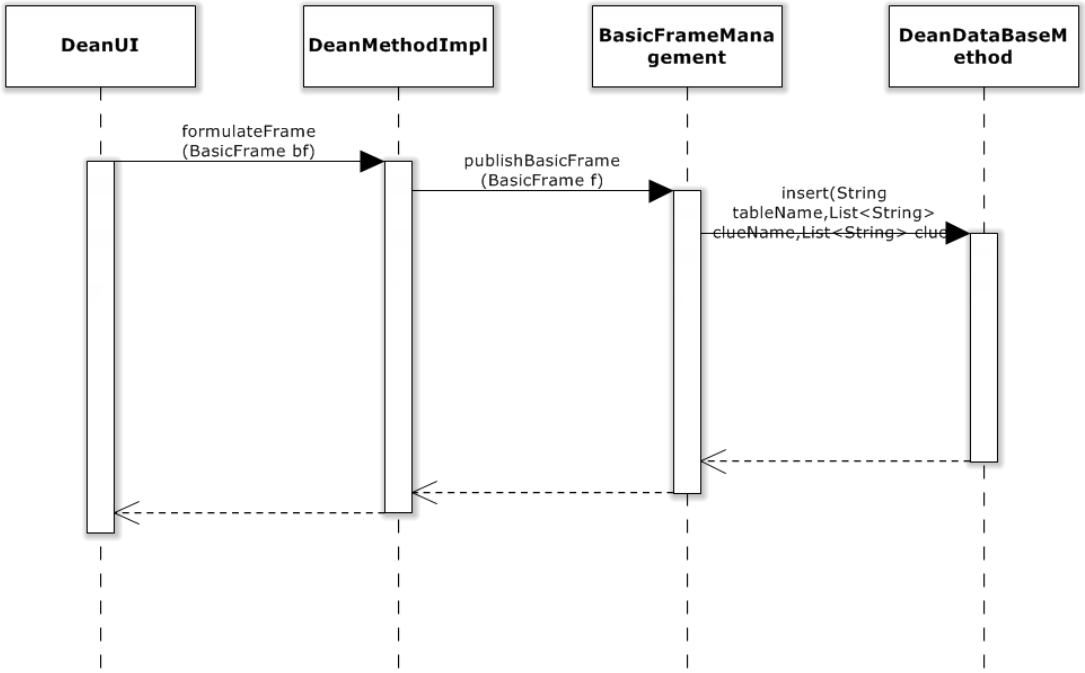

图 5

图 6 表明了教务系统中，教务员修改基本框架策略时各个对象之间的协作。

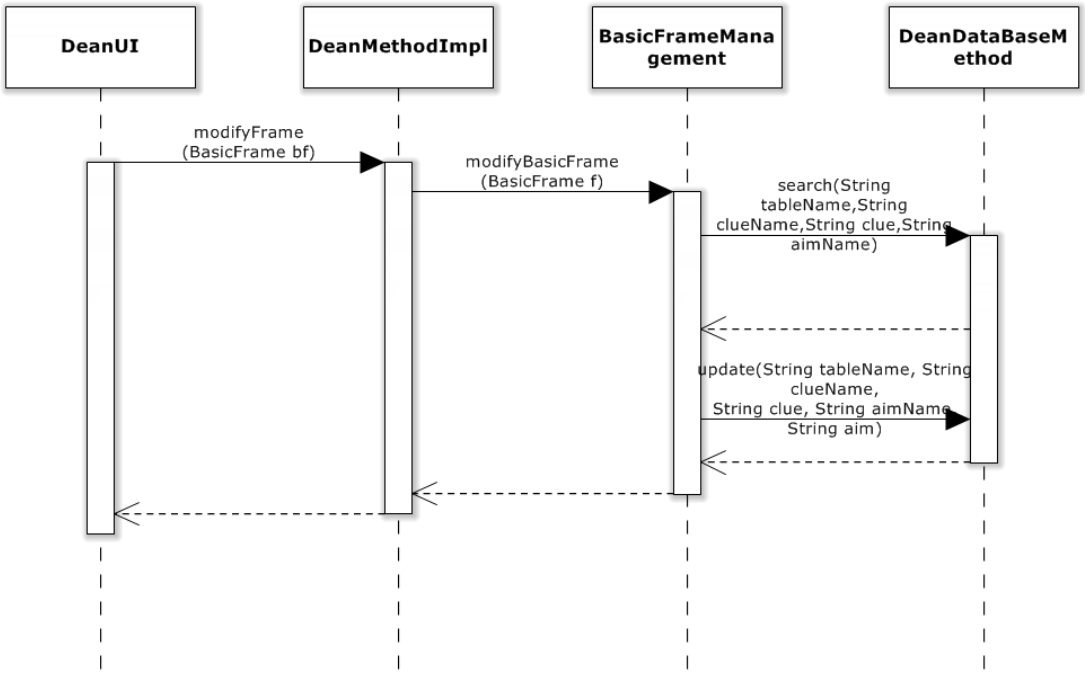

图 6

图 7 表明了教务系统中，教务员查看基本框架策略的顺序图。

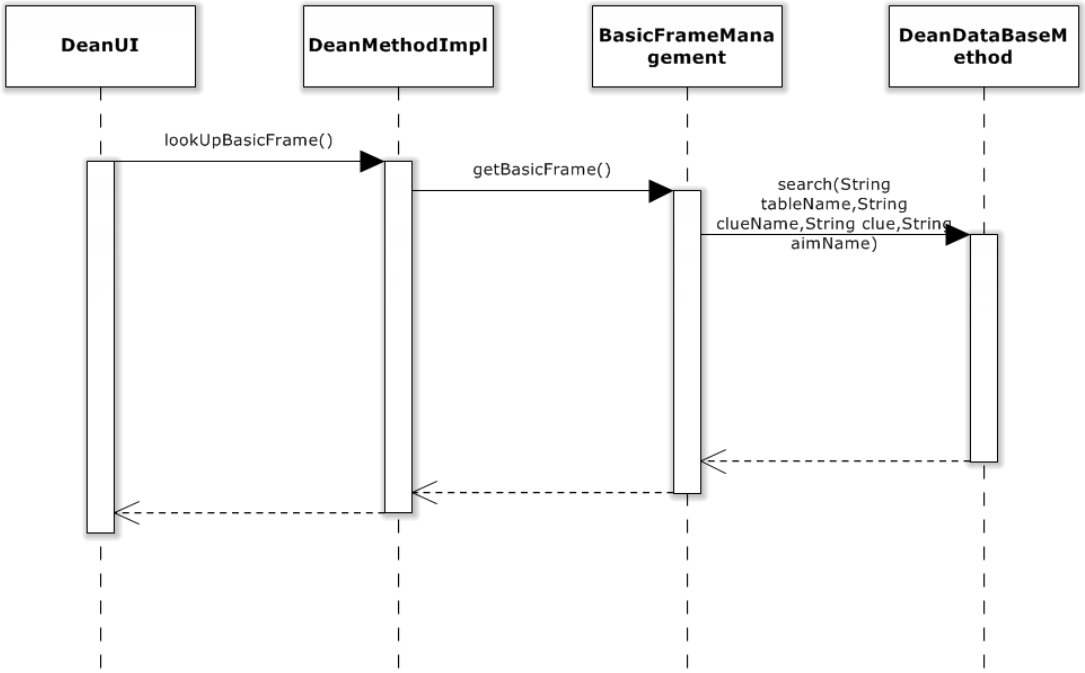

图 7

图 8 表明了教务系统中，教务员查看各院系教学计划的顺序图。

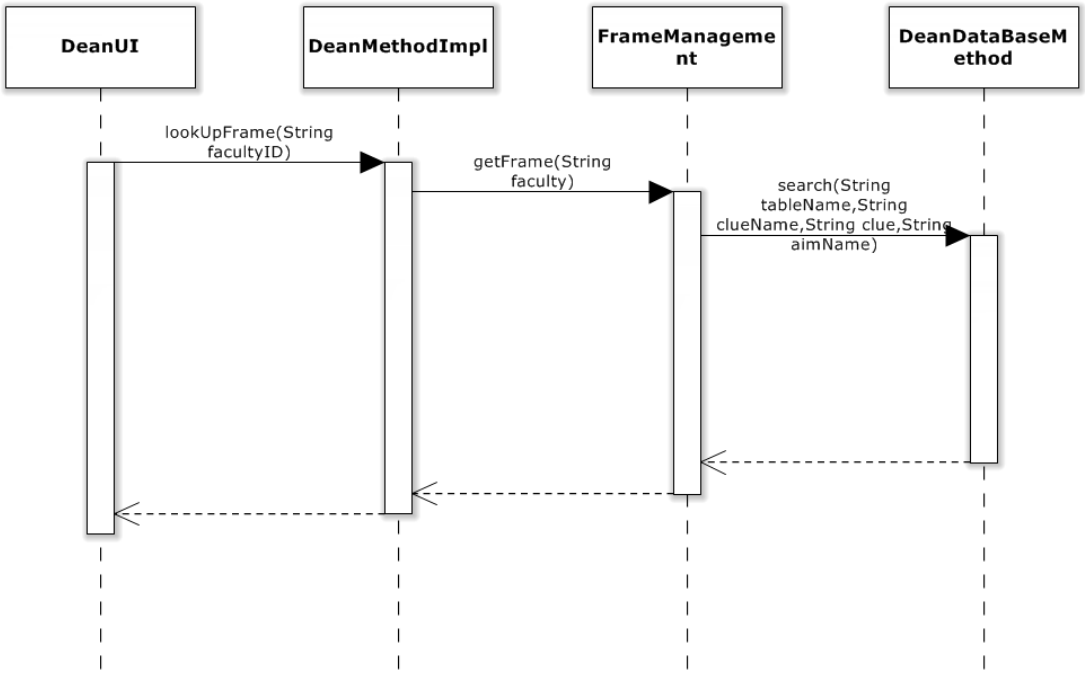

图 8

图 9 表明了教务系统中，教务员查看所有课程的顺序图。

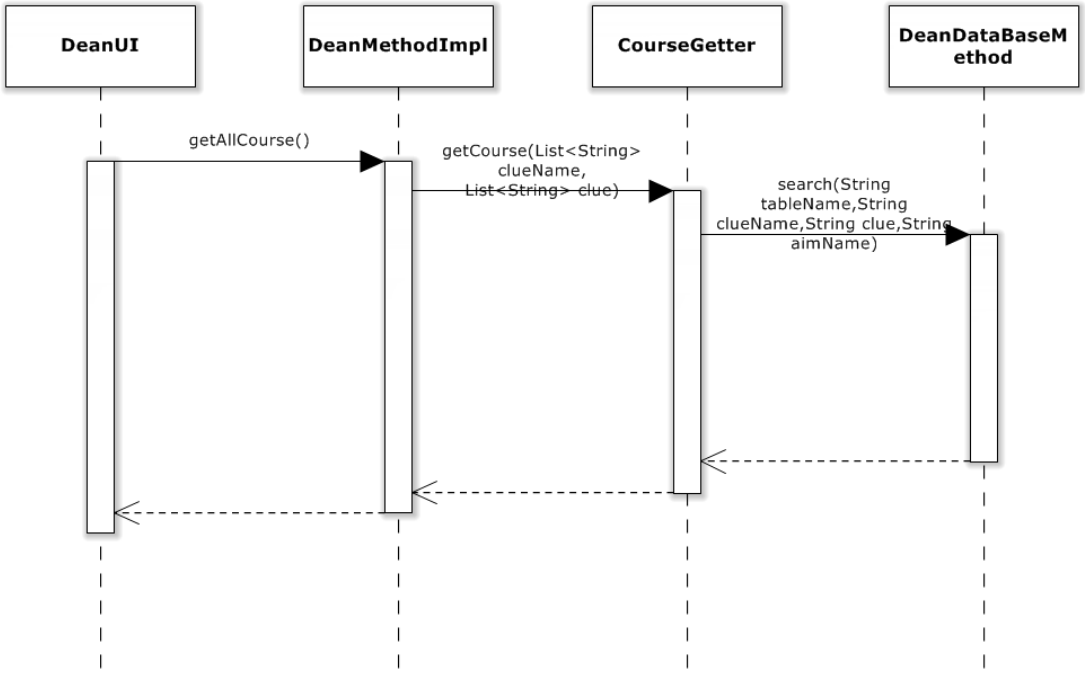

图 9

图 10 表明了教务系统中，教务员查看指定院系的课程的顺序图。

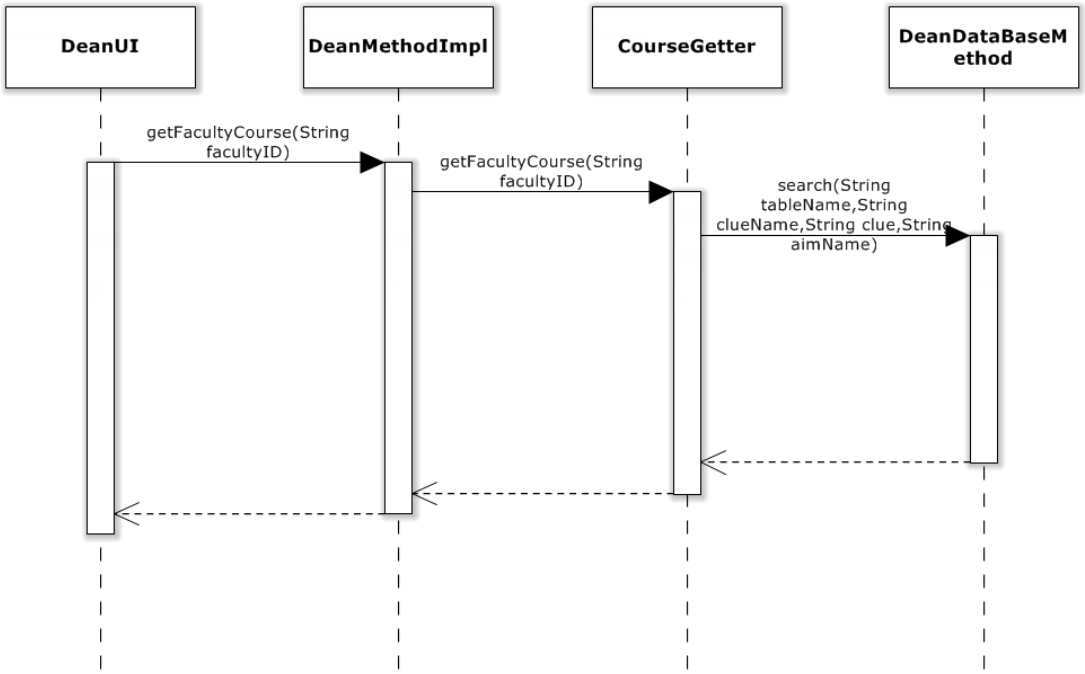

图 10

图 11 表明了教务系统中，教务员查看所有教师的顺序图。

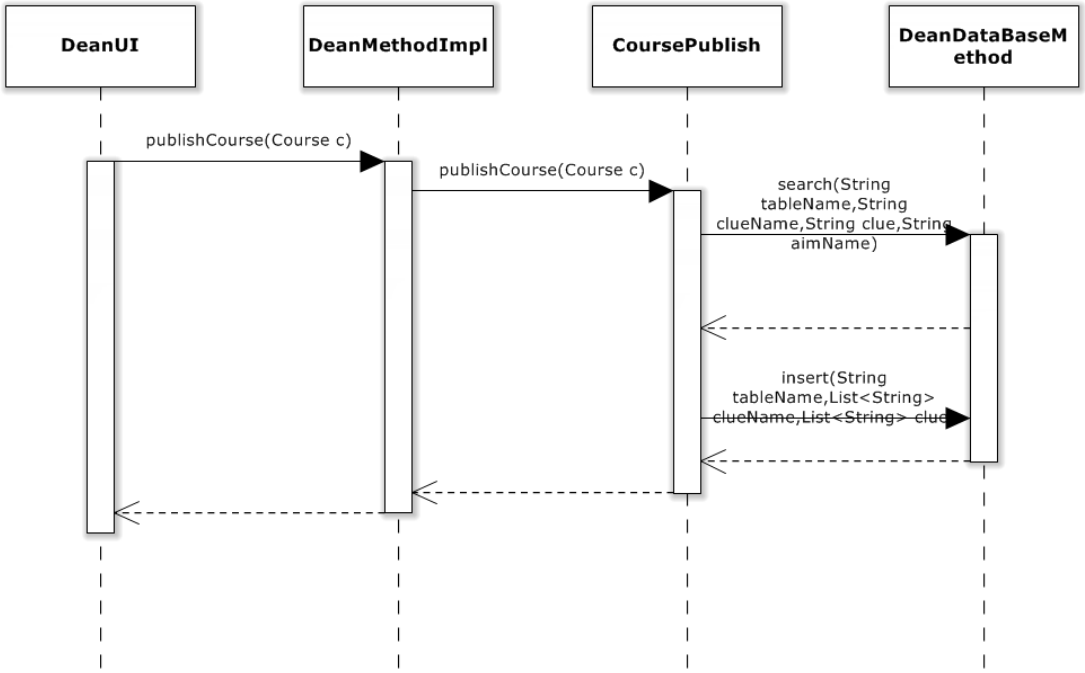

图 11

图 12 表明了教务系统中，教务员查看指定院系的教师的顺序图。

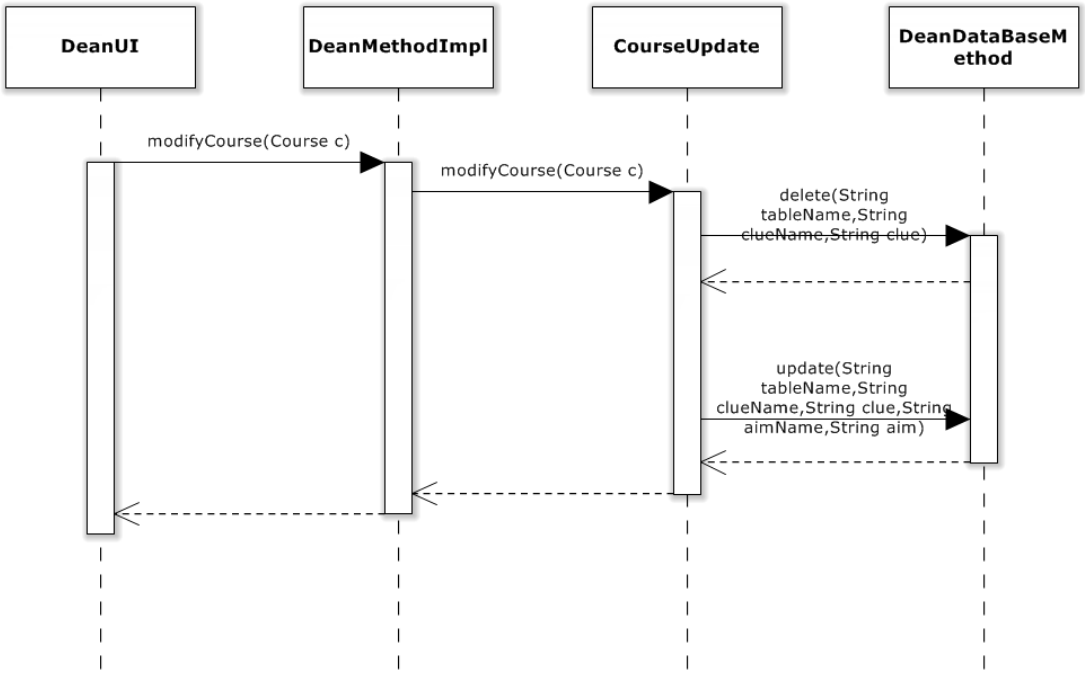

图 12

图 13 表明了教务系统中，教务员发布公共课程的顺序图。

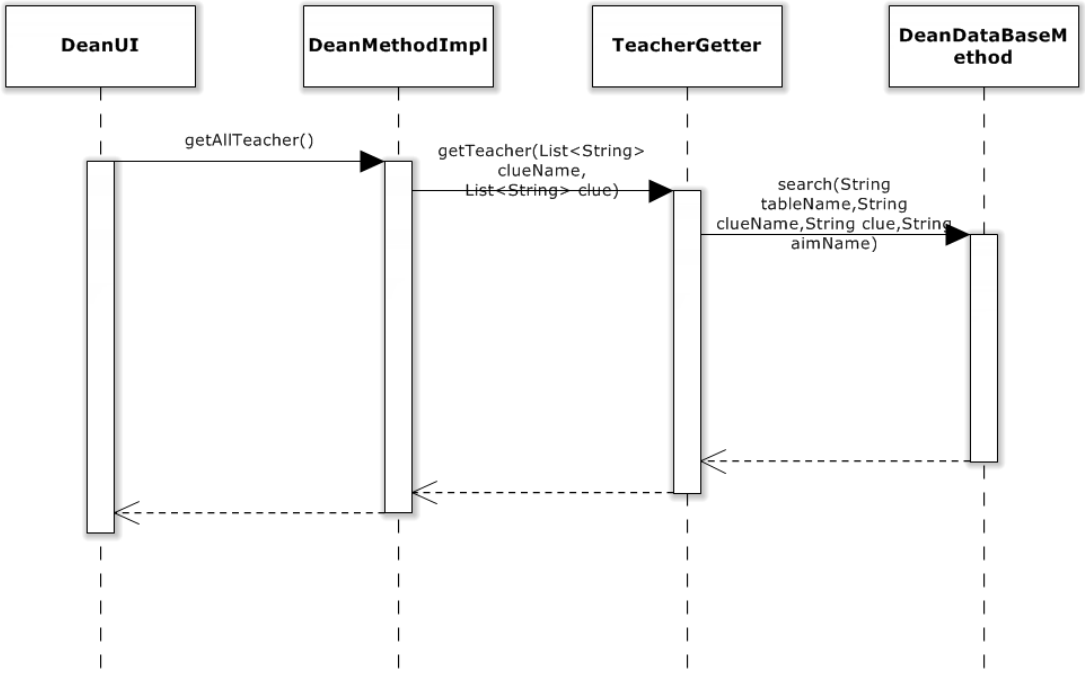

图 13

图 14 表明了教务系统中，教务员修改公共课程的顺序图。

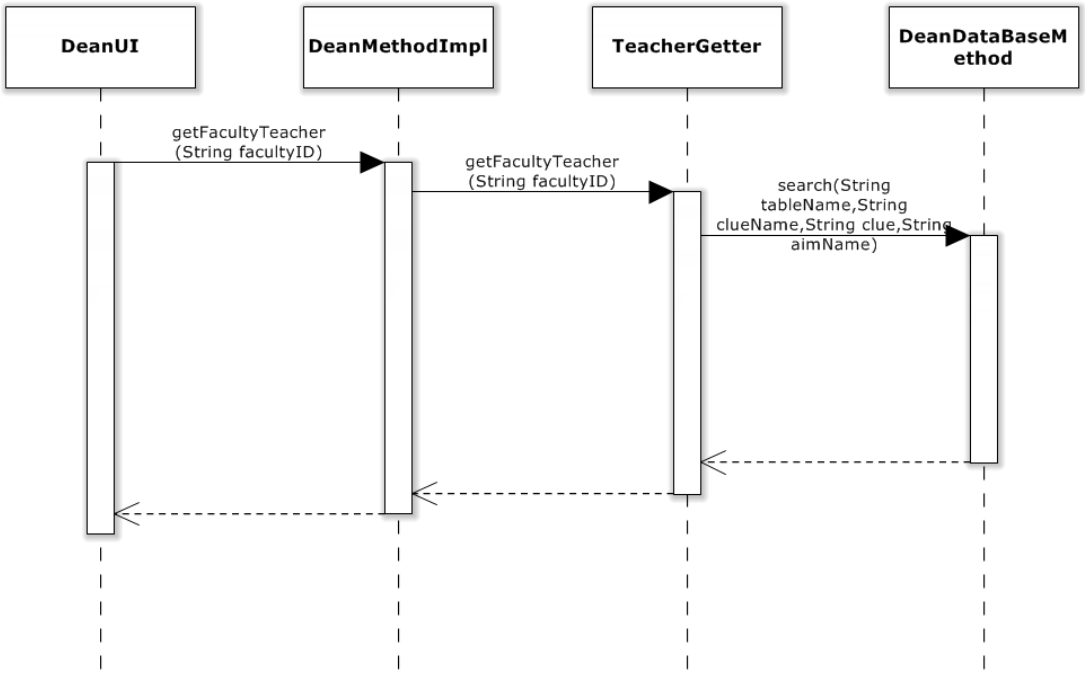

图 14

(5) 业务逻辑层的设计原理

利用单件模式，每个界面需要访问的业务逻辑由各自的控制器决定使用何种实现方法。

利用策略模式，DeanMethodImpl 所需实现的方法都委托到其他类来实现，提高了代码的复用性和易修改性。

4.1.2 facultyDeanImpl 模块

(1) 模块描述

facultyDeanImpl 模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

facultyDeanImpl 模块的职责与接口参见软件体系结构描述文档表 9。

(2) 整体结构

根据体系结构设计，我们将系统分为展示层（presentation）、业务逻辑层（logic）、数据层（data）。每一层之间为了增加灵活性，我们会添加接口。展示层和业务逻辑层之间添加 com.logicService.FacultyDeanMethod 接口。业务逻辑层和数据层之间添加 com.dataService.FacultyDeanDatabaseMethod 接口。