



---

# 选课系统设计报告

---

复旦大学计算机科学与技术学院



雷建坤 10300240065

郭俊石 10300240031

2013-6-11

# 目录

一、系统概述 .....	2
二、领域模型 .....	3
2.1 领域模型图 .....	3
2.2 领域模型说明 .....	3
三、系统顺序图 .....	5
3.1 学期选课设置场景 .....	5
3.2 学生学期选课场景 .....	6
四、具体实现 .....	6
4.1 Hibernate 实现领域模型 .....	6
4.2 Hibernate 实现一对多映射 .....	7
4.3 Hibernate 实现多对多映射 .....	8
4.4 控制器实现业务逻辑 .....	9
五、测试 .....	9
5.1 junit 测试方法 .....	9
5.2 测试用例 .....	9
5.3 测试覆盖率 .....	10
六、开发环境和设置 .....	12

# 一、系统概述

模拟选课系统的主要功能是为某院系学生提供每学期的选课服务。

系统中含有课程库, 含有课程信息包括: 课程名称、代号(在课程库中唯一), 另外也包含了本系教师清单。教务人员在每个学期开始时从课程库中选择课程, 新建一个学期的课程列表。选择课程后再设定此课程的上课地点、选课人数上限、上课时间, 任课教师。上课时间是每周 1-5, 上午有 4 个课时, 下午有 3 个课时, 每门课可占用多个连续的课时。

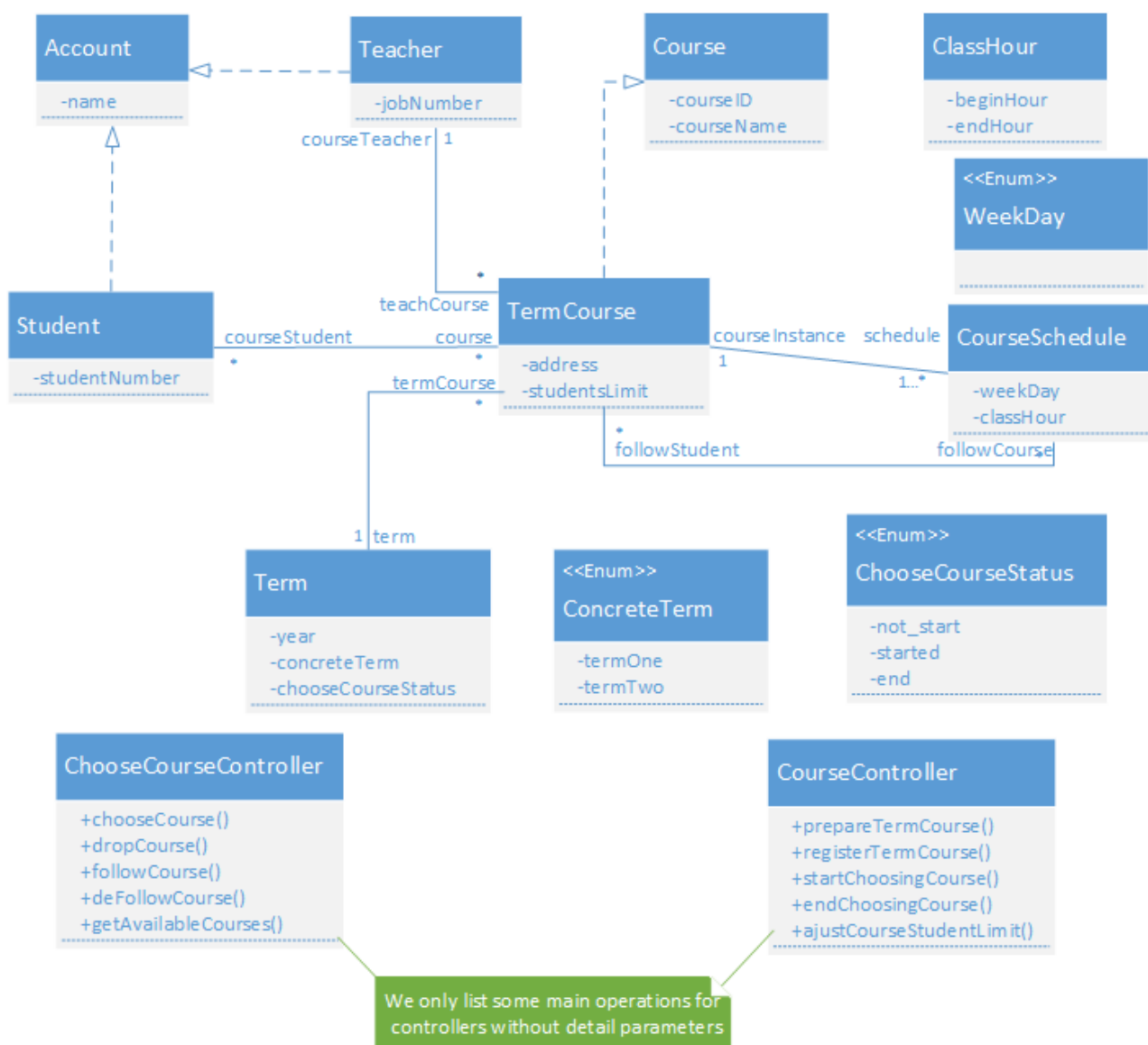
由教务人员启动学期的选课后, 学生才能开始使用此选课系统。在开始选课后, 学生可以浏览所有课程, 或根据课程名称、代号搜索课程进行选课, 但所选课程不得和已选课程的课时有重叠的部分(不区分必修和选修, 每门课都是选修)。

