

****

**面向对象的分析与设计小组报告**

|  |  |
| --- | --- |
| 课程名称 | 面向对象的分析与设计 |
| 成员姓名 | SY2121109 李阳 |
|  | ZY2121122 杨沛业 |
|  | LS2121201 卜永祺 |
|  | SY2121119 闫涛 |
|  | ZY2105113 孙逸凡 |

目录

[一、题目简介 2](#_Toc21956)

[1.1报名 2](#_Toc20450)

[1.2编排 3](#_Toc32232)

[1.3成绩录入 3](#_Toc16550)

[1.4成绩查询 3](#_Toc27691)

[二、 用户需求 3](#_Toc16337)

[2.1自然语言需求（用户故事） 3](#_Toc28139)

[2.2系统需求（EARS） 4](#_Toc9581)

[三、 UML需求模型 5](#_Toc17062)

[3.1用例图： 5](#_Toc17456)

[3.2概念类图： 5](#_Toc1700)

[3.3系统顺序图： 6](#_Toc12804)

[四、 UML设计模型 8](#_Toc16350)

[4.1类图： 8](#_Toc28109)

[4.2包图： 9](#_Toc1845)

[4.3顺序图： 9](#_Toc5203)

[4.4架构设计： 14](#_Toc22572)

# 一、题目简介

北京航空航天大学游泳比赛报名系统目前用于全校游泳比赛的报名相关工作。现有校内各项目体育比赛均为报名信息收集、人工录入的原始方式，极大地消耗了参赛队及比赛组织方的时间精力，费时费力，程序繁琐，效率较低并且容易出错。现需要一套报名系统，于系统内进行报名的相关工作，由后台自动处理相关信息，已使得比赛报名工作顺利、高效进行。需求具体如下：

#### 1.1报名

* 系统中设置报名入口，填入相关信息后提交，即可报名完成。
* 管理员可设置报名的开始、截止时间，于规定期限内可进行信息填报和修改，报名截止后则无法进行报名信息的修改。
* 以各学/书院为单位进行报名，每学/书院可注册或分配1个报名账号，系统内只允许报名账号进行报名和信息修改。
* 后台统计各代表队报名情况，并生成能够导出的报名结果。