为了隔离业务逻辑职责和逻辑控制职责，我们增加了 FacultyDeanMethodController,这样 FaucltyDeanMethodController 会将对院系教务员业务的业务处理逻辑委托给 FacultyDeanMethodImpl 对象。Dean, Course, Teacher, Course,FacultyDean,BasicFrame, Frame 是作为院系教务员的持久化对象被添加到设计模型中去的。FacultyDeanMethodImpl 采用策略模式，将方法分解到 CourseGetter,CoursePublish,CourseUpadta,TeacherGetter,FacultyDeanGetter,BasicFrameManagement,FrameManagement,Login,PasswordChange 中去实现，提高了代码的复用性和易修改性。

deanImpl 模块的设计如图 15 所示。

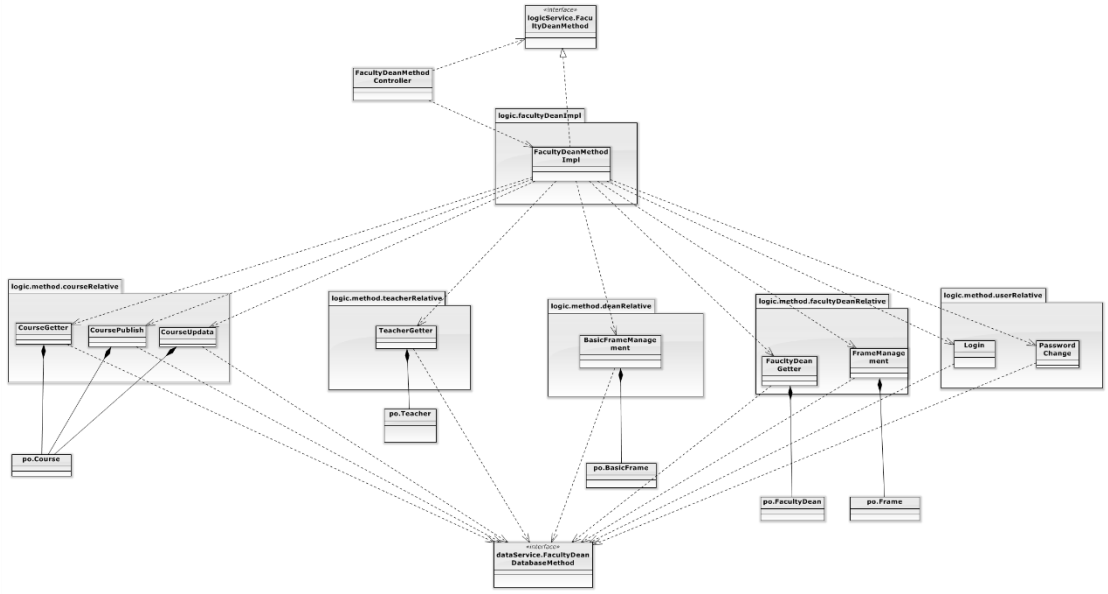

图 15 facultyDeanImpl 模块各个类的设计

facultyDeanImpl 模块各个类的职责如表 4 所示。

表 4 facyltyDeanImpl 模块各个类的职责

模块	职责
FacultyDeanMethodController	负责实现院系教务员所需要服务于 FacultyDeanMethodImpl 的解藕
FacultyDeanMethod	院系教务员服务的方法接口
FacultyDeanMethodImpl	院系教务员所需服务具体方法的实现
FacultyDeanDatabaseMethod	提供数据库服务方法的接口

（剩余相关类的职责见表 13）

(3) 模块内部类的接口规范

FacultyDeanMethodController 的接口规范如表 5 所示。

表 5 FacultyDeanMethodController 的接口规范

提供的服务		
FacultyDeanMethodController .getFacultyDeanMethod	语法	public static FaacultyDeanMethod getFacultyDeanMethod()
	前置条件	无
	后置条件	返回一个已 FacultyDeanMethod 为接口的对象

FacultyDeanMethodImpl 的接口规范如表 6 所示

表 6 FacultyDeanMethodImpl 的接口规范

提供的服务（供接口）		
FacultyDean.login	语法	public boolean login(String ID,String password) throws RemoteException;
	前置条件	ID,password 符合输入规则
	后置条件	系统判断，当 ID 和 password 匹配 return true，反之 return false；