当一门课的选课人数达到上限时, 学生不能再选此门课程。但可能选择“开始关注”\“取消关注”此课程, 对每位关注此课程的学生, 当有空余名额时(由于退课或教务人员增加了选课人数), 系统会发送通知, 学生可在上线后阅读通知。已选上这门课程的学生不得再关注此课程, 选上课程后自动解除关注。学生可以在系统开放时随时查看自己的课表, 也能查看过去以前的选课课表。

教务人员最后可以关闭此学期的选课活动。

## 二、领域模型

### 2.1 领域模型图



### 2.2 领域模型说明

在这个领域模型中，我们总共有 13 个类。

- 主要的 Hibernate 数据库映射的类有 7 个：

**Account**（用户抽象类）、**Teacher**（老师）、**Student**（学生）、**Course**（课程抽象类）、**TermCourse**（学期课程）、**Term**（学期）、**CourseSchedule**（课程上课时间）；

- 3 个枚举类和 1 个课时类：

**ConcreteTerm**：期枚举类，如第一学期和第二学期

**ChooseCourseStatus**：选课状态枚举类，如未开始、已开始和已结束

WeekDay: 星期枚举类

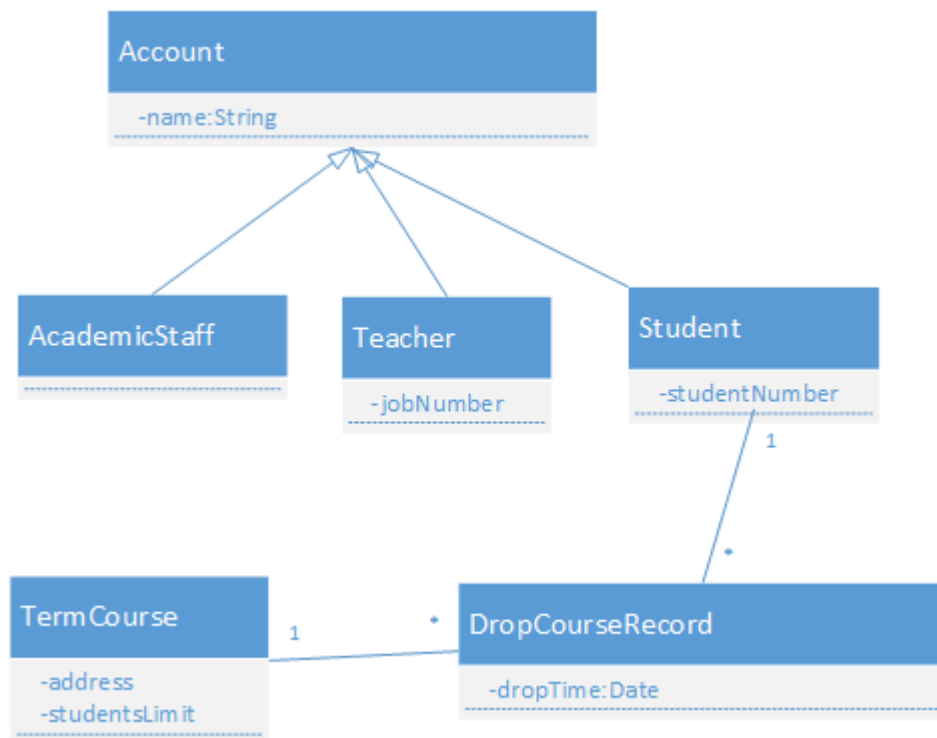
ClassHour: 课时类, 记录开始课时和结束课时

- 2 两个基本控制器类:

CourseController: 课程管理控制器

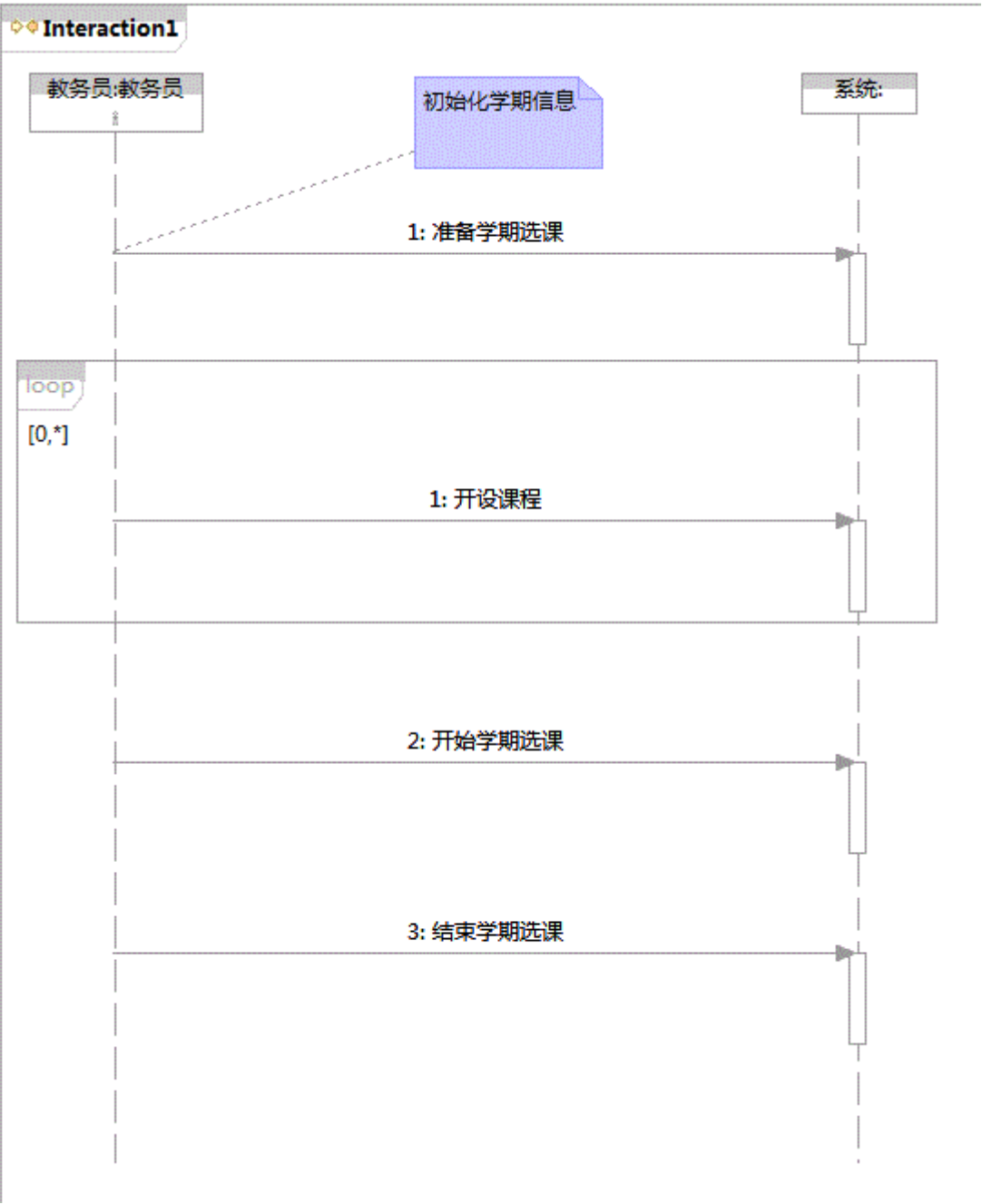
ChooseCourseController: 选课控制器

- 后来在实现中又另外添加了两个类: AcademicStaff (教务员) 和 DropCourseRecord (退课记录)