#### 1.2编排

* 报名截止后，后台根据运动员的报名信息进行自动编排，并生成可导出的分组表。

#### 1.3成绩录入

* 赛后比赛组织方可将所有参赛成绩录入至系统内（Excel 文件上传，系统 自动读取）

#### 1.4成绩查询

* 运动员可自行至系统中查看历次比赛的成绩数据。

# 用户需求

#### 2.1自然语言需求（用户故事）

1. 作为运动员，我想登录账号以便使用更多功能。
2. 作为运动员，我想查询比赛成绩以便知道比赛结果。

（3）作为运动员，我想知道比赛的分组信息以便获取比赛信息。

（4）作为学/书院管理员，我想登录账号以便使用更多功能。

（5）作为学/书院管理员，我想录入报名信息以便完成报名。

（6）作为学/书院管理员，我想更改报名信息以便修正错误信息。

（7）作为学/书院管理员，我想知道比赛的分组信息以便获取比赛信息。

（8）作为学/书院管理员，我想查询比赛成绩以便知道比赛结果。

（9）作为管理员，我想登录账号以便使用更多功能。

（10）作为管理员，我想设置报名开始，截止时间，以便规定报名时间范围。

（11）作为管理员，我想更改报名信息以便修改错误信息。

（12）作为管理员，我想分配报名账号以便学/书院报名。

（13）作为管理员，我想知道比赛的分组信息以便获取比赛信息。

（14）作为管理员，我想查询比赛成绩以便知道比赛结果。

（15）作为赛事组织方，我想登录账号以便使用更多功能。

（16）作为赛事组织方，我想增加或修改赛事信息以便完善赛事。

（17）作为赛事组织方，我想录入比赛成绩。

（18）作为赛事组织方，我想知道比赛的分组信息以便获取比赛信息。

#### 2.2系统需求（EARS）

（1）学院/书院账号需要登录 UB

（2）学生账号需要登录 UB

（3）系统管理员账号需要登陆 UB

（4）赛事组织方账号需要登录 UB

（5）如果账号未能登录，那么登录功能将调用错误处理信息 UW

（6）学院/书院账号可以报名 UB

（7）当学院/书院账号登录进入主页面后，系统应该给出近期所有安排的赛事以供账号进行选择 EV

（8）当学院/书院账号点击某个赛事后，尝试进入赛事报名页面 EV

（9）如果学院/书院账号在尝试进入赛事报名页面后，系统应该检查该赛事的时间期限，如果超出那么应拒绝该账号进入赛事报名界面 UW

（10）当学院/书院账号停留在赛事报名页面时，可以多次提交学生报名信息 ST

（11）系统管理员账号和赛事组织方账号可以设置每个系统账号最大报名人数，也可以同时限制赛事的最大报名人数 OP

（12）当学院/书院账号提交学生的报名信息时，应该提供学生更详细的信息比如学号、姓名、性别 EV

（13）在学院/书院账号提交学生的报名信息期间，应调用错误处理程序处理异常（查看输入信息是否正确） ST

（14）当学院/书院账号停留在赛事报名页面时，可以修改学生的报名信息 ST

（15）学院/书院账号可以保存报名信息并退出系统 UB

（16）学生账号和学院/书院账号在可以查看某个赛事的报名信息，系统管理员账号和赛事组织方账号可以对报名信息进行查看以及修改 UB

（17）系统管理员账号和赛事组织方账号可以在主页面增加新的比赛赛事或者删除赛事UB

（18）系统管理员和赛事组织方账号在增加了新的比赛赛事后，需要给出赛事更详细的描述，如赛事名称、赛事开始结束时间、赛事限制人数等等 EV

（19）系统管理员和赛事组织方账号可以修改赛事的详细信息 EV

（20）在报名结束后，系统可以对报名信息进行自动编排，并且生成可导出的分组表 EV

（21）所有账号都可以可以查看某个赛事导出的分组表 UB

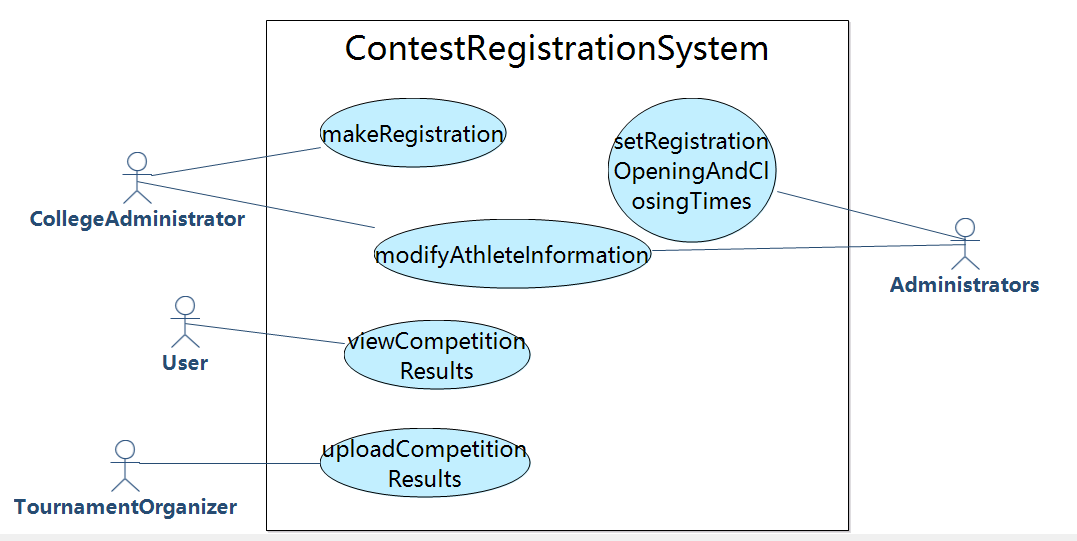
（22）赛事组织方可以将赛事结果录入系统内，该信息可以在对应的赛事界面中查看 UB