FacultyDean.getSelf	语法	public FacultyDean getSelf(String ID) throws RemoteException;
	前置条件	一位院系教务员已经登录
	后置条件	返回该院系教务员信息
FacultyDean.changePassword	语法	public boolean changePassword(String ID,String originalPassword,String password) throws RemoteException;
	前置条件	一位院系教务员已经登录
	后置条件	院系教务员修改密码。传入的 原密码若正确 return true，若原密码错误 return false。
FacultyDean.publishClass	语法	public boolean publishClass(String ID,Course c) throws RemoteException;
	前置条件	一位院系教务员已经登录
	后置条件	发布课程，若课程已存在，则发布失败 return false，反之 return true。
FacultyDean.getClassList	语法	public List<Course> getClassList(String ID) throws RemoteException;
	前置条件	一位院系教务员已经登录
	后置条件	院系教务员查看本院系课程列表 return
FacultyDean.getClass	语法	public Course getClass(String cID) throws RemoteException;
	前置条件	一位院系教务员已经登录
	后置条件	查看任何课程信息
FacultyDean.modifyClass	语法	Public Boolean modifyClass(Course c) throws RemoteException;
	前置条件	一位院系教务员已经登录
	后置条件	修改课程信息

需要的服务（需接口）

服务名	服务
DatabaseMethod.getConnection	得到数据库的服务引用
DatabaseMethod.search(String tableName,List clueName,List clue,List aimName)	在数据库中的数据进行搜索
DatabaseMethod.update(String tableName,String clueName,String clue,String aimName,String aim)	更新数据库中信息
DatabaseMethod.insert(String tableName,List<String> clueName,List clue)	向数据库中插入条目
DatabaseMethod.delete(String tableName,List<String> clueName,List<String> clue)	删除数据库中的条目