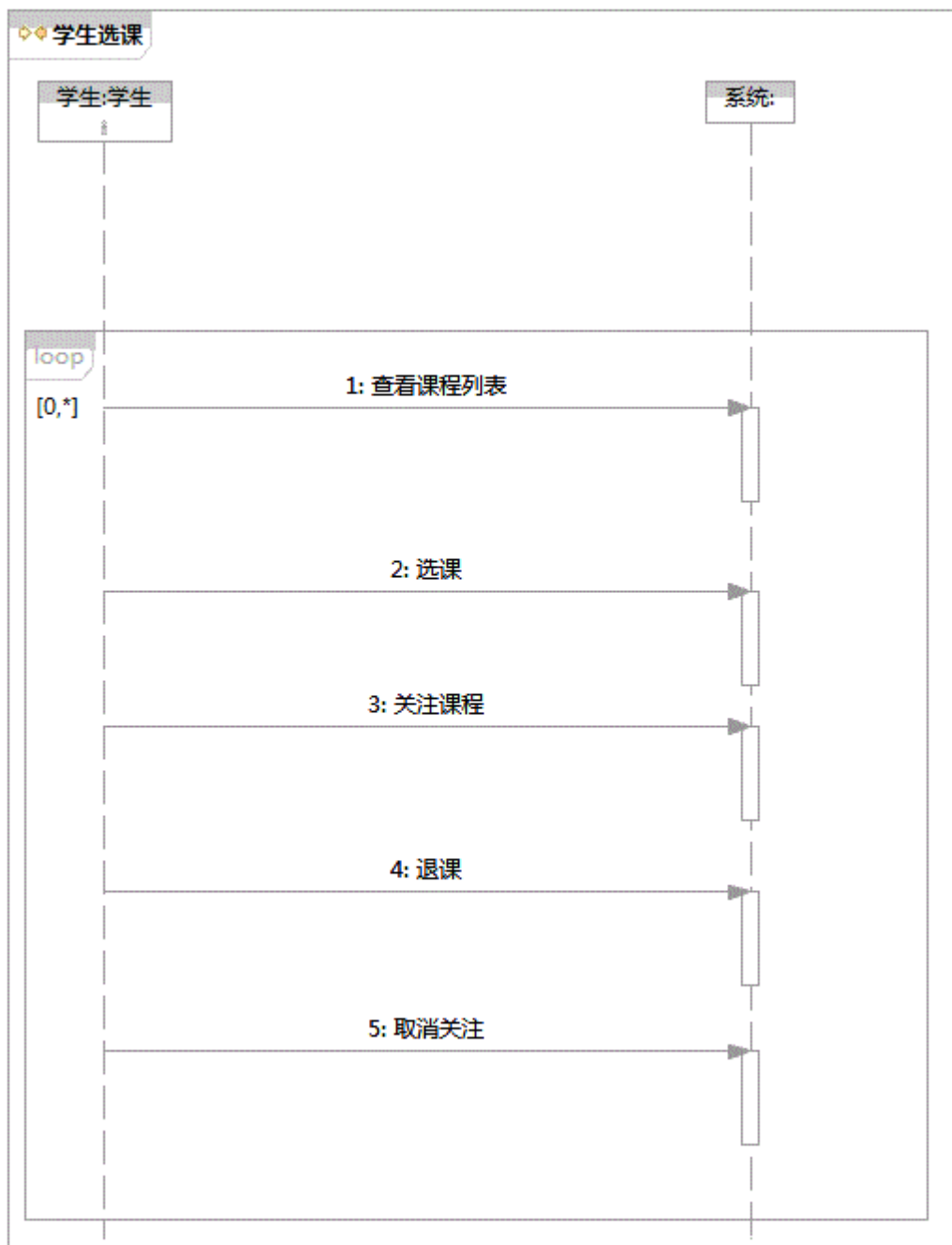


# 三、系统顺序图

## 3.1 学期选课设置场景



## 3.2 学生学期选课场景



## 四、具体实现

### 4.1 Hibernate 实现领域模型

在本实验中我们用 **Hibernate** 来实现领域模型中与数据相关的持久化存储。**Hibernate** 提供了一种面向对象的持久化存储的实现方式，我们只要建立一些与领域模型相关的类就可以通过面向对象的方法直接去操作一个对象，进行数据库的增删改查等操作。在包 `com.raysmond.hibernate` 中，我们一共建立了 **14 Java** 类来映射领域模型。