（23）所有账号在进入某个赛事后可以查看比赛的结果 UB

（24）学生账号可以查看自己历次比赛的成绩数据 UB

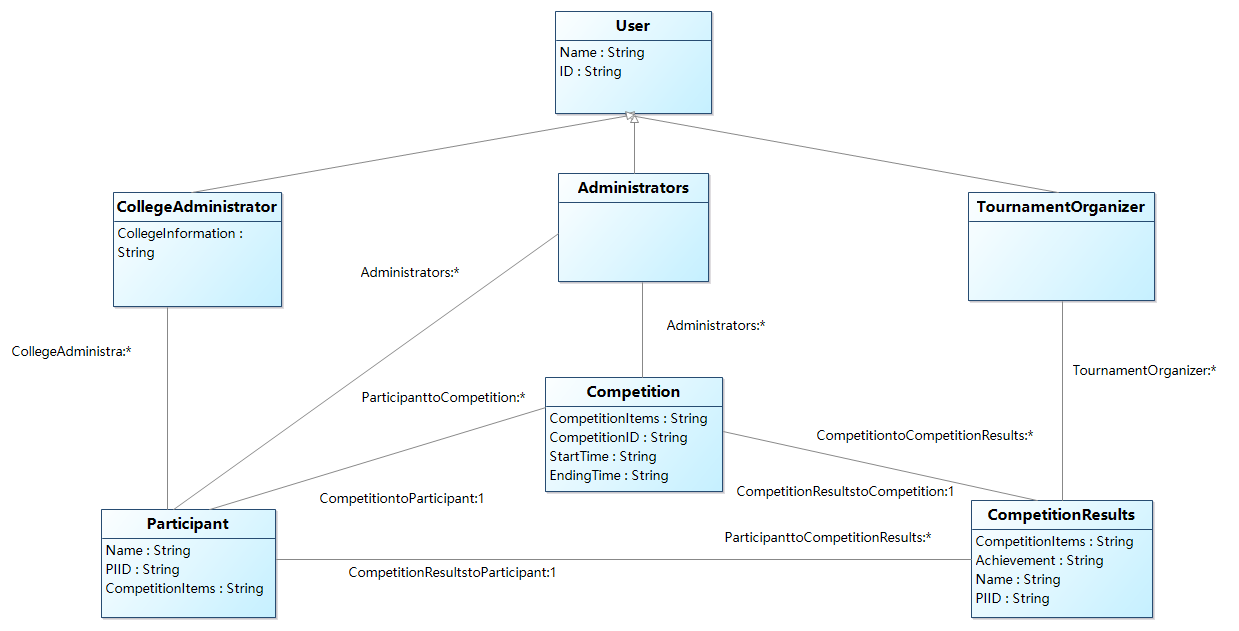
# UML需求模型

## 3.1用例图

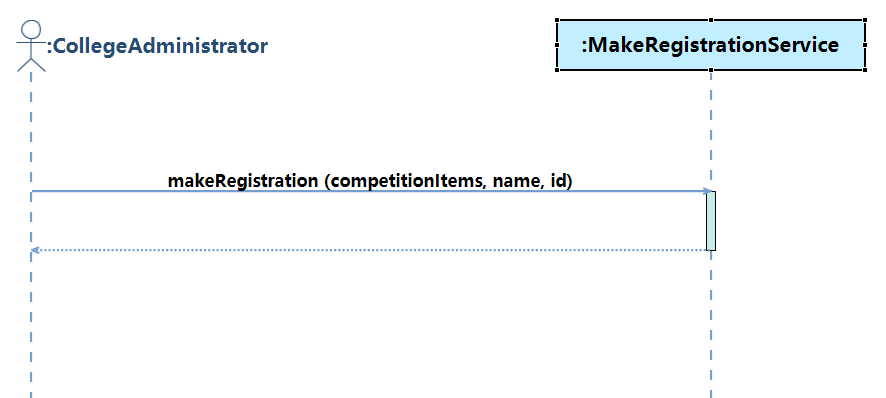


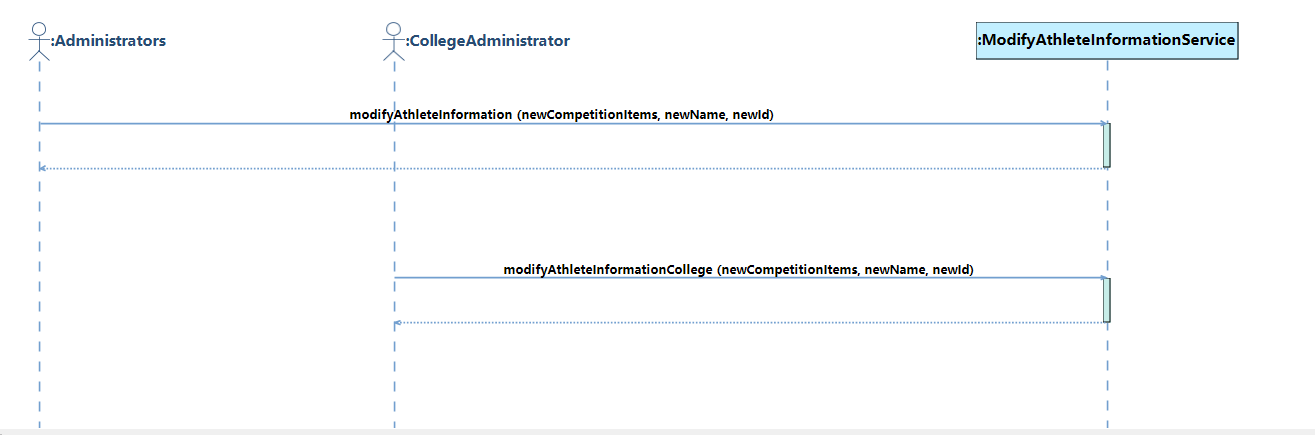
共有5个Actor。普通用户（User）只具有查看比赛结果的功能。学院管理者（CollegeAdministrator）能注册运动员和修改运动员信息。管理者（Administrator）能设置赛事注册时间和设置运动员信息。赛事组织方（TournamentOrganizer）能上传比赛成绩。