(4) 业务逻辑层的动态模型

图 16 表明了选课系统中，院系教务员登录，院系教务员输入帐号和密码后，院系教务员业务逻辑处理的相关对象之间的协作。

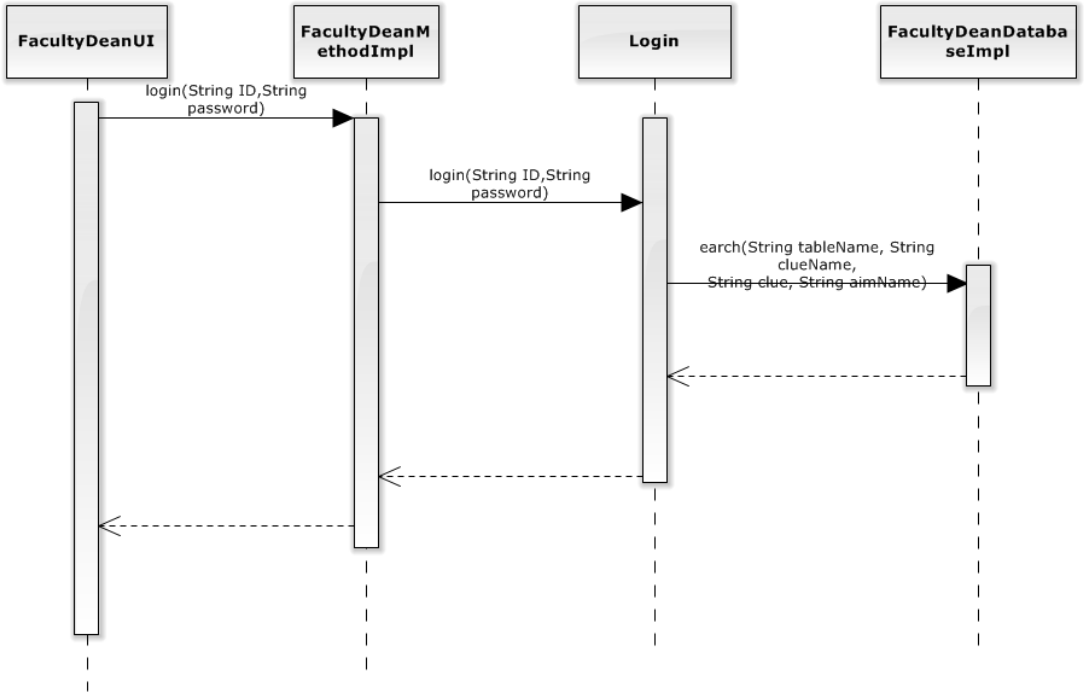

图 16 登录的顺序图

图 17 表明了选课系统中，院系教务员更改密码，院系教务员输入帐号、原密码和新密码后，院系教务员业务逻辑处理的相关对象之间的协作。

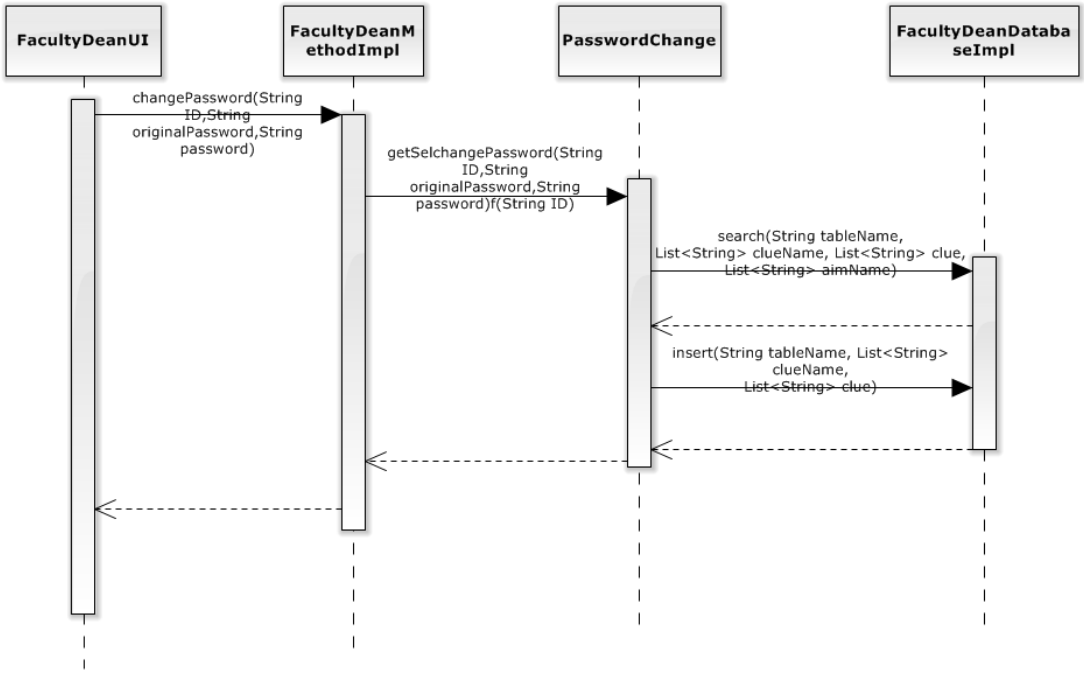

图 17 修改密码的顺序图

图 18 表明了选课系统中，院系教务员想要得到自己的基本信息，院系教务员业务逻辑处理的相关对象之间的协作。

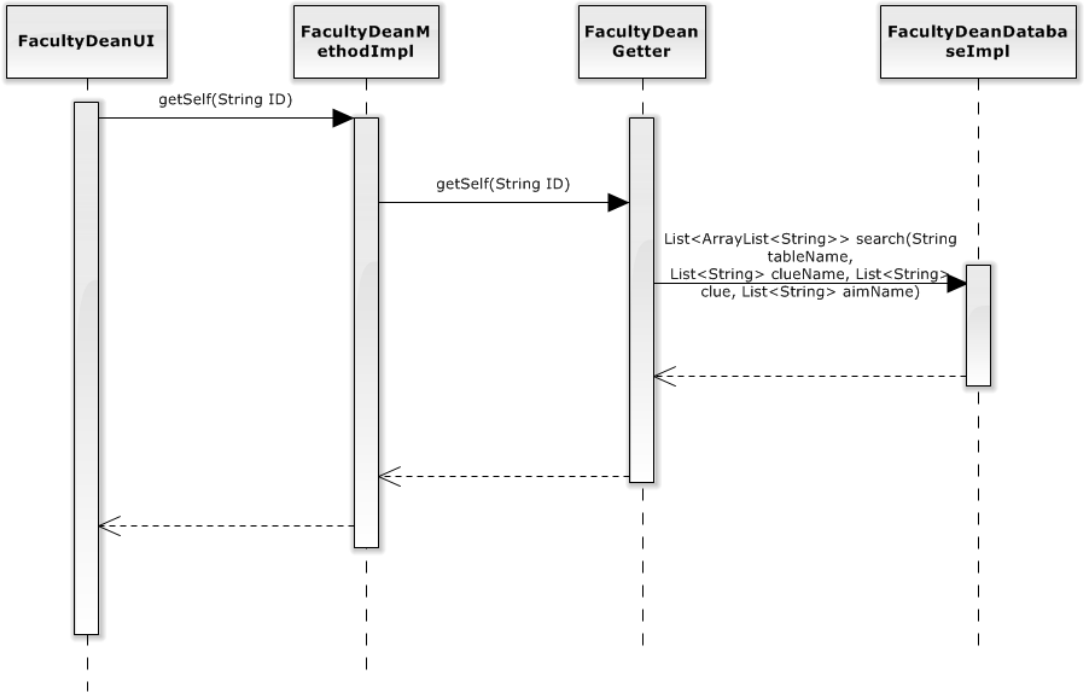

图 18 得到用户基本信息的顺序图

图 19 表明了选课系统中，院系教务员发布教学计划，院系教务员制定教学计划后，院系教务员业务逻辑处理的相关对象之间的协作。

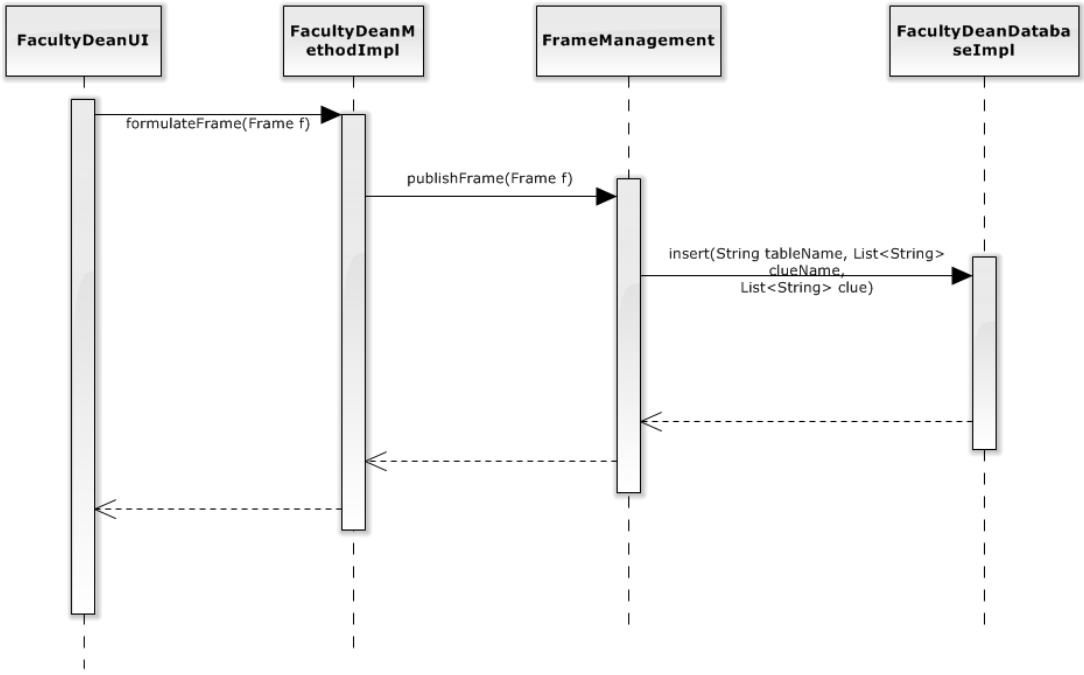

图 19 制定教学计划的顺序图

图 20 表明了选课系统中，院系教务员修改教学计划，院系教务员修改教学计划后，院系教务员业务逻辑处理的相关对象之间的协作。

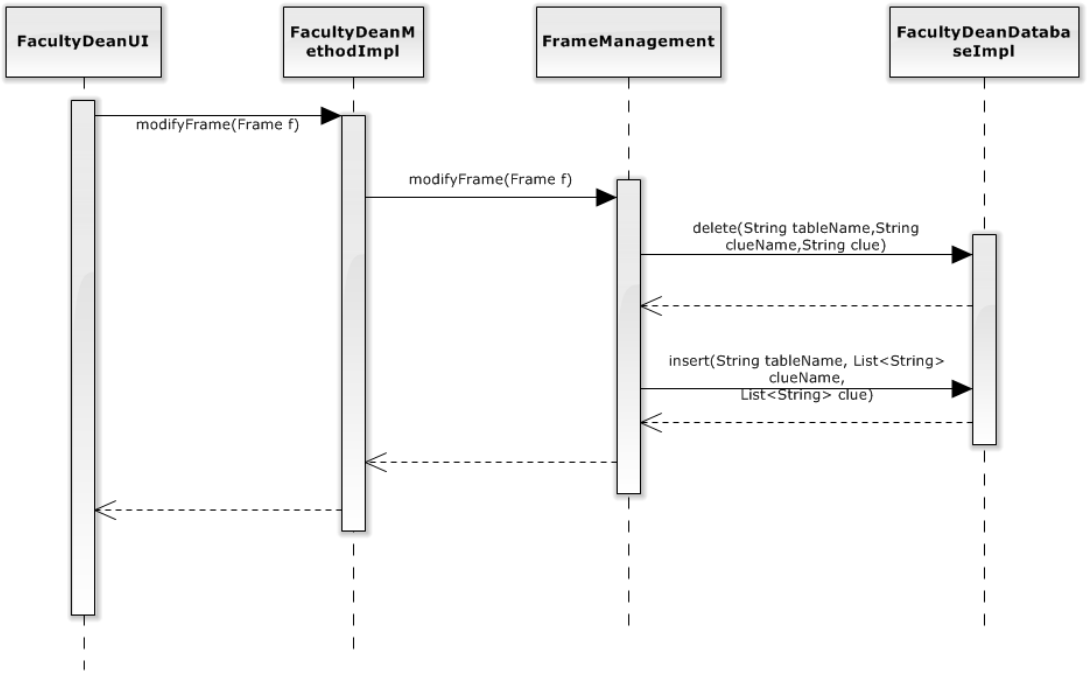

图 20 修改教学计划的顺序图

图 21 表明了选课系统中，院系教务员想要查看教学计划，院系教务员业务逻辑处理的相关对象之间的协作。

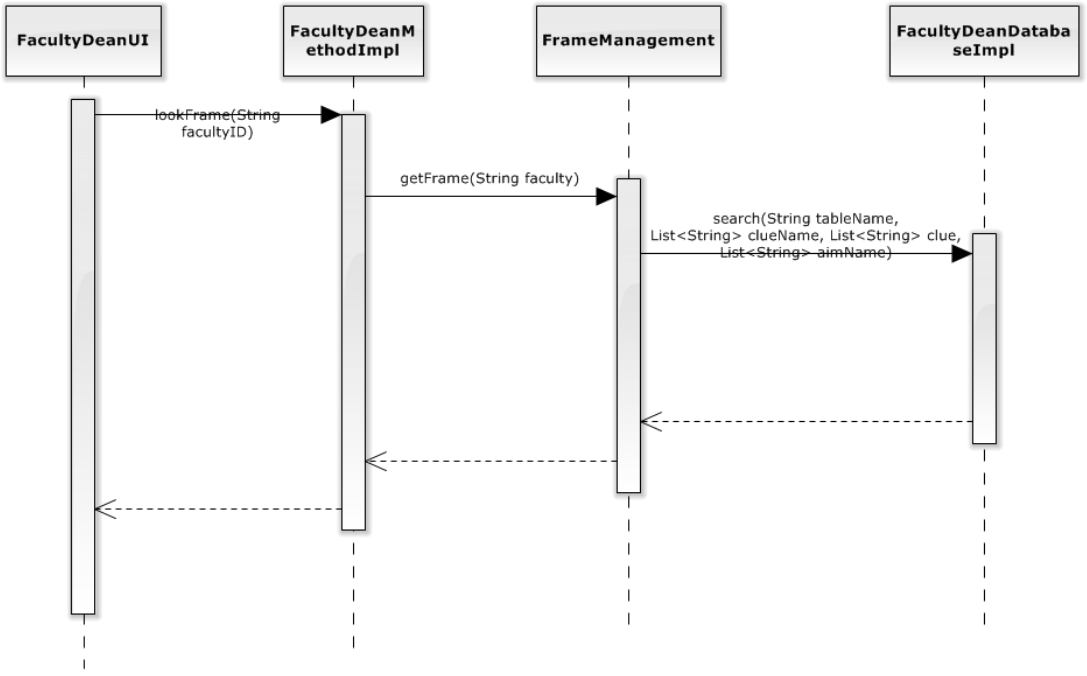

图 21 查看教学计划的顺序图

图 22 表明了选课系统中 院系教务员想要查看整体框架策略，院系教务员业务逻辑处理的相关对象之间的协作。

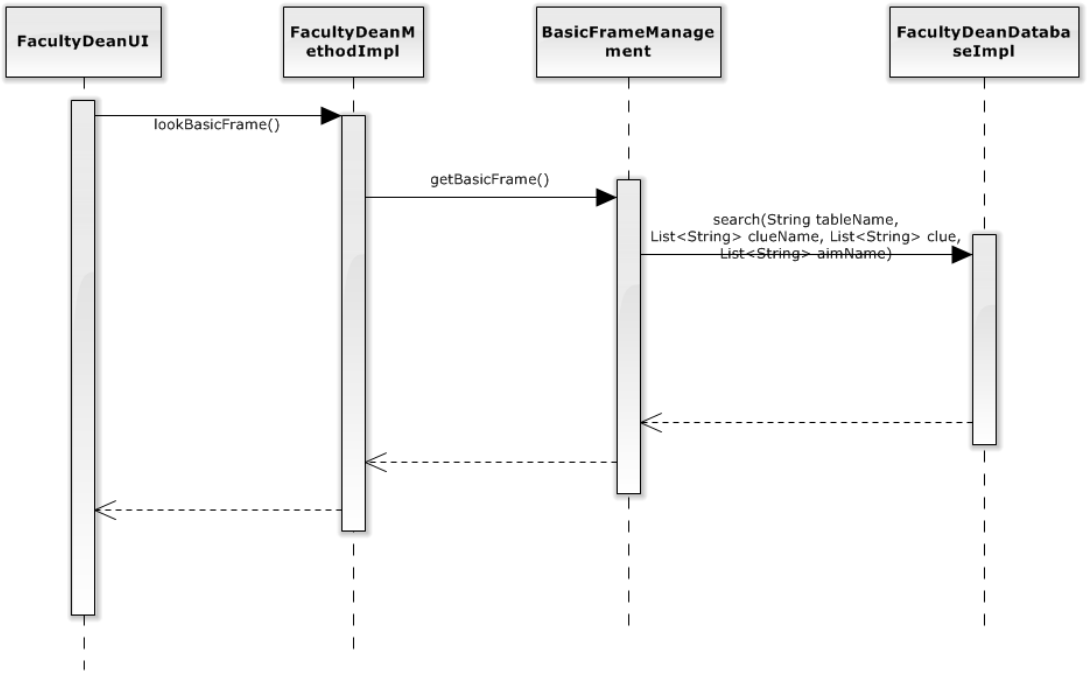

图 22 查看整体框架策略的顺序图

图 23 表明了选课系统中，院系教务员发布课程，院系教务员输入课程信息后，院系教务员业务逻辑处理的相关对象之间的协作。

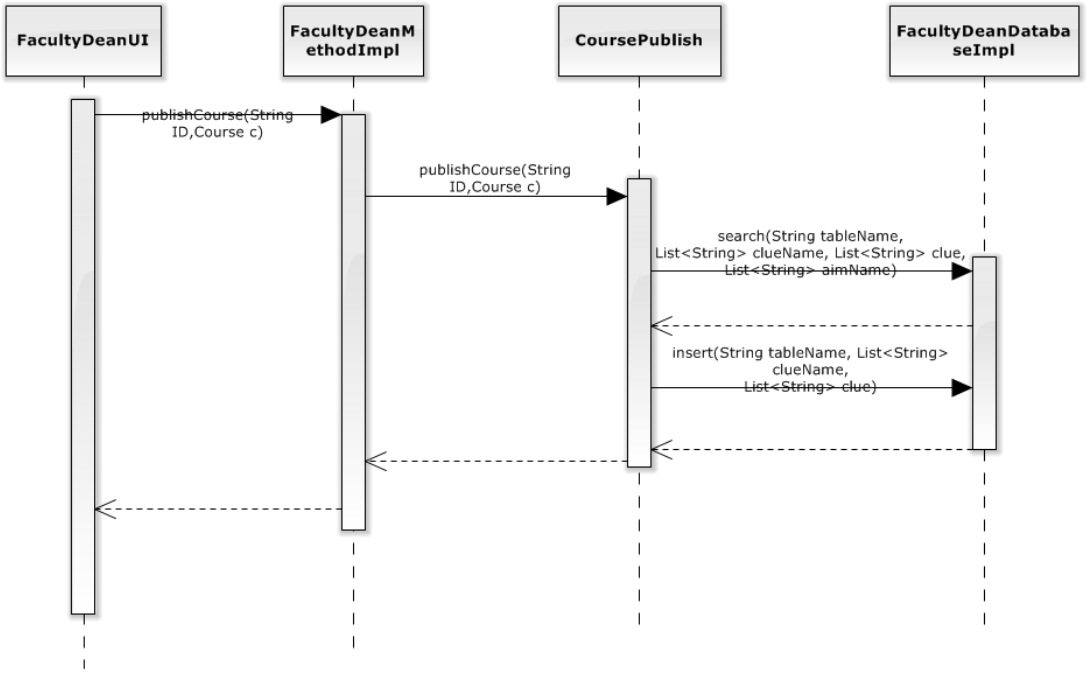

图 23 发布课程的顺序图

图 24 表明了选课系统中， 院系教务员修改课程，院系教务员修改课程信息后，院系教务员业务逻辑处理的相关对象之间的协作。

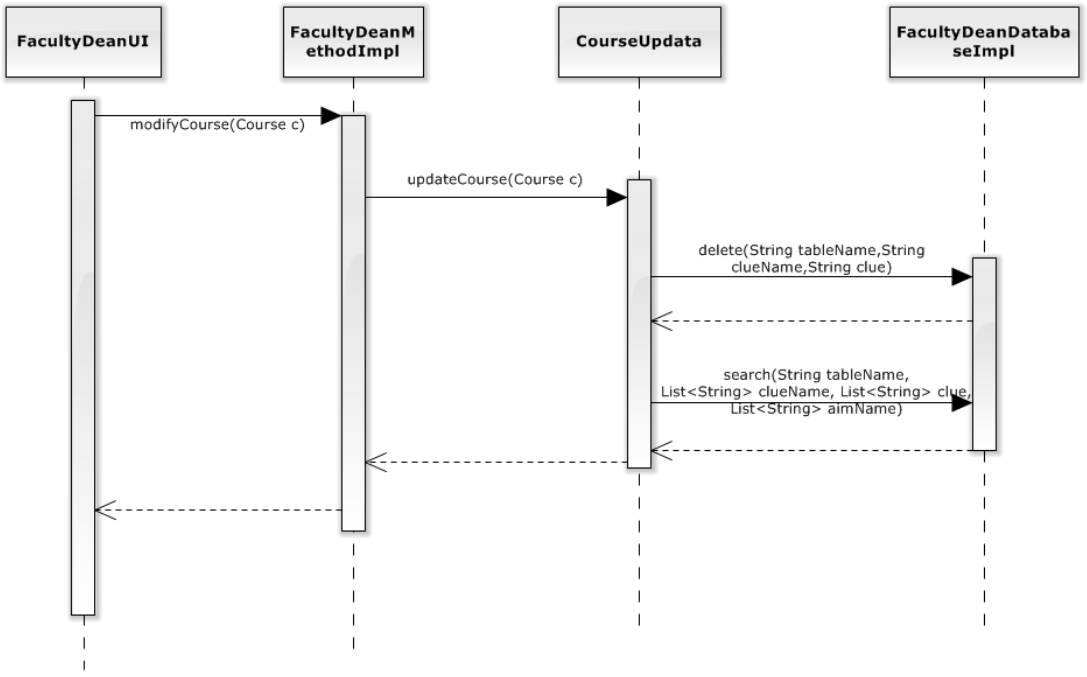

图 24 修改课程的顺序图

图 25 表明了选课系统中，院系教务员想要得到某个课程信息，院系教务员输入课程 ID 后，院系教务员业务逻辑处理的相关对象之间的协作。

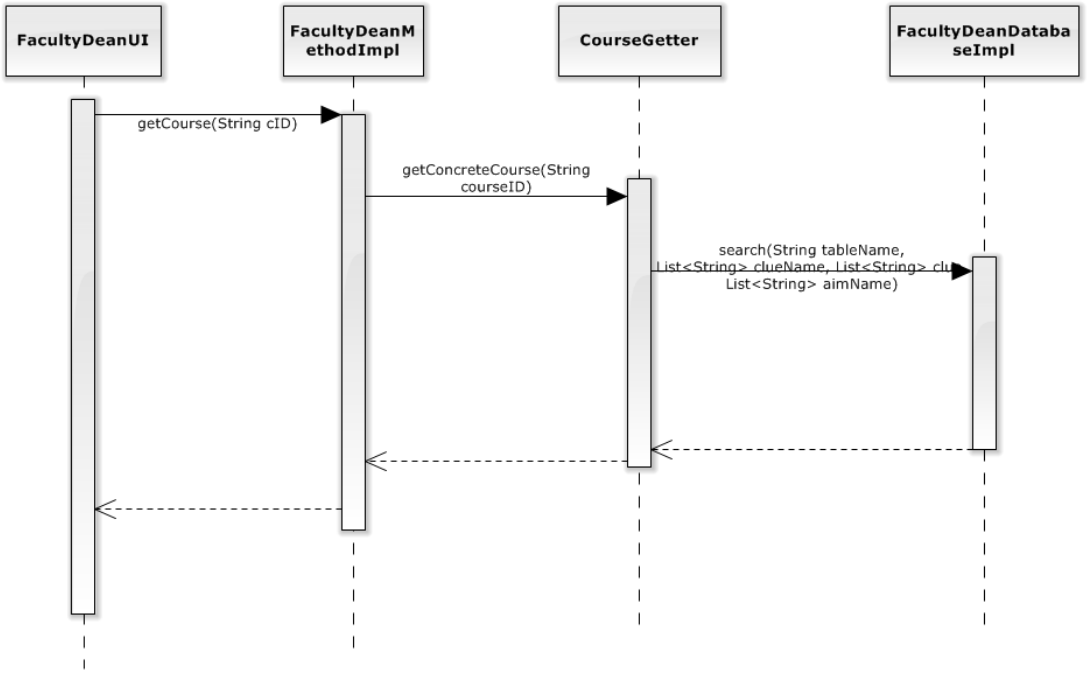

图 25 获取课程信息的顺序图

图 26 表明了选课系统中，院系教务原想要获取院系课程列表，院系教务员业务逻辑处理的相关对象之间的协作。

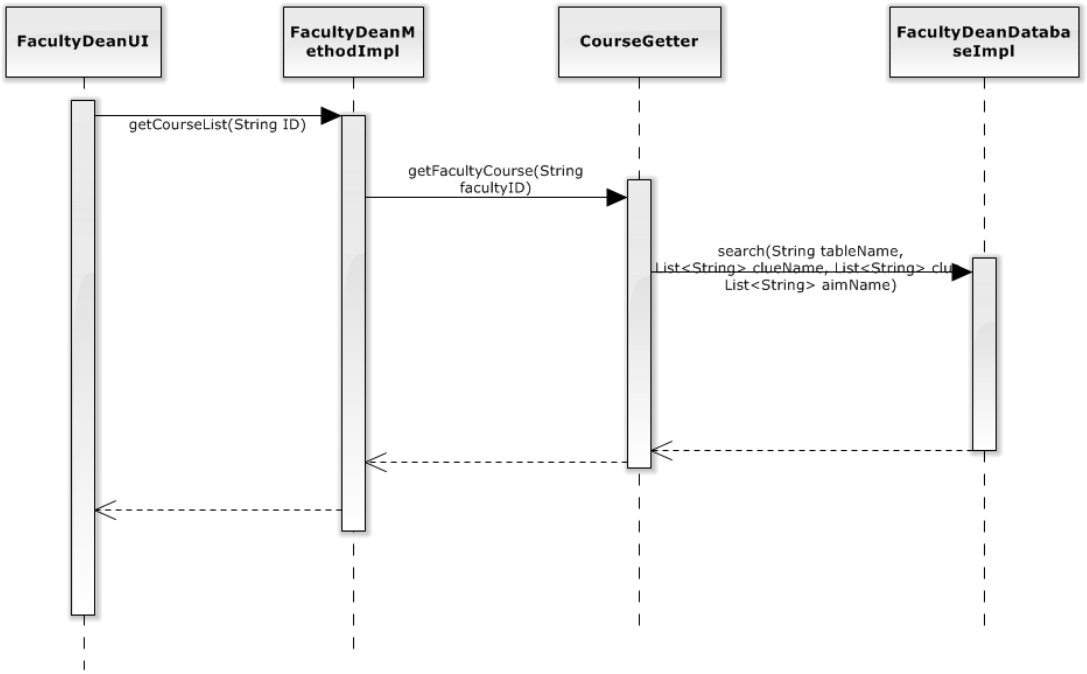

图 26 获取院系课程列表的顺序图

(5) 业务逻辑层的设计原理

利用单件模式，每个界面需要访问的业务逻辑由各自的控制器决定使用何种实现方法。

利用策略模式，FacultyDeanMethodImpl 所需实现的方法都委托到其他类来实现，提高了代码的复用性和易修改性。

4.1.3 teacherImpl 模块

(1) 模块描述

teacherImpl 模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

teacherImpl 模块的职责与接口参见软件体系结构描述文档表 9

(2) 整体结构

根据体系结构设计，我们将系统分为展示层（presentation）、业务逻辑层（logic）、数据层（data）。每一层之间为了增加灵活性，我们会添加接口。展示层和业务逻辑层之间添加 com.logicService.TeacherMethod 接口。业务逻辑层和数据层之间添加 com.dataService.TeacherDatabaseMethod 接口。为了隔离业务逻辑职责和逻辑控制职责，我们增加了 TeacherMethodController,这样 TeacherMethodController 会将对院系教务员业务的业务处理逻辑委托给 TeacherMethodImpl 对象。Course,Teacher 是作为老师业务中的持久化对象被添加到设计模型中去的。TeacherMethodImpl 采用策略模式，将方法分解到 CourseGetter, CourseInfroFiln, TeacherGetter, BasicFrameManagement, Login, PasswordChange 中去实现，提高了代码的复用性和易修改性。