在实际的 MySQL 数据库中可以看到 Hibernate 自动建立了如下数据表：










 **xksystem**  
All tables of the xksystem schema

Table Name ▲	Engine	Rows	Data length	Index length	Update time
 account	InnoDB	0	16 kB	0 B	
 account_course	InnoDB	0	16 kB	32 kB	
 course	InnoDB	4	16 kB	32 kB	
 courseschedule	InnoDB	4	16 kB	16 kB	
 dropcoursererecord	InnoDB	0	16 kB	32 kB	
 student_choose_course	InnoDB	0	16 kB	32 kB	
 student_follow_course	InnoDB	0	16 kB	32 kB	
 term	InnoDB	2	16 kB	0 B	

## 4.2 Hibernate 实现一对多映射

在我们的领域模型中，也有这种一对多的关系，如教师和学期课程的关系。在 Hibernate 我们用 **annotation** 这种方式来实现。

例如 **Teacher** 类和 **TermCourse** 类的一对多关系：



```
@Entity
public class TermCourse extends Course {

    @ManyToOne
    private Teacher teacher;

    // other fields or methods
}

@Entity
public class Teacher extends Account {

    // term courses thought by the teacher
    @OneToMany(mappedBy = "teacher")
    private Collection<TermCourse> courses = new ArrayList<TermCourse>();

    // other fields or methods
}
```

如果想在一对多的关系加上数据库的 **CASCADE** 关系的话，也很容易，例如可以把

```
@OneToMany(mappedBy = "teacher")
```

写成 `@OneToMany(mappedBy = "course", cascade = { CascadeType.ALL })`



这样当 teacher 删除的时候，对应的所有课程也会被删除。

## 4.3 Hibernate 实现多对多映射

多对多是一种比较复杂的关系，不过利用 Hibernate 也是很容易实现的。

例如课程和学生的选课关系，多对多：



```
@Entity
public class TermCourse extends Course {

    // the students in the course
    @ManyToMany(cascade = { CascadeType.PERSIST,
        CascadeType.MERGE }, mappedBy = "choosedCourses")
    private Collection<Student> courseStudents = new ArrayList<Student>();

    // other fields and methods
}

@Entity
public class Student extends Account {

    //Courses chosen by the student
    @ManyToMany(
        targetEntity=com.raysmond.hibernate.TermCourse.class,
        cascade = {CascadeType.PERSIST, CascadeType.MERGE},
        fetch=FetchType.LAZY
    )
    @JoinTable(name = "Student_Choose_Course")
    private Collection<TermCourse> choosedCourses
        = new ArrayList<TermCourse>();
}
```

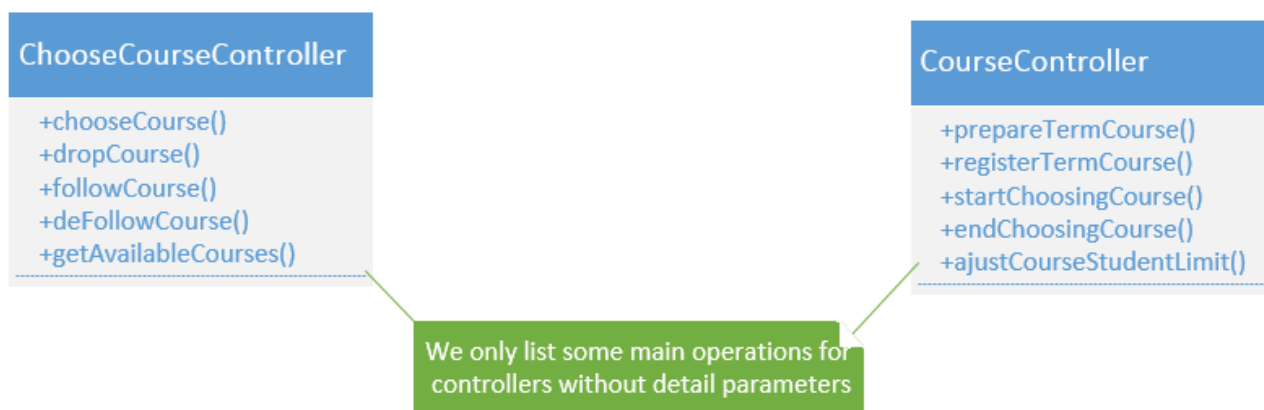
其中@JoinTable 用于指定这个多对多关系在数据库中的表名。在数据库中多对多关系是新建了一张表来表示的。如果不指定表名的话，那么这两个类的所有多对多关系都会在同一张表中表示，那么就会出现很多的冗余数据。对于上面的这个例子在数据库中对应的表如下：