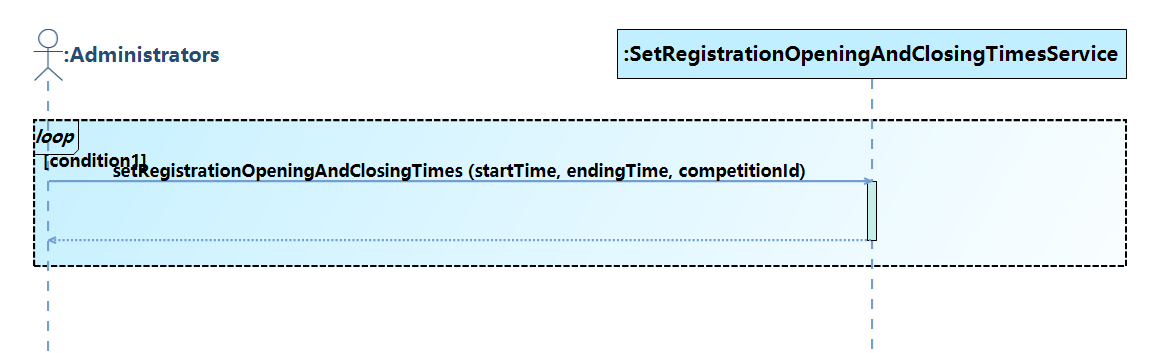
## 3.2概念类图

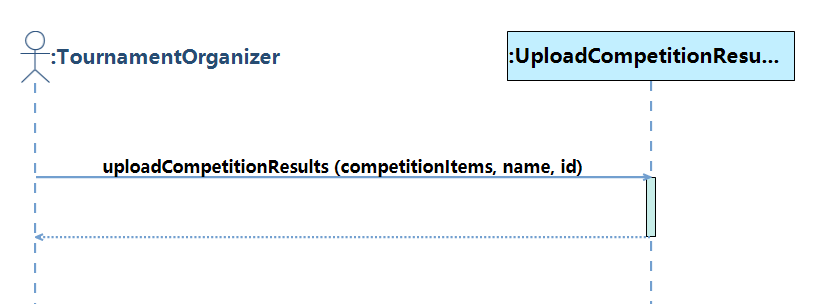


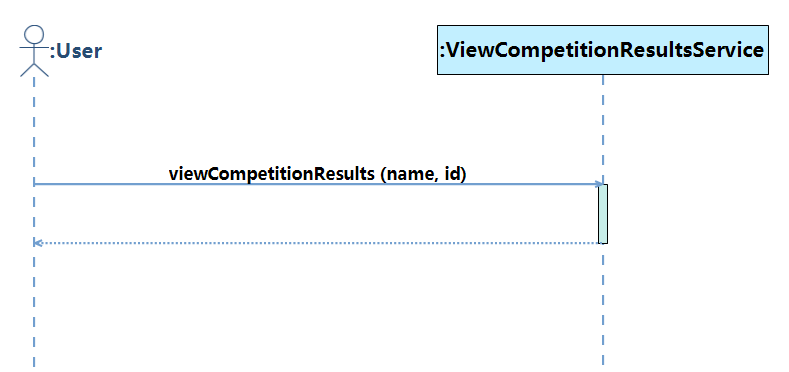
## 3.3系统顺序图





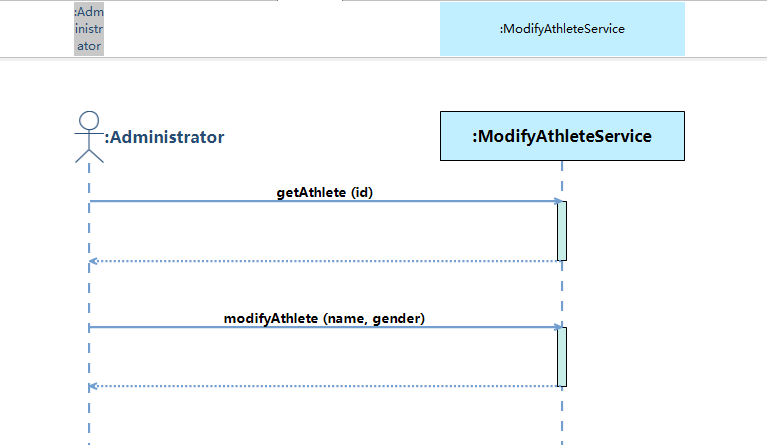


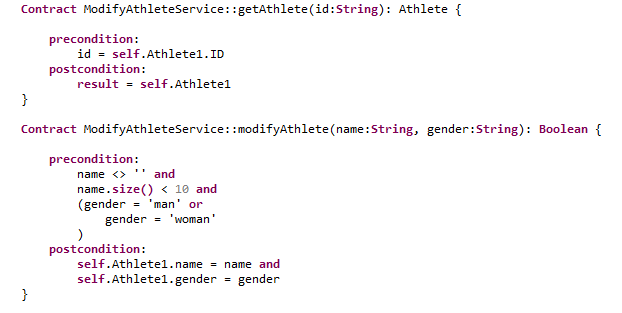




## 3.4 OCL

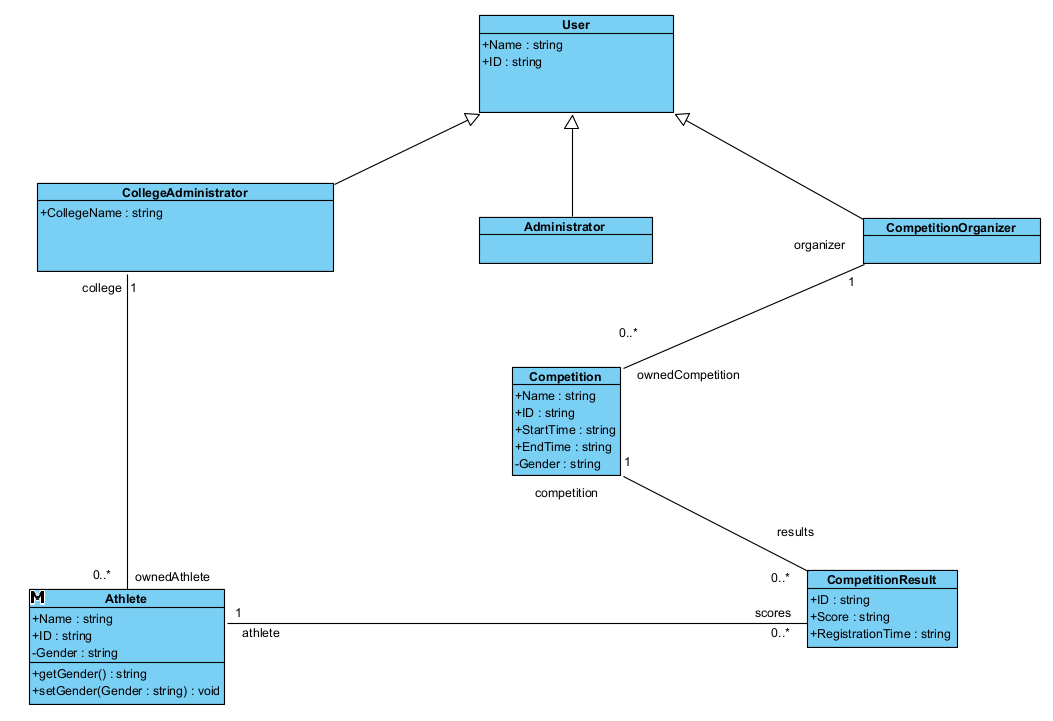
管理员先发送运动员ID给Service，Service查询并返回该运动员。管理员接着对运动员的属性进行修改，将修改后的信息发送给Service。Service先对修改后的信息进行检查。name属性不能为空，长度不能超过10；gender属性只能为man或woman。检查通过后对Athlete类对象进行属性修改。





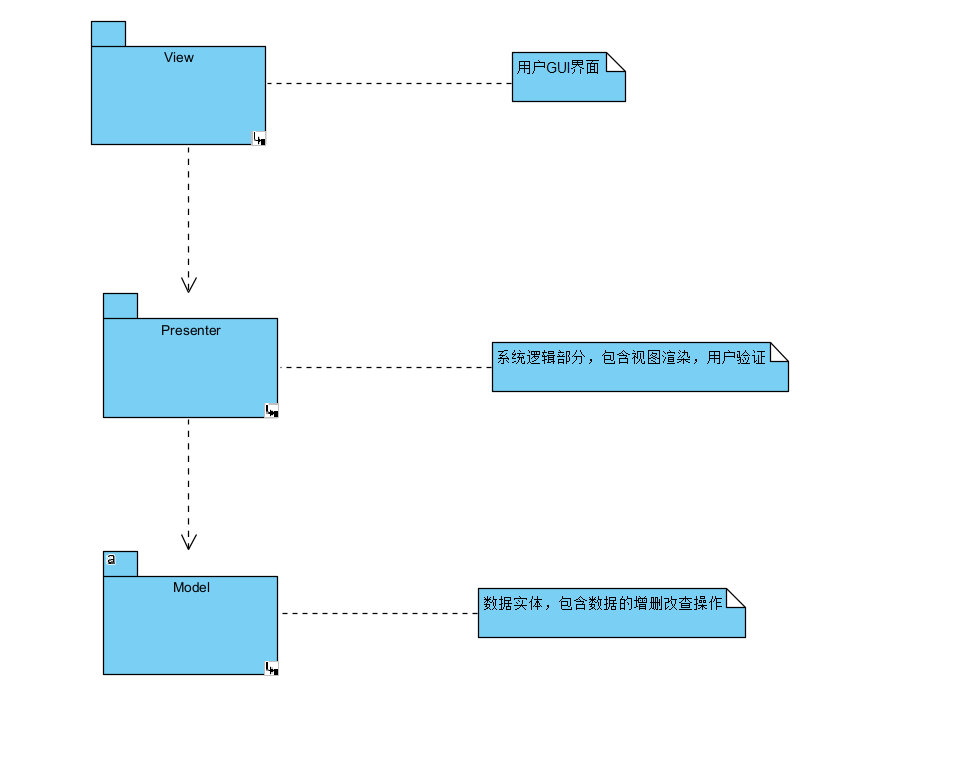
# UML设计模型

## 4.1类图



因为所有角色均能查看比赛成绩，所以将所有角色设置为User的子类，拥有Name和ID两个属性。运动员具有name，id和gender属性，运动员的学院从属信息在他与CollegeAdministrator之间的关联上，他参与的比赛、比赛的成绩 和注册比赛时间等信息被记录在CompetitionResult的类上。当运动员参加某项赛事，这个类就会被创建并与对应运动员和赛事相连。赛事类（Competition）包含了赛事自身的信息，其赛事主办方信息由它与赛事主办方类的关联提供，参与赛事的运动员由它与CompetitionResult之间的关联提供。