teacherImpl 模块的设计如图 27 所示。

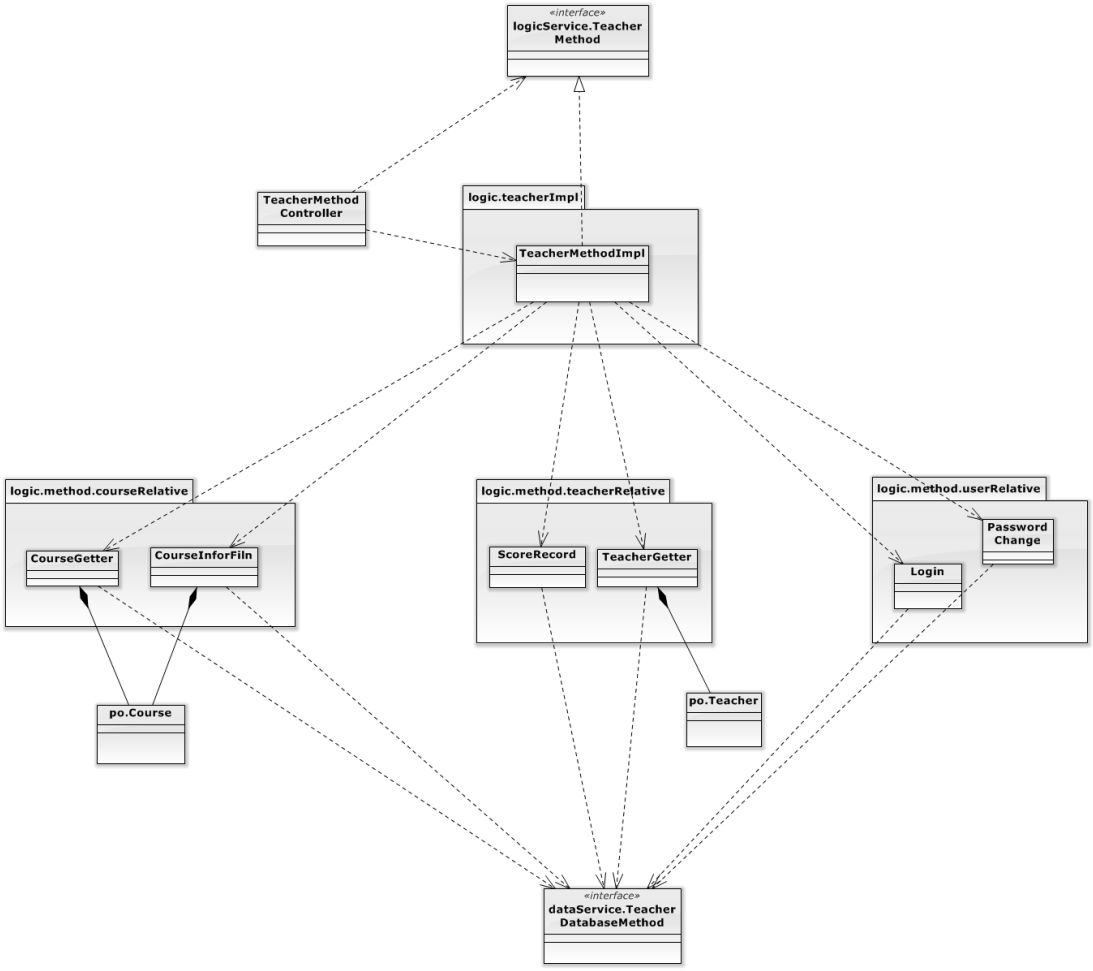

图 27 teacherImpl 模块各个类的设计

teacherImpl 模块各个类的职责如表 7 所示。

表 7 teacherDeanImpl 模块各个类的职责

模块	职责
TeacherMethodController	负责实现老师所需要服务于 TeacherMethodImpl 的解藕
TeacherMethod	老师服务的方法接口
TeacherMethodImpl	老师所需服务具体方法的实现
TeacherDatabaseMethod	提供数据库服务方法的接口

（剩余相关类的职责见表 13）

(3) 模块内部类的接口规范

TeacherMethodController 的接口规范如表 8 所示。

表 8TeacherMethodController 的接口规范

提供的服务		
TeacherMethodController .getTeacherMethod	语法	public static TeacherMethod getTeacherMethod()
	前置条件	无
	后置条件	返回一个已 TeacherMethod 为接口的对象

TeacherMethodImpl 的接口规范如表 9 所示。

表 9TeacherMethodImpl 的接口规范

提供的服务（供接口）		
TeacherMethod.login	语法	public boolean login(String ID,String password) throws RemoteException;
	前置条件	无
	后置条件	系统判断，当 ID 和 password 匹配 return true，反之 return false；
TeacherMethod.getSelf	语法	public Teacher getSelf(String ID) throws RemoteException;
	前置条件	一位教师已经登录
	后置条件	返回该院系教务员信息
TeacherMethod. changePassword	语法	public boolean changePassword(String ID,String originalPassword,String password) throws RemoteException;
	前置条件	一位教师已经登录
	后置条件	教师修改密码。传入的 原密码若正确 return true，若原密码错误 return false。
TeacherMethod. filnCourseInfor	语法	public boolean filnCourseInfor(String cID,String text) throws RemoteException;
	前置条件	一位老师已经登录
	后置条件	老师填写课程信息，若成功 return true，反之 return false
TeacherMethod. getCourseStudent	语法	public List<Student> getCourseStudent(String cID) throws RemoteException;
	前置条件	一位老师已经登录
	后置条件	老师查看选课学生，返回选择该课程的学生列表。
TeacherMethod.recordScore	语法	public boolean recordScore(String cID,Map score) throws RemoteException;
	前置条件	一位老师已经登录
	后置条件	老师录入成绩

需要的服务（需接口）	
服务名	服务
DatabaseMethod.getConnection	得到数据库的服务引用
DatabaseMethod.search(String tableName,List clueName,List clue,List aimName)	在数据库中的数据进行搜索
DatabaseMethod.update(String tableName,String clueName,String clue,String aimName,String aim)	更新数据库中信息
DatabaseMethod.insert(String tableName,List<String> clueName,List clue)	向数据库中插入条目
DatabaseMethod.delete(String tableName,List<String>	删除数据库中的条目