Table Name:	student_choose_course	Database:	xksystem	Comment
Columns and Indices	Table Options	Advanced Options		
Column Name	Datatype	NOT NULL	AUTO INC	Flags
courseStudents_ID	INT(11)	✓		<input type="checkbox"/> UNSIGNED <input type="checkbox"/> ZEROFILL
choosedCourses_ID	INT(11)	✓		<input type="checkbox"/> UNSIGNED <input type="checkbox"/> ZEROFILL

注意：不管是一对多还是多对多的关系，在使用 Hibernate 的 session 存储时，要记得同时

把关系设计的两个对象同时存储起来。

## 4.4 控制器实现业务逻辑



我们主要有两个控制器用于实现业务逻辑相关的操作。一个课程管理控制器（**CourseController**），一个是选课控制器（**ChooseCourseController**）。

# 五、测试

## 5.1 junit 测试方法

在本实验中我们利用 **junit** 来测试我们的程序，主要是利用白盒测试的方法在测试功能正确性的同时力求较高的测试覆盖率（代码覆盖）。当然我们不可能所有方法都测一遍，一些如 **getter** 和 **setter** 类的于业务逻辑没有多大关系的方法就不需要进行测试。本实验的测试主要侧重于 **Hibernate** 数据存储和业务逻辑处理两个方面。

## 5.2 测试用例

在本实验中，我们一个有 6 个 **junit** 测试类，共有 20 个测试方法。

```

com.raysmond.test.TestTermCourse
void testConflictCourse()
void testAddNewScheduleToCourse()

```

```

com.raysmond.test.TestSimpleCreate
void testCreateTermCourse()
void testCreateTerm()
void testCreateTeacher()
void testCreateStudent()
void testCreateDropCourseRecrod()
void testCreateCourseSchedule()

```

```

com.raysmond.test.TestIChooseCourseController
IChooseCourseController controller
void testChooseCourse()
void testDropCourse()
void testGetAvailableCourses()
void testStudentFollowTermCourse()
void testStudentDefollowCourse()

```

```

com.raysmond.test.TestICourseController
ICourseController courseController
void testRegisterTermCourse()
void testStartChoosingCourse()
void testEndChoosingCourse()
void testPrepareChooseCourse()
void testAjustCourseStudentLimit()

```

```

com.raysmond.test.TestCourseSchedule
void testCourseScheduleOverlap()

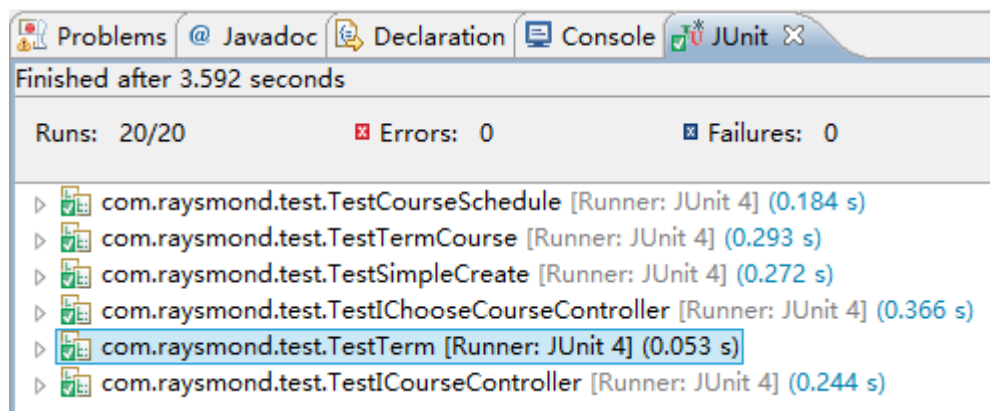
```

```

com.raysmond.test.TestTerm
ICourseController courseController
void testIsConflictTermCourse()