## 4.2包图

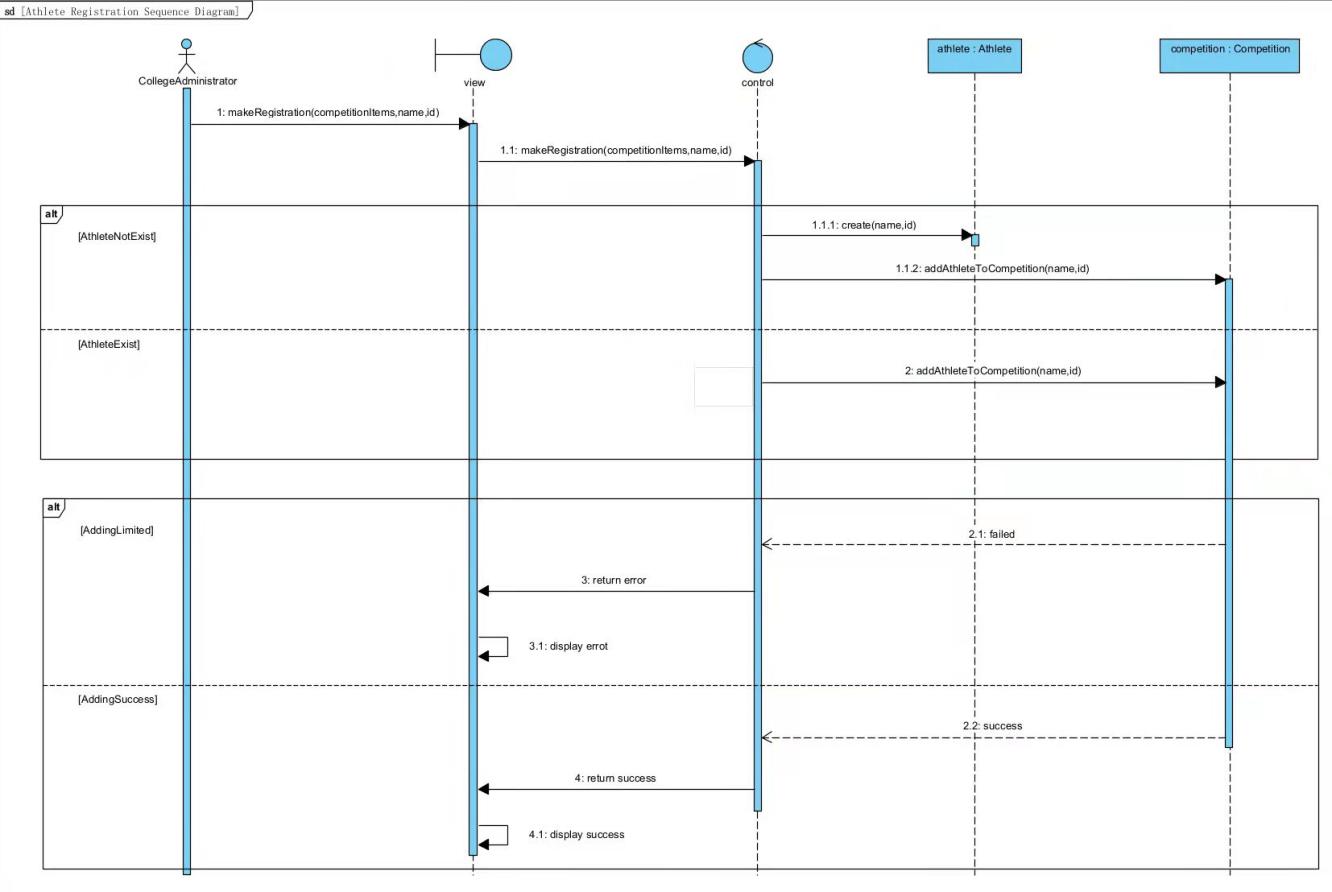


本系统架构采用了MVP架构。最上层的View负责显示，中间的Presenter负责逻辑处理，Model提供数据。

此架构模型与视图完全分离，我们可以修改视图而不影响模型。可以更高效地使用模型，因为所有的交互都发生在Presenter内部。我们可以将一个Presenter用于多个视图，而不需要改变Presenter的逻辑。