clueName,List<String> clue)	
DatabaseMethod.getConnection	得到数据库的服务引用

(4) 业务逻辑层的动态模型

图 28Teacher_login 在选课系统中，当用户在登陆界面输入 ID 和密码后，登陆业务逻辑处理的相关对象的合作。

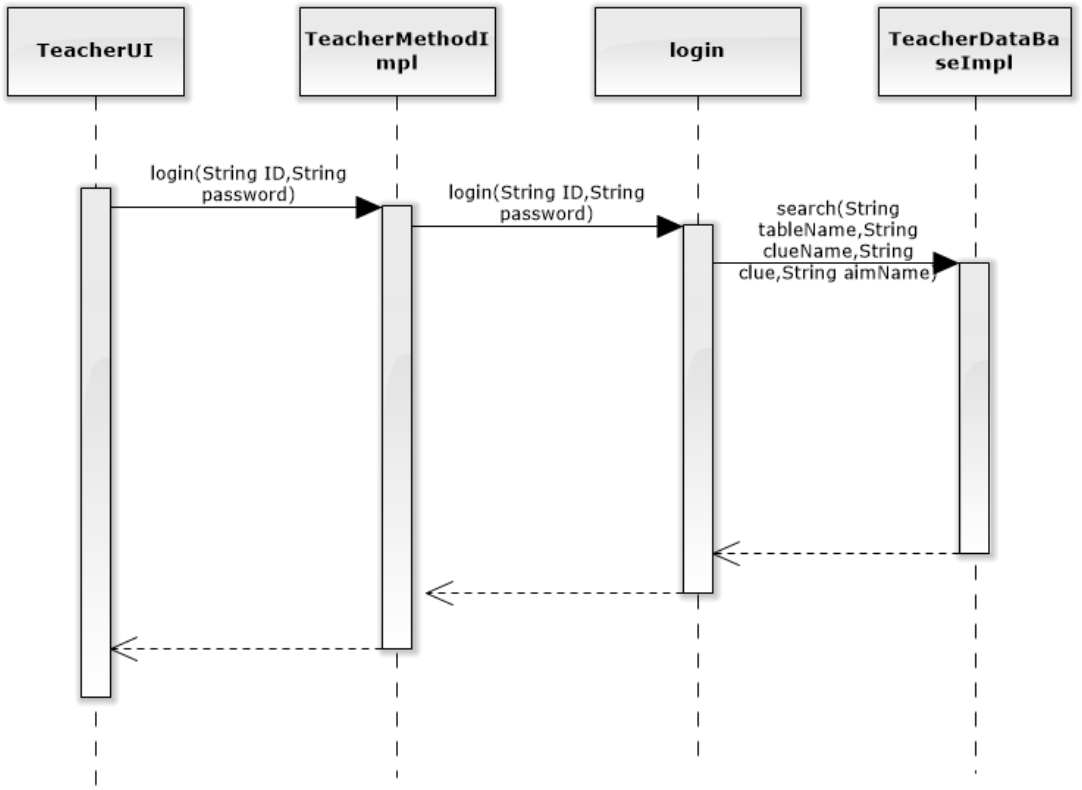

图 28

图 29Teacher_PasswordChange 在选课系统中，当老师在输入 ID、原始密码、新密码后，修改密码业务逻辑处理的相关对象的合作。

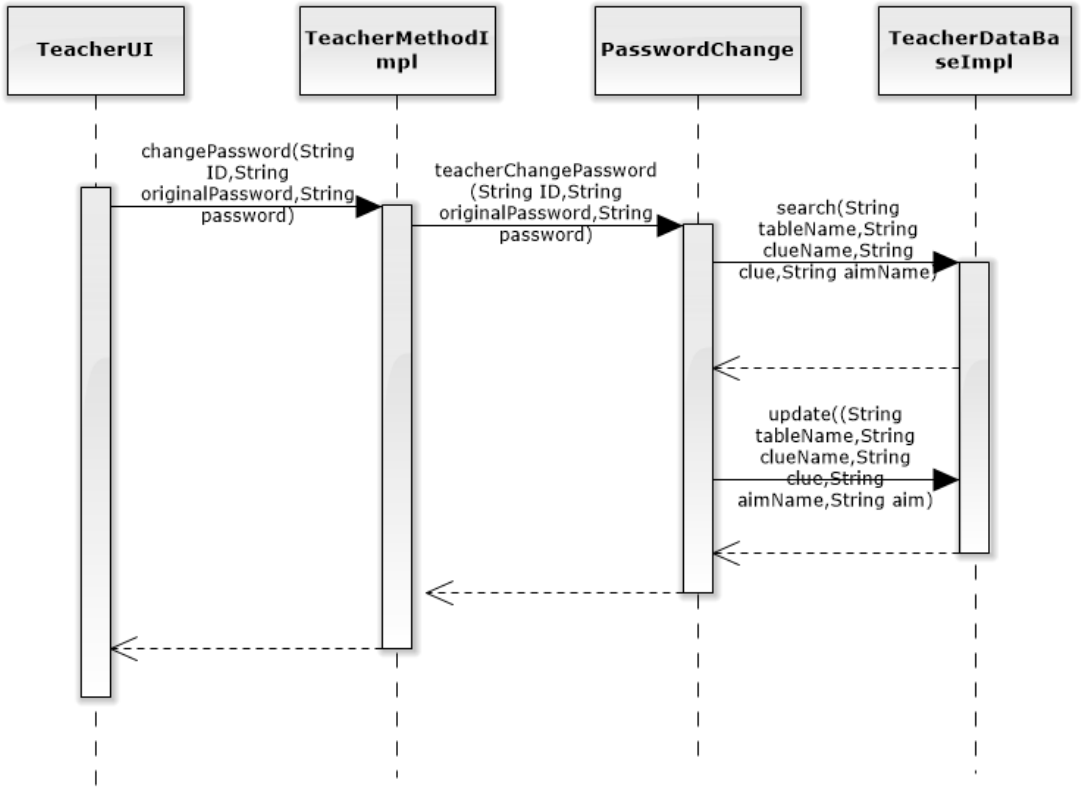

图 29

图 30Teacher_getCourseStudent 在选课系统中，当老师在输入课程 ID 后，展示课程所有学生业务逻辑处理的相关对象的合作。

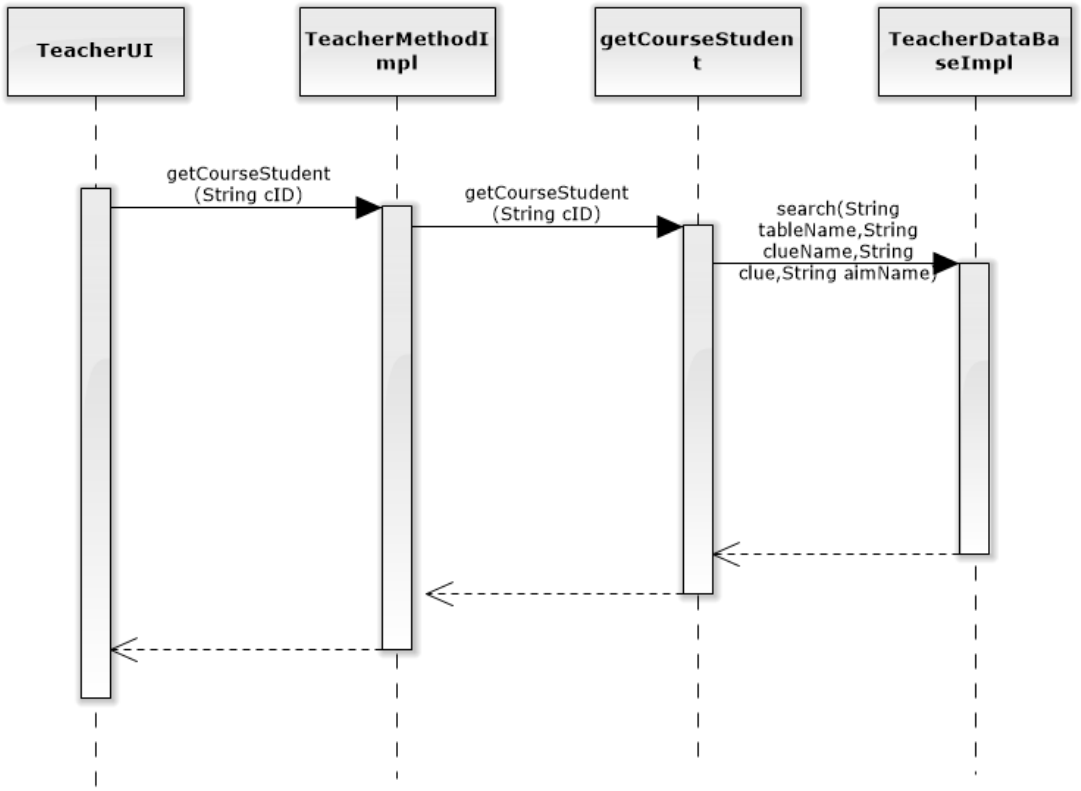

图 30

图 31Teacher_filnCourseInfo 在选课系统中，当老师输入课程 ID、课程描述信息后，完善课程信息业务逻辑处理的相关对象的合作。

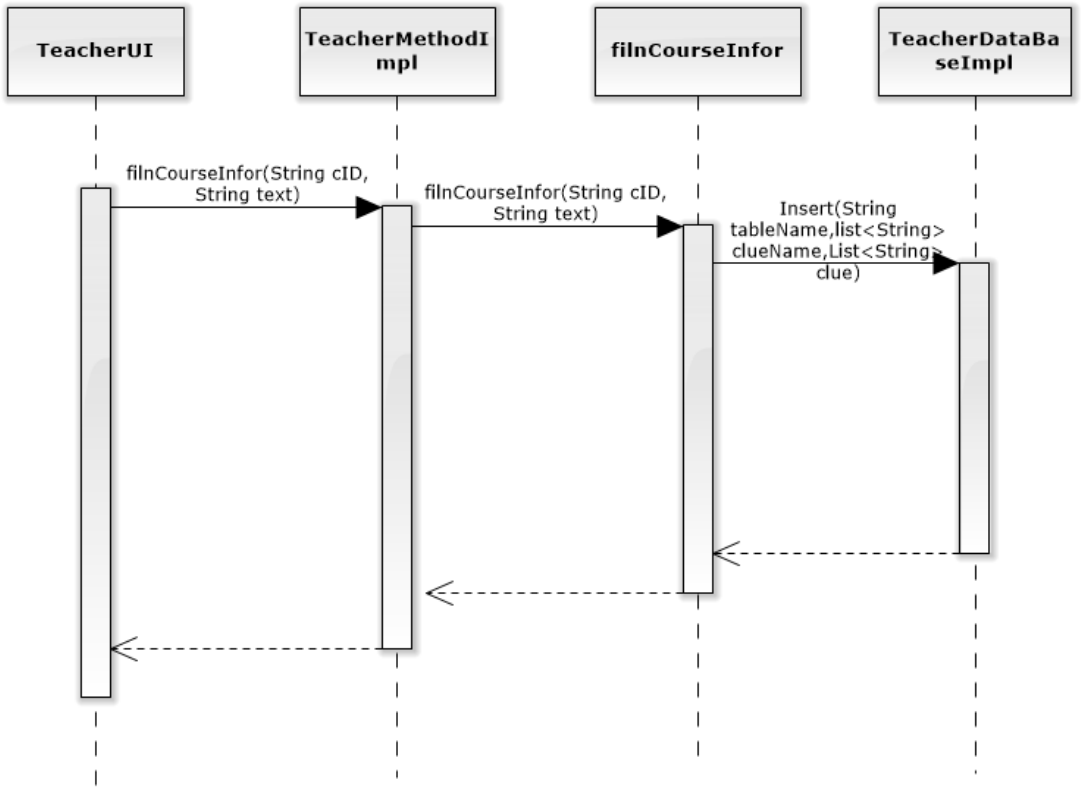

图 31

图 32Teacher_ScoreRecord 在选课系统中，当老师输入课程 ID、与学生对应的成绩时，登记成绩业务逻辑处理的相关对象的合作。

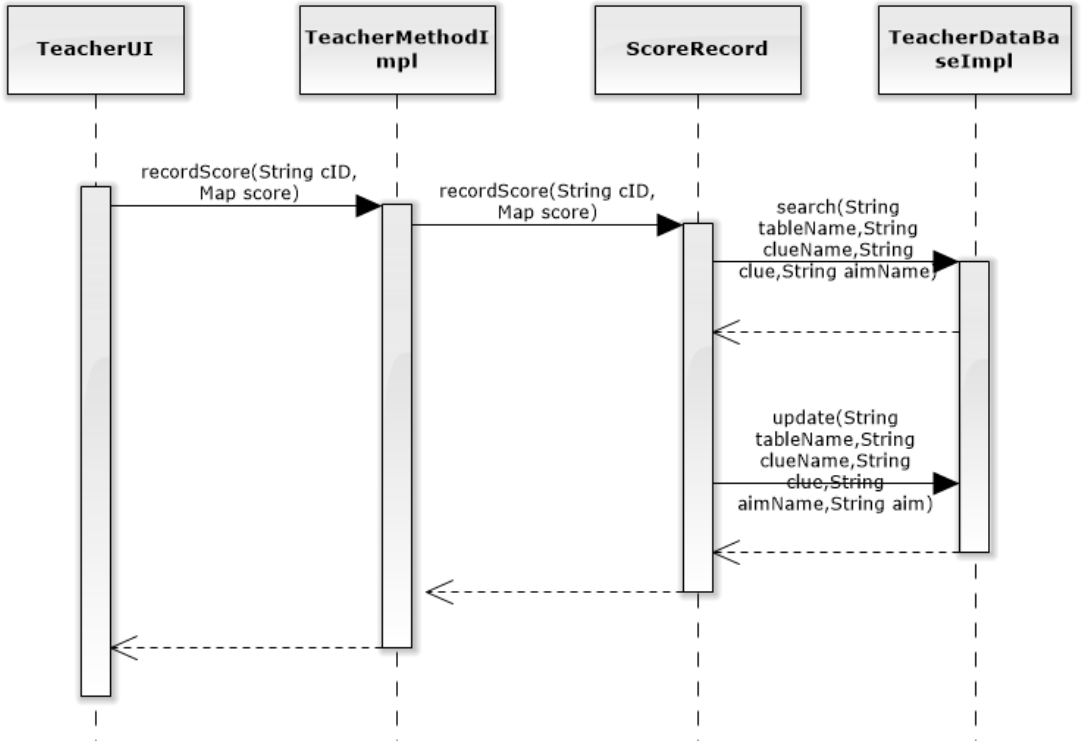

图 32

图 33 Teacher_getSelf 在选课系统中，当老师输入 ID 后，显示老师信息业务逻辑处理的相关对象的合作。

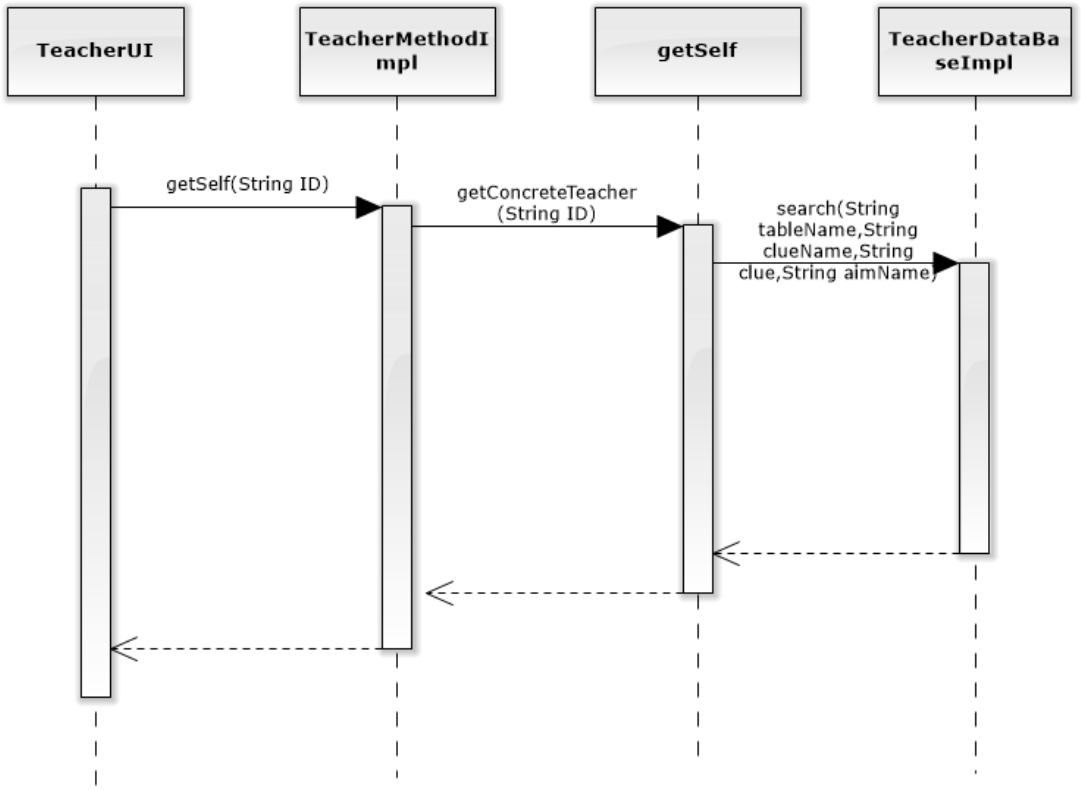

图 33

图 34 Teacher_getCourseList 在选课系统中，当老师输入课程 ID 后，显示课程列表业务逻辑处理的相关对象的合作。

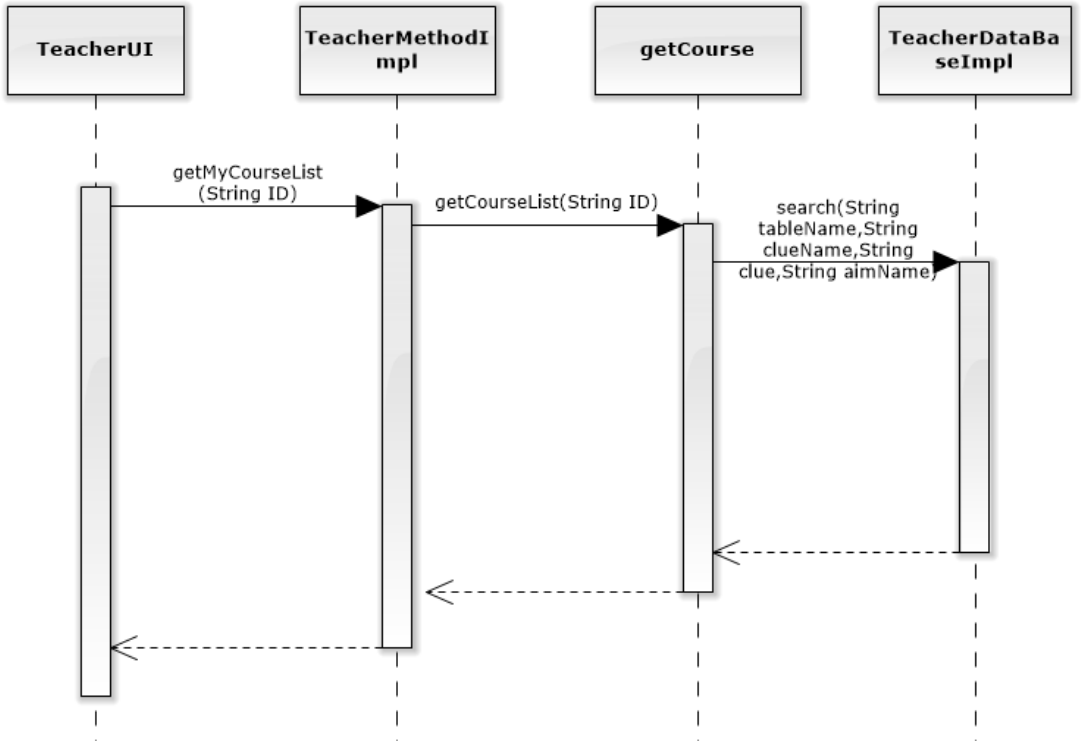

图 34

(5) 业务逻辑层的设计原理

利用单件模式，每个界面需要访问的业务逻辑由各自的控制器决定使用何种实现方法。

利用策略模式，TeacherMethodImpl 所需实现的方法都委托到其他类来实现，提高了代码的复用性和易修改性。

4.1.4 studentImpl 模块

(1) 模块描述

studentImpl 模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

studentImpl 模块的职责与接口参见软件体系结构描述文档表 9。

(2) 整体结构

根据体系结构设计，我们将系统分为展示层（presentation）、业务逻辑层（logic）、数据层（data）。每一层之间为了增加灵活性，我们会添加接口。展示层和业务逻辑层之间添加 com.logicService.StudentMethod 接口。业务逻辑层和数据层之间添加 com.dataService.StudentDatabaseMethod 接口。为了隔离业务逻辑职责和逻辑控制职责，我们增加了 StudentMethodController,这样 StudentMethodController 会将对院系教务员业务的业务处理逻辑委托给 StudentMethodImpl 对象。Student,Course 是作为院系教务员的持久化对象被添加到设计模型中去的。StudentMethodImpl 采用策略模式，将方法分解到 CourseGetter, CourseSelect, StudentCourseListGetter, StudentScoreGetter,StudentGetter,Login,PasswordChange 中去实现，提高了代码的复用性和易修改性。