```

在 Eclipse 测试通过：



## 5.3 测试覆盖率

在本实验中我们用 Clover 这个 Eclipse 插件来统计测试的覆盖率。使用 Clover 插件，运行整个 junit 测试包的时候，就会生成详细的测试覆盖率分析报告。

本实验中我们的总体测试覆盖率为 85%以上，基本概况如下：

Show: All classes				
Element		Cov%	Av Me Cpx	Cpx
✚ xksystem		<div><div></div></div> 86.5%	1.3	177.0
▸ com.raysmond.hibernate		<div><div></div></div> 85.3%	1.2	89.0
▸ com.raysmond.hibernate.controller		<div><div></div></div> 84.3%	3.2	32.0
▸ com.raysmond.test		<div><div></div></div> 100.0%	1.2	25.0
▸ edu.fudan.ss.persistence.hibernate.common		<div><div></div></div> 41.7%	1.1	31.0

## Metrics for: xksystem

### Structure

Packages: 4  
Files: 29  
Classes: 29  
Methods: 133  
Statements: 514  
Branches: 72

### Test Executions

Executed Tests: 20  
Passes: 20  
Fails: 0  
Errors: 0

### Source

LOC:	2,100	NC LOC:	1,383
Total Cmp:	177	Cmp Density:	0.3
Avg Method Cmp:	1.3		




**Coverage** 22 classes, 335 / 432 elements

77.5% 

**Test Results** 20 / 20 tests 0.44 secs

100%

### Most Complex Packages

1. 85.3%  com.raysmond.hibernate (89)
2. 84.3%  com.raysmond.hibernate.controller (32)
3. 36.4%  edu.fudan.ss.persistence.hibernate.common (28)

### Most Complex Classes

1. 85.5%  TermCourse (28)
2. 81.4%  IChooseCourseControllerImpl (21)
3. 34.8%  PMHibernateImpl (19)
4. 90%  Term (14)
5. 94.6%  CourseSchedule (13)

### Top 17 Project Risks

BaseModelObject ObjectFactoryHibernateInterceptor PMHibernateImpl **IChooseCourseControllerImpl**  
ICourseControllerImpl Course TermCourse Account Term Student Teacher CourseSchedule DropCourseRecord WeekDay ConcreteTerm  
ChooseCourseStatus AcademicStaff

当然有一些没有测试到方法，分析一下，可以发现一些 **getter** 和 **setter** 方法没有覆盖到，例如 **Teacher** 类中的测试覆盖分析（如下图）。可以发现其中设置集合的一些方法我们没有覆盖到，毕竟我们很少会设置一个集合直接赋值给某一个对象，一般都是通过集合的 **add()** 方法直接把集合元素添加进去。这类方法没有测试到也无很大关系。

```

15 // term courses thought by the teacher
16 @OneToMany(mappedBy = "teacher")
17 private Collection<TermCourse> courses = new ArrayList<TermCourse>();
18
19 4 public static Teacher create(String name,String jobNumber,IPersistenceManager
20 4 Teacher teacher = new Teacher();
21 4 teacher.setName(name);
22 4 teacher.setJobNumber(jobNumber);
23 4 pm.save(teacher);
24 4 return teacher;
25 4 }
26
27 // getters and setters
28 4 public Collection<TermCourse> getCourses() {
29 4 return courses;
30 4 }
31
32 0 public void setCourses(Collection<TermCourse> courses) {
33 0 this.courses = courses;
34 0 }
35
36 0 public String getJobNumber() {
37 0 return jobNumber;
38 0 }
39
40 4 public void setJobNumber(String jobNumber) {
41 4 this.jobNumber = jobNumber;
42 4 }
43 }

```

注释：在图中有绿色背景的方法代码表示在测试中被覆盖到的，而有红色背景的方法代码则没有被覆盖到的。

## 六、开发环境和设置

操作系统：Windows 8 64bit

编程开发软件：Spring Tool Suite 3.2.0.RELEASE

数据库：MySQL5

数据库设置：

数据库名：xksystem

用户名：xksystem

密码：xksystem