## 4.3顺序图

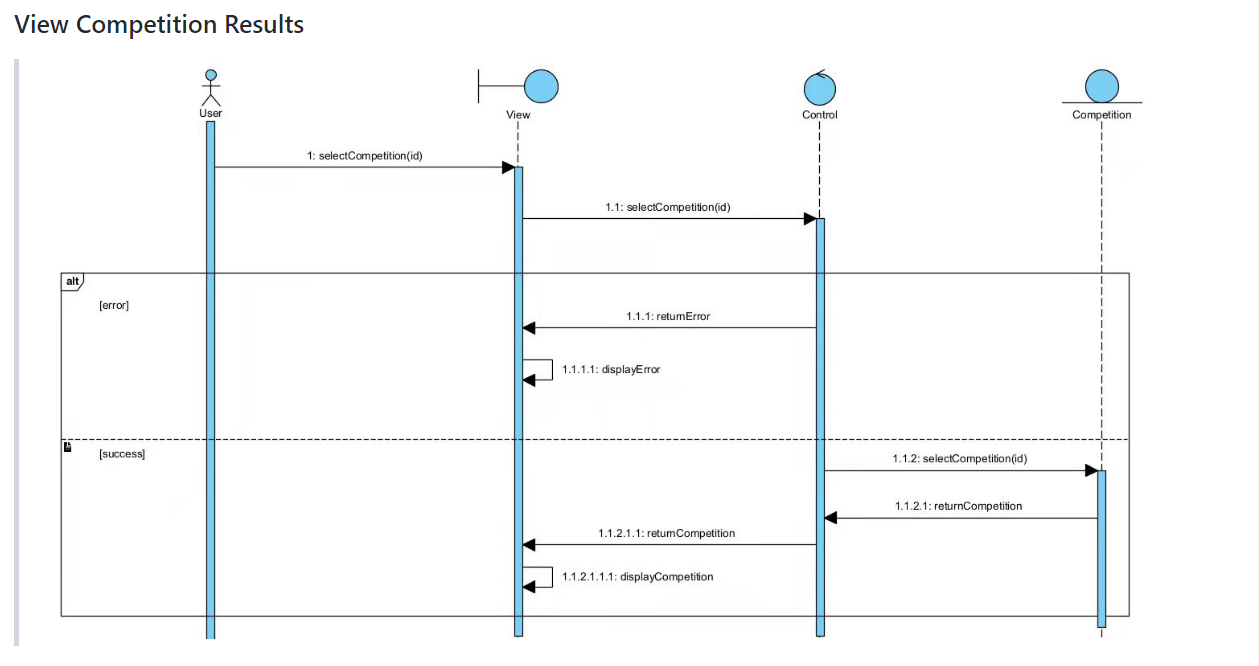
Athlete Registration：



北京航空航天大学游泳比赛报名系统中，提出了学院管理员(College Administrator)可以为参赛运动员注册比赛的功能。

该功能需要学院管理员在前端页面输入比赛名称（competitionItems）、运动员名称（name）、运动员编号信息（id）。之后该信息将会被传递到control层，系统首先判断待登记比赛的运动员是否注册。如果没有注册的话需要在Model层通过运动员名称、编号新建运动员信息。在新建了运动员信息之后就要进行运动员与赛事的绑定。由于有多种原因会导致该过程的失败（比赛报名人数超过上限、比赛不存在……），因此应该对应拥有报名成功、报名失败两种结果。该信息由Model层传向Control层，最终由Control层将报名结果传递到用户界面上。至此，一名运动员的报名流程结束。如果需要继续运动员报名流程那么重复执行该过程。

View Competition Results：

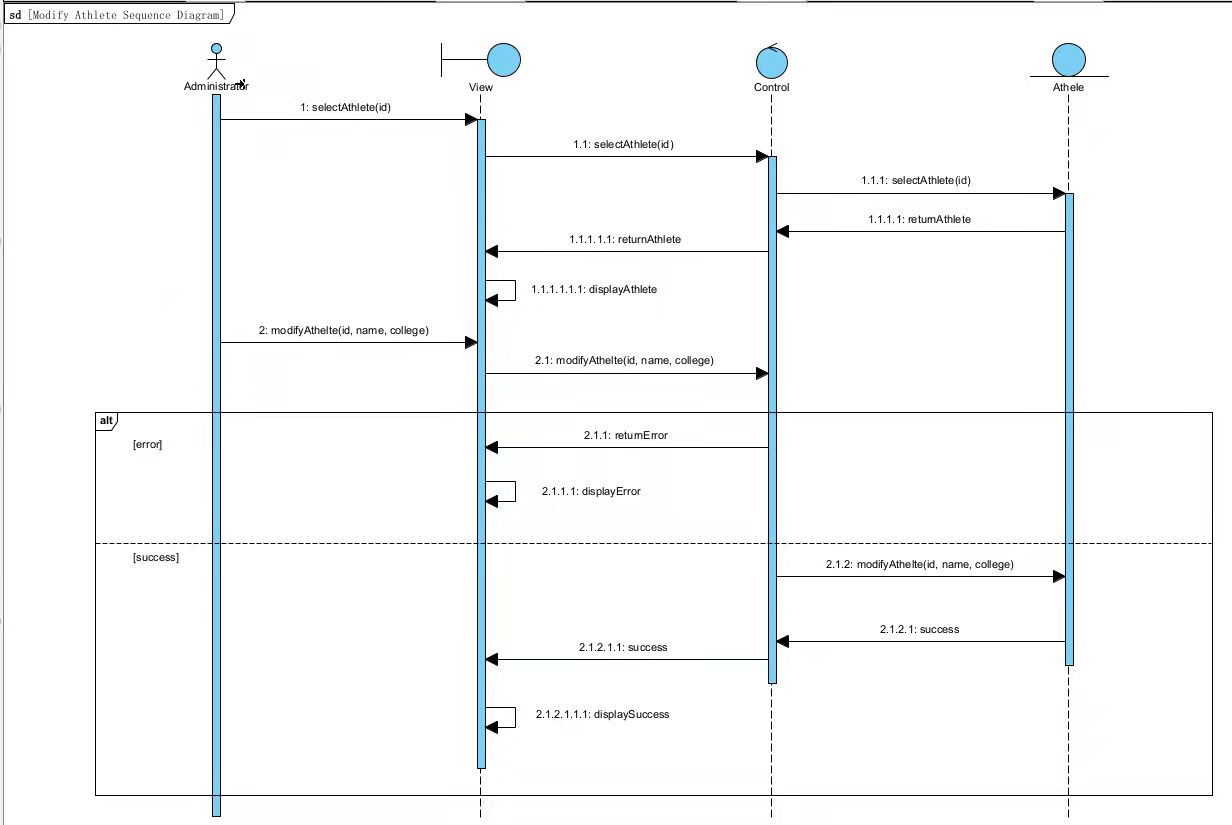


1. 在北京航空航天大学游泳比赛报名系统背景说明中，提出了对运动员可以浏览比赛结果的具体需求：运动员可自行至系统中查看历次比赛的成绩数据
2. 在用户故事中，对此需求做了进一步的提炼和概况：

作为运动员，我想查询比赛成绩以便知道比赛结果。

1. 在运动员查看比赛结果的顺序图中，首先，运动员在视图中选定想要查看的比赛，每场比赛由唯一的id标记，视图将标记该比赛的id发送给后端控制器。若该比赛还未开始或比赛成绩还未录入，控制器向视图发送信息表示请求错误，视图向运动员展示错误信息。若控制器接收到的请求合法，控制器会根据比赛id向数据存储服务器请求该比赛的结果信息，数据存储服务器向控制器返回比赛结果信息，控制器向视图返回比赛结果信息，视图向运动员展示比赛结果。