studentImpl 模块的设计如图 35 所示

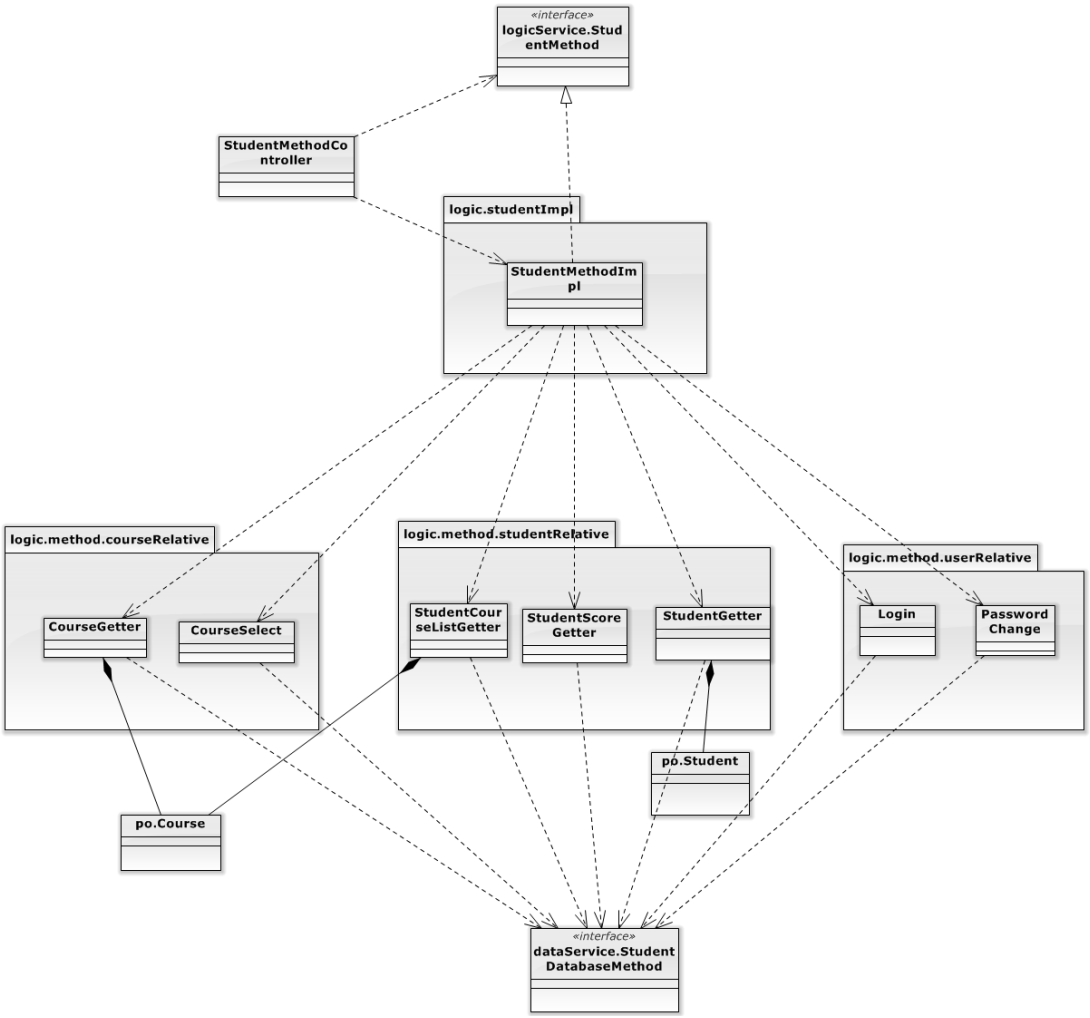

图 35 studentImpl 模块各个类的设计

studentImpl 模块各个类的职责如表 10 所示。

表 10 studentDeanImpl 模块各个类的职责

模块	职责
StudentMethodController	负责实现老师所需要服务于 StudentMethodImpl 的解耦
StudentMethod	老师服务的方法接口
StudentMethodImpl	老师所需服务具体方法的实现
StudentDatabaseMethod	提供数据库服务方法的接口

（剩余相关类的职责见表 13）

(3) 模块内部类的接口规范

StudentMethodController 的接口规范如表 11 所示。

表 11 StudentMethodController 的接口规范

提供的服务		
StudentMethodController .getStudentMethod	语法	public static StudentMethod getStudentMethod()
	前置条件	无
	后置条件	返回一个已 StudentMethod 为接口的对象

StudentMethodImpl 的接口规范如表 12 所示。

表 12 StudentMethodImpl 的接口规范

提供的服务（供接口）		
StudentMethod.login	语法	public boolean login(String ID,String password) throws RemoteException;
	前置条件	用户输入有效的账号和密码。
	后置条件	根据账号查找是否存在相应的用户，根据账号验证密码是否正确，如果通过验证，则登录成功，返回 True 否则，返回 False
StudentMethod.getSelf	语法	public Student getSelf(String ID) throws RemoteException;
	前置条件	一位学生已经登录
	后置条件	返回该学生信息
StudentMethod. changePassword	语法	public boolean changePassword(String ID,String originalPassword,String password) throws RemoteException;
	前置条件	一位学生已经登录，且用户输入有效的账号，原密码和现密码。
	后置条件	学生修改密码。根据账号查找是否存在相应的用户，根据账号验证旧密码是否正确，如果通过验证，则将密码修改为现密码，且返回 True，否则，返回 False
StudentMethod. selectCourse	语法	public boolean selectCourse(String ID,String cID) throws RemoteException;
	前置条件	一位学生已经登录，且用户输入有效地学生 ID 和课程 ID

	后置条件	学生选择课程，根据学生 ID 查找是否存在相应的用户，根据课程 ID 查找是否存在相应的课程，如果通过验证，则学生选课成功，且返回 True，否则，返回 False
StudentMethod. quitCourse	语法	public boolean quitCourse(String ID,String cID) throws RemoteException;
	前置条件	一位学生已经登录，且用户输入有效地学生 ID 和课程 ID
	后置条件	学生退选课程，根据学生 ID 查找是否存在相应的用户，根据课程 ID 查找是否存在相应的课程，如果该学生已选择该课程，则退课成功，返回 True，如果该学生未选择该课程，则系统提示该学生未选择该课程，返回 False，其余情况均返回 False
StudentMethod.getScore	语法	public int getScore(String ID,String cID) throws RemoteException;
	前置条件	一位学生已经登录，且用户输入有效地学生 ID 和课程 ID
	后置条件	学生查看课程成绩，根据学生 ID 查找是否存在相应的用户，根据课程 ID 查找是否存在相应的课程，如果该学生已选择该课程，则显示学生成绩，返回 True，如果该该课程未登记成绩，则系统提示未登记成绩，返回 False，如果该学生未选择该课程，则系统提示该学生未选择该课程，返回 False，其余情况均返回 False
StudentMethod.getCourseList	语法	public List<Course> getCourseList(String ID) throws RemoteException;
	前置条件	一位学生已经登录，且用户输入有效地学生 ID
	后置条件	学生查看自己的选课列表，根据账号查找是否存在相应的用户，通过验证，则返回选课列表
StudentMethod.getCourseList	语法	public Course course(String cID) throws RemoteException;
	前置条件	一位学生已经登录，且用户输入有效地课程 ID
	后置条件	学生查看自己的选课列表，根据课程 ID 查找是否存在相应的课程，通过验证，则返回课程信息