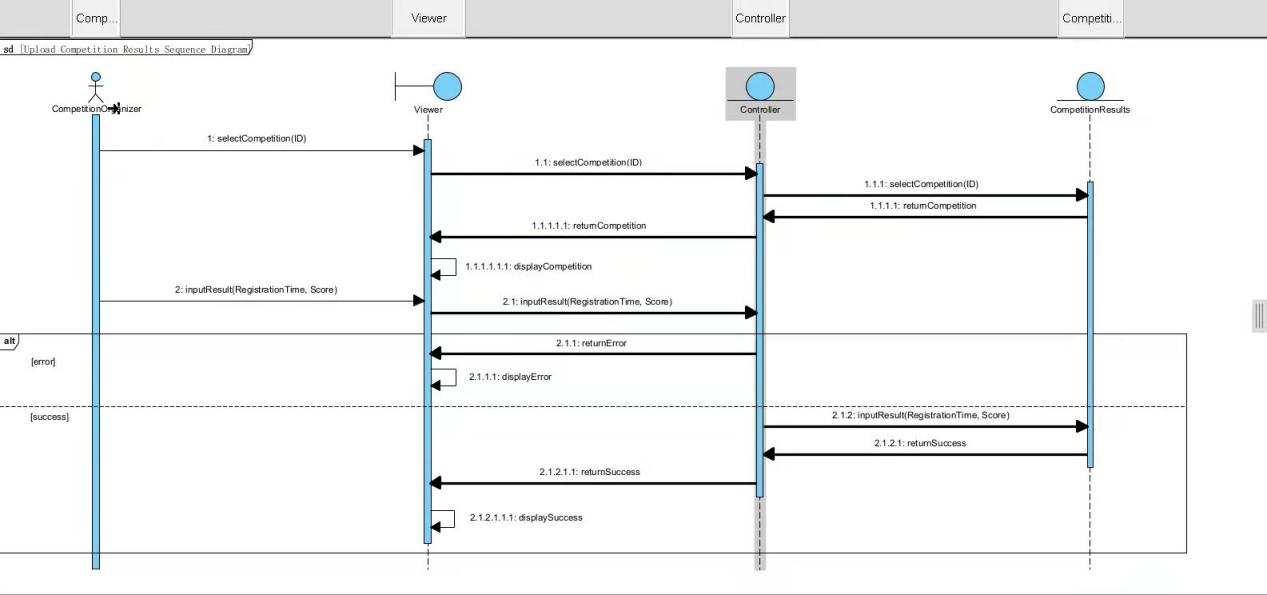
Modify Athlete：



修改运动员信息用例下的顺序图。管理员现在View界面选择需要修改的运动员，View将运动员id发送给Control，Control向数据库访问运动员信息，将信息发送给View，View接着显示此运动员的信息。

管理员在界面中修改好运动员信息后，发送修改后的运动员信息给Control，Control对修改后的信息判断是否规范，如果格式有问题，返回错误给View，View显示该错误。如果信息无误，则发送给数据库，数据库进行修改并发送成功修改信息，Control将此信息发送给View，View显示信息已经成功修改。

Upload Competition Results：



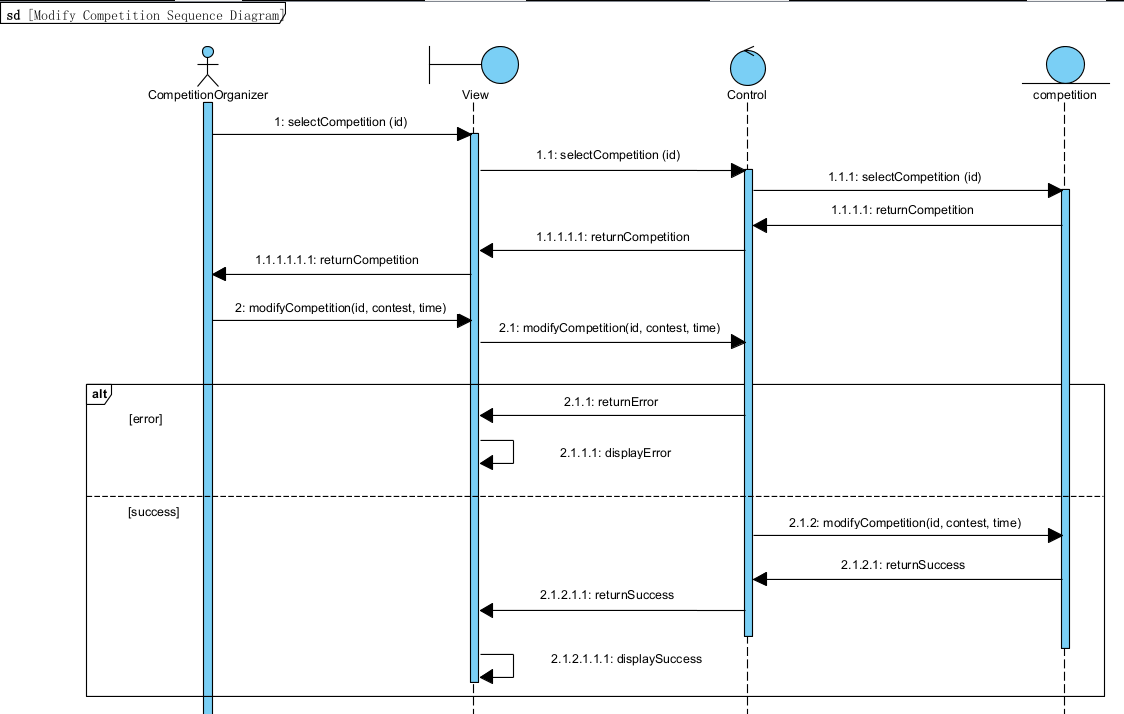
根据北航体育赛事报名系统的系统用例图，赛事组织者（CompetitionOrganizer）可以输入赛事结果(CompetitionResults)，包括赛事的ID等信息。

根据MVC框架构建了这个顺序图的四个节点：CompetitionOrganizer（赛会组织者）、用户界面（Viwer）、控制器（Controller）和比赛结果（CompetitionResults）。四个节点和之间的组织关系共同组成了业务模型（Model）。

首先赛事组织者（CompetitionOrganizer）先要选择输入哪场比赛的结果，将选择赛事信息（selectCompetition）通过ID传递给用户界面（Viwer），再传递到控制器（Controller），继续传递到比赛结果（CompetitionResults）。这时选择比赛成功，模型将选中的赛事返回（returnCompetition）到控制器（Controller），再返回到用户界面（Viwer），最后显示出来（displayCompetition）。

选择赛事成功后，赛事组织者（CompetitionOrganizer）会输入登记信息（inputResult），包括登记时间（RegistrationTime）和分数（Score）传递给用户界面（Viwer），再传递到控制器（Controller）。如果登记结果失败，则控制器（Controller）会返回等级出现错误信息（returnError）到控制器（Controller），再返回到用户界面（Viwer），最后显示出来（displayError）；如果登记结果成功，则信息会继续传递到比赛结果（CompetitionResults）。这时登记比赛结果成功，模型将成功信息返回（returnSuccess）到控制器（Controller），再返回到用户界面（Viwer），最后显示出来（displaySuccess）。