(4) 业务逻辑层的动态模型

图 36 表明了选课系统中，学生登录，学生输入帐号和密码后，学生业务逻辑处理的相关对象之间的协作。

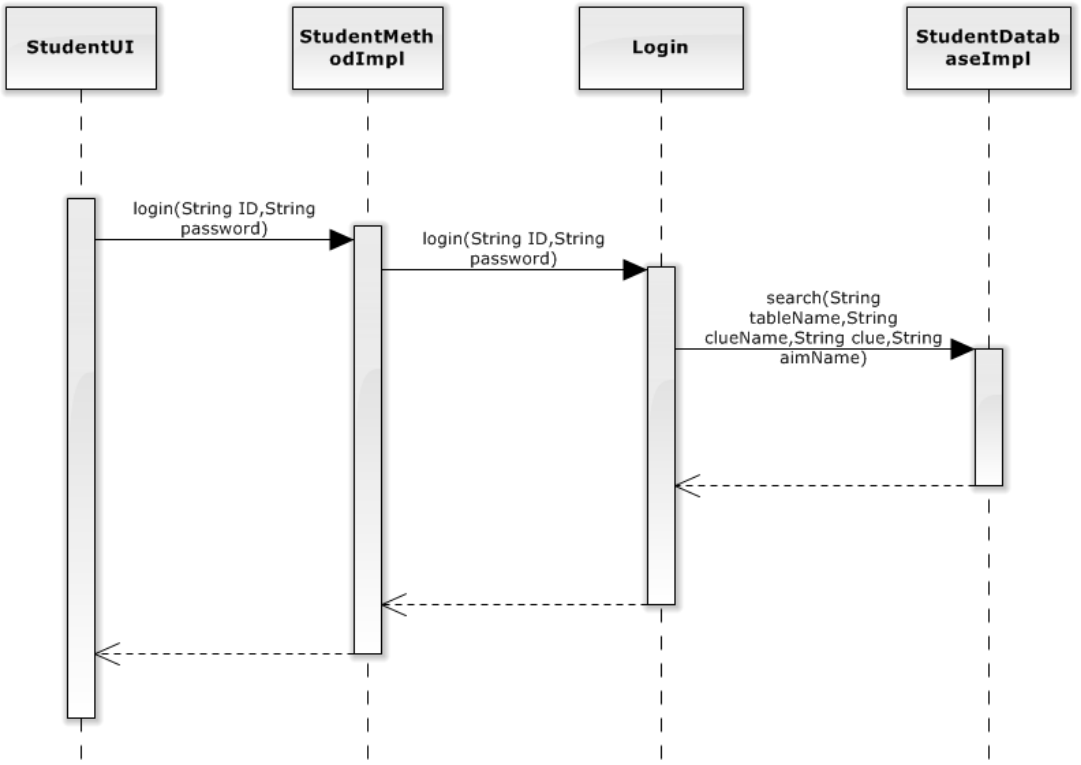

图 36 登录的顺序图

图 37 表明了选课系统中，学生更改密码，学生输入帐号、原密码和新密码后，学生业务逻辑处理的相关对象之间的协作。

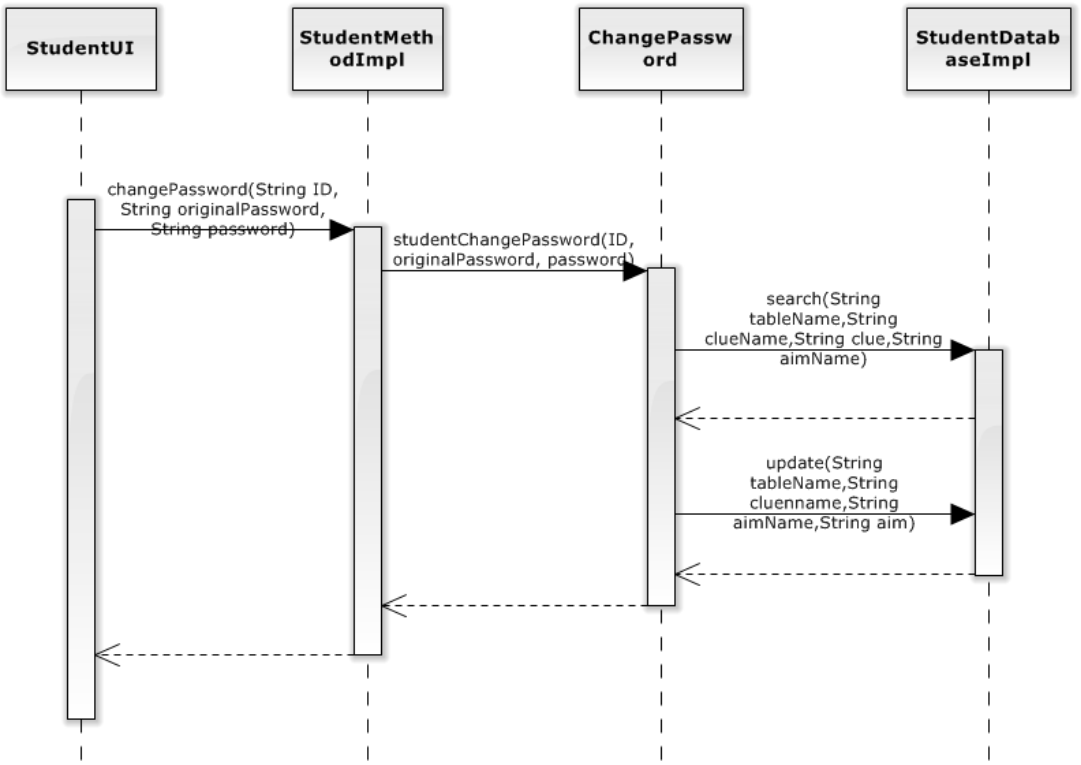

图 37 修改密码的顺序图

图 38 表明了选课系统中，学生想要退选自己的课程，学生业务逻辑处理的相关对象之间的协作。

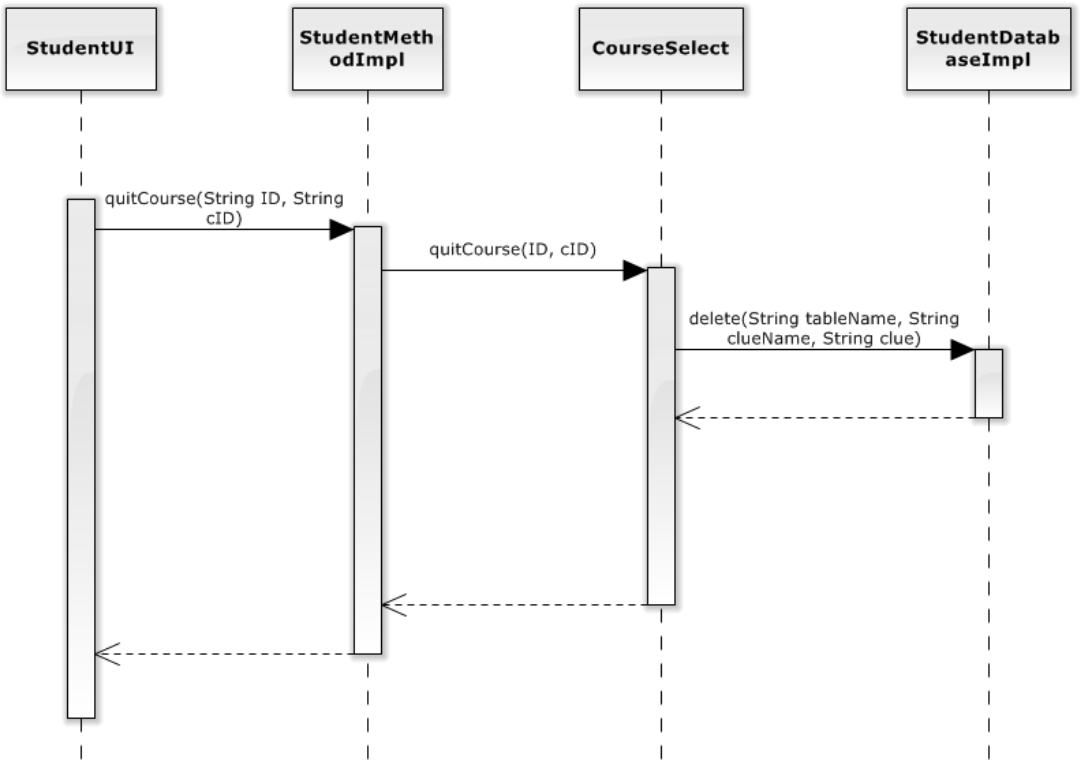

图 38 退选课程的顺序图

图 39 表明了选课系统中，学生想要得到选课列表，学生业务逻辑处理的相关对象之间的协作。

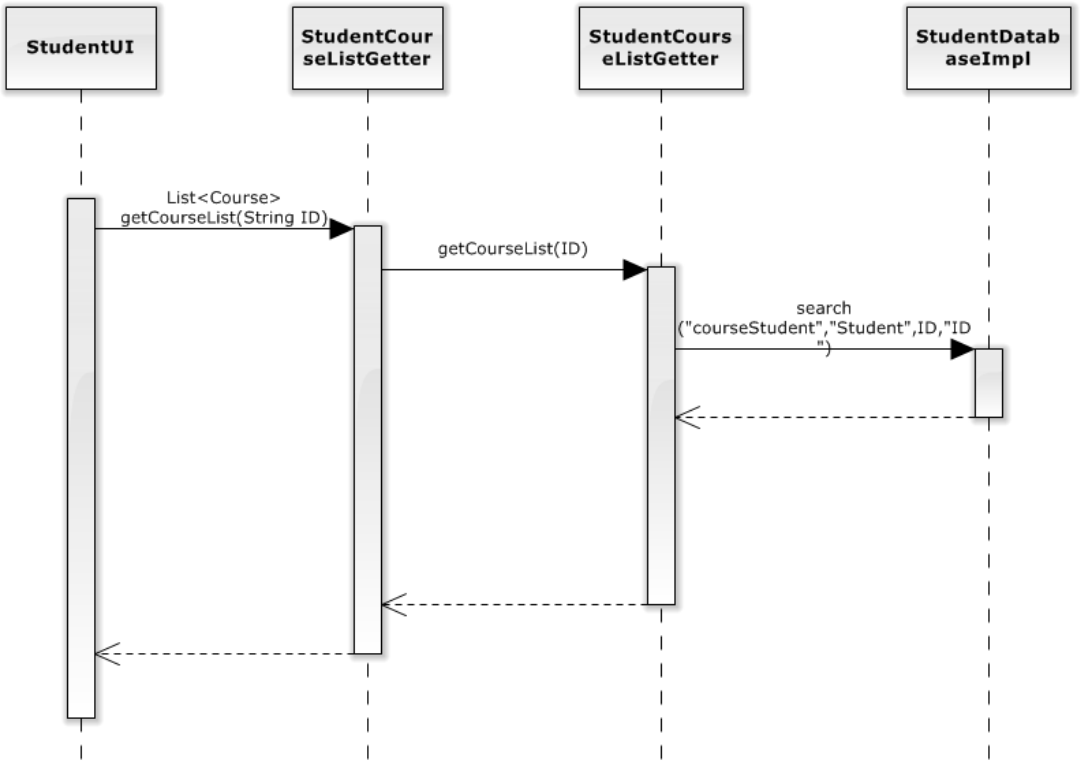

图 39 获得选课列表的顺序图

图 40 表明了选课系统中，学生想要查看自己的课程成绩，学生业务逻辑处理的相关对象之间的协作。

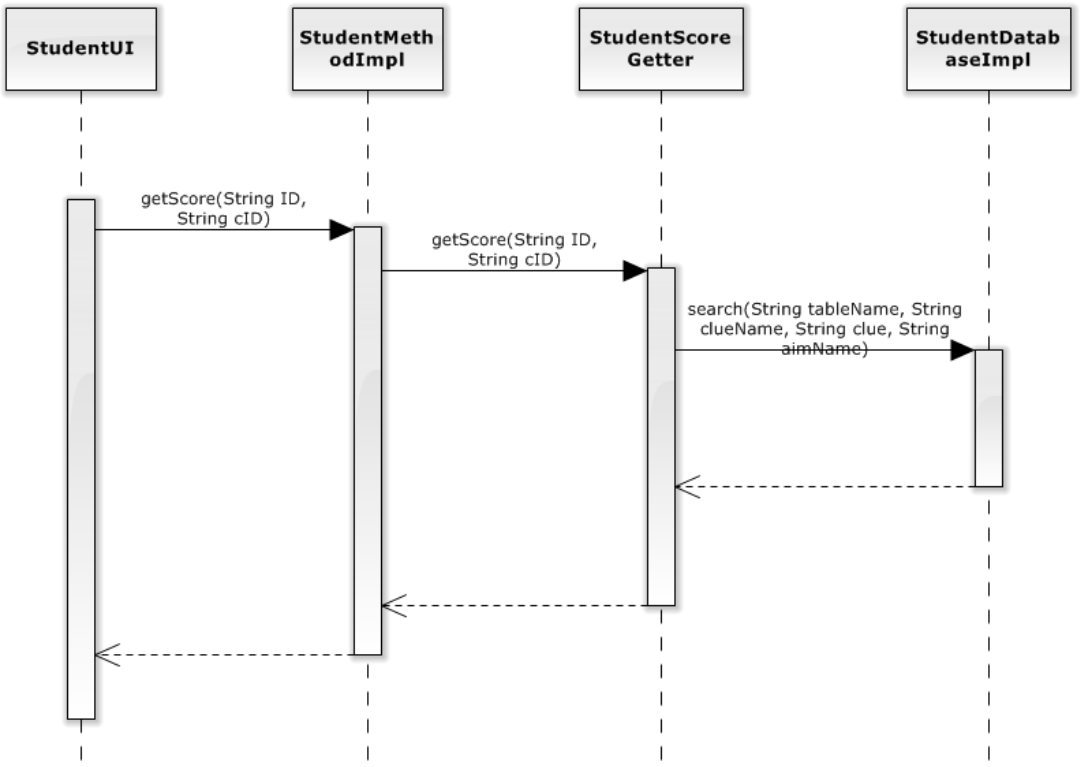

图 40 查看课程成绩的顺序图

图 41 表明了选课系统中，学生想要查看任意课程，学生输入课程号后，学生业务逻辑处理的相关对象之间的协作。

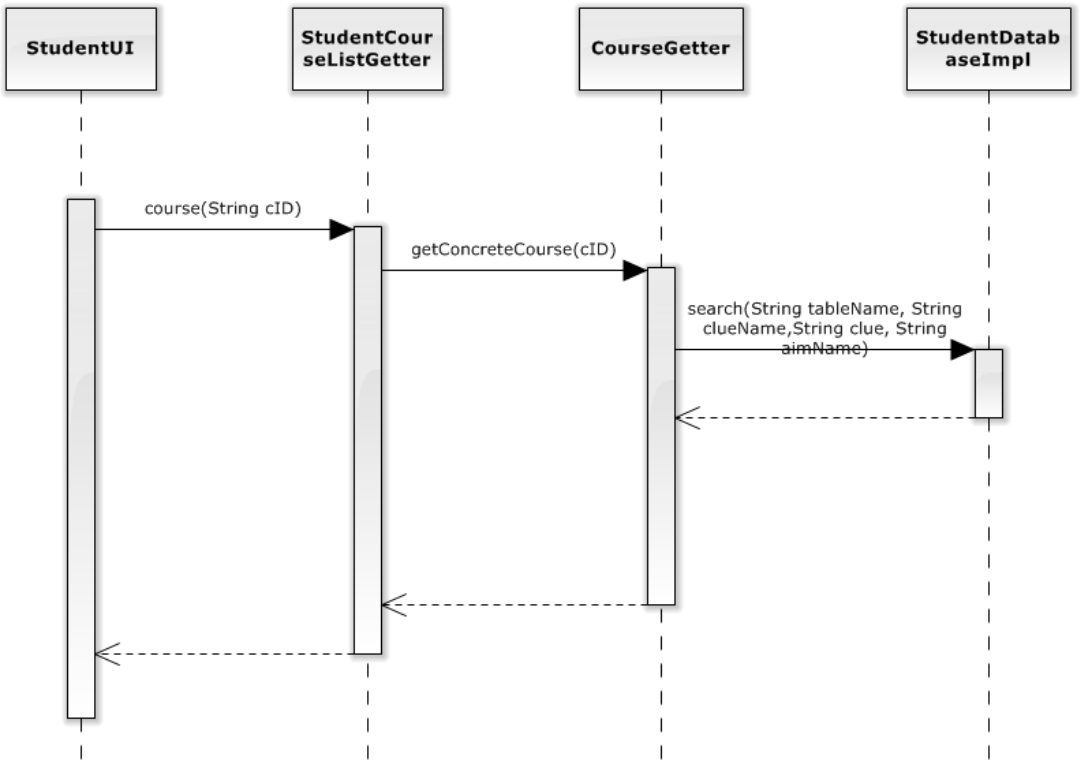

图 41 查看任意课程的顺序图

图 42 表明了选课系统中，学生想要得到自己的基本信息，学生业务逻辑处理的相关对象之间的协作。

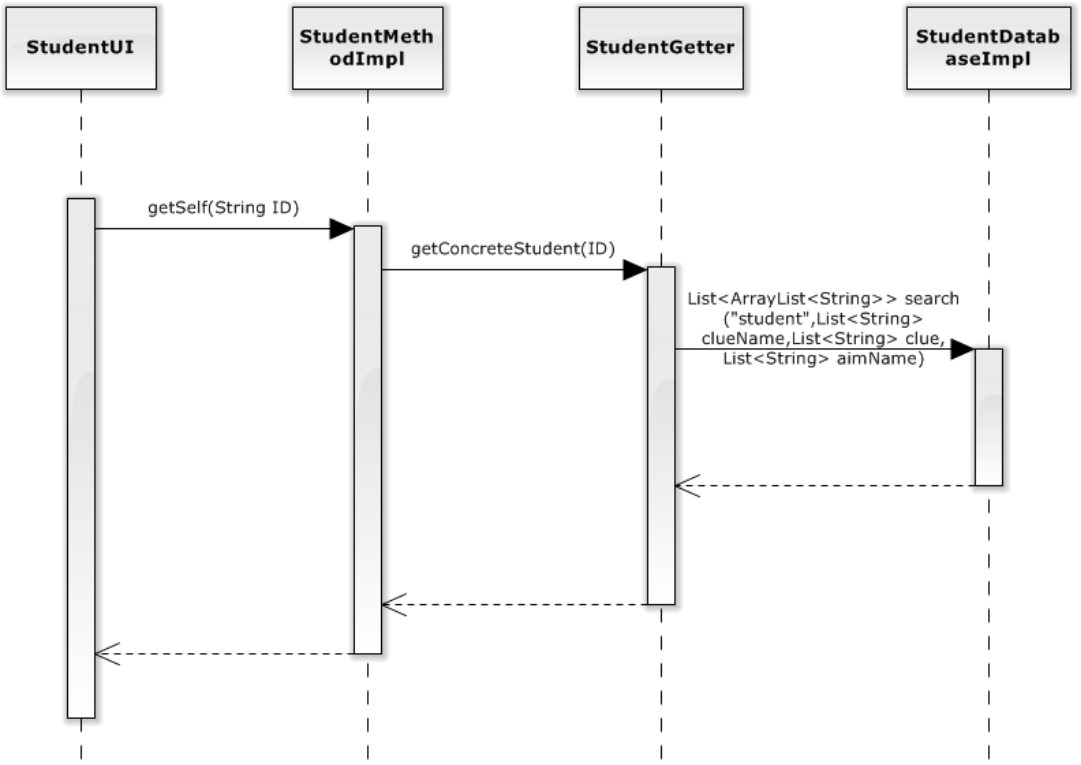

图 42 获取自身信息的顺序图

图 43 表明了选课系统中，学生想要选择课程时，学生业务逻辑处理的相关对象之间的协作。

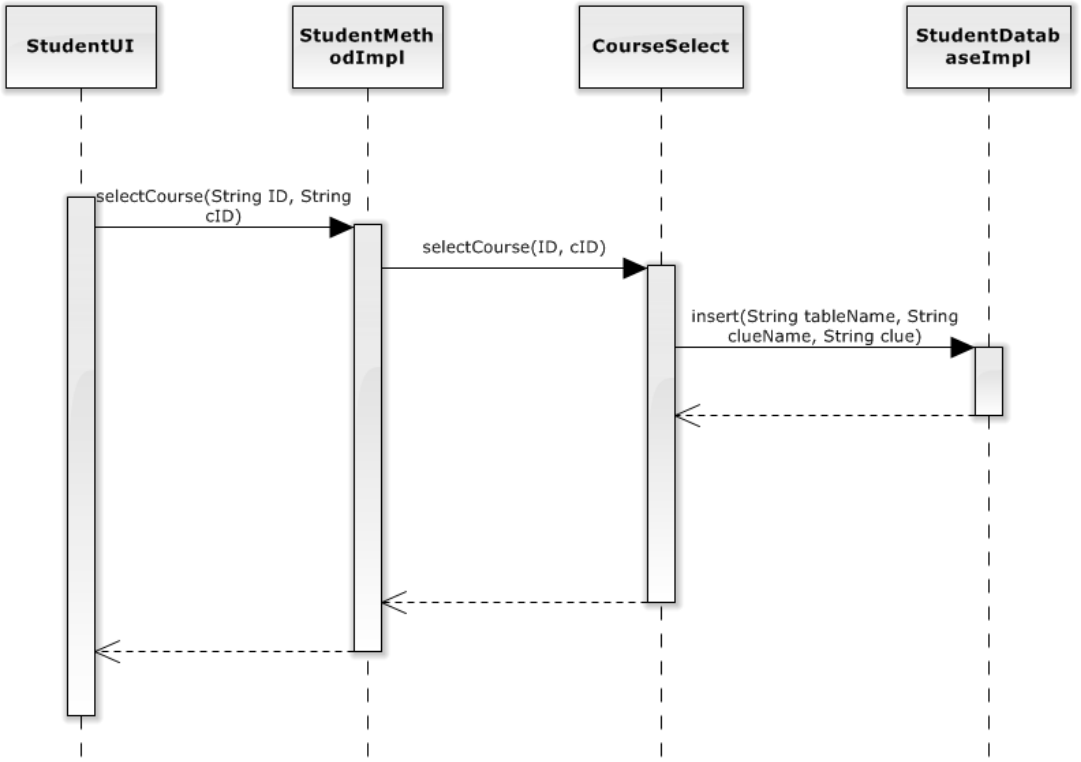

图 43 选择课程的顺序图

(5) 业务逻辑层的设计原理

利用单件模式，每个界面需要访问的业务逻辑由各自的控制器决定使用何种实现方法。

利用策略模式，StudentMethodImpl 所需实现的方法都委托到其他类来实现，提高了代码的复用性和易修改性。

5 附录

表 13

模块	职责
CourseGetter	根据课程 ID 或院系 ID 得到课程
CourseInforFiln	向课程添加 scprit 信息
CoursePublish	发布课程
CourseSelect	选择课程
CourseStudentGetter	获取某个课程的选课学生
CourseUpdata	更新课程信息
BasicFrameManagement	创建、修改、查看整体框架
DeanGetter	得到教务员
FacultyDeanGetter	得到院系教务员
FrameManagement	创建、修改、查看教学计划
StudentCourseListGetter	得到某个学生的选课列表
StudentGetter	根据学生 ID 或院系 ID 得到学生
StudentScoreGetter	根据学生 ID 和课程 ID 得到该学生在该课程的成绩
ScoreRecord	录入成绩
TeacherCourseListGetter	根据老师 ID 得到该老师的课程列表
TeacherGetter	根据老师 ID 或院系 ID 得到老师
Login	登录
PasswordChange	更改密码