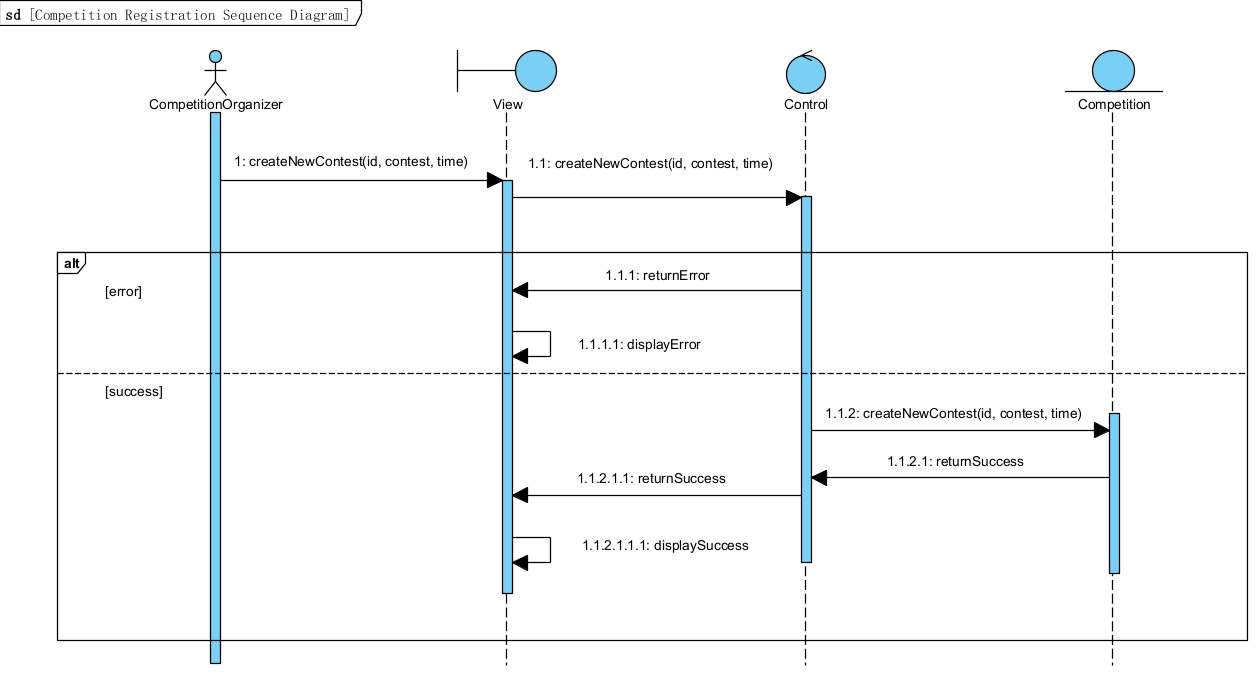
Modify Competition：



在北京航空航天大学游泳比赛报名系统背景说明中，提出管理员可设置报名的开始、截止时间，于规定期限内可进行信息填报和修改。作为管理员（CompetitionOrganizer），首先管理员可以通过赛事的（ID）查询当前的所有赛事，并选择是否对其进行修改，如果选择进行修改，需要输入赛事（ID）、赛事类型（contest）、赛事时间（time）。

若该比赛报名时间以结束，则无法进行报名信息的修改，控制器向视图发送信息表示请求错误，视图向管理员展示错误信息。若控制器接收到的请求合法，控制器会根据输入的信息对比赛信息进行修改，控制器向视图发送信息表示修改成功，视图向管理员展示修改成功。

Competition Registration：

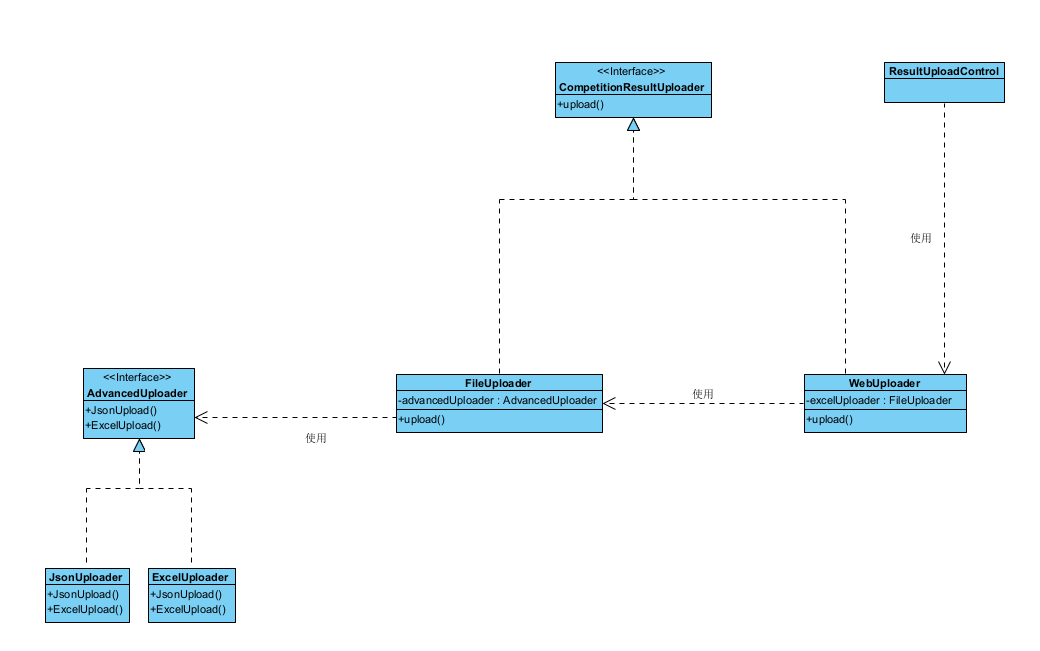


在北京航空航天大学游泳比赛报名系统背景说明中，提出管理员可设置报名的开始、截止时间。作为管理员（CompetitionOrganizer），管理员可以创建新的赛事，需要输入赛事（ID）、赛事类型（contest）、赛事时间（time）。

若该比赛报名信息不合理或者ID已被占用以，则无法创建新的赛事，控制器向视图发送信息表示请求错误，视图向管理员展示错误信息。若控制器接收到的请求合法，控制器会根据输入的信息对比赛进行创建，控制器向视图发送信息表示创建成功，视图向管理员展示创建成功。

## 4.4架构设计

适配器模式（Adapter Pattern）：



系统上传比赛成绩时，能手动在网页上录入，也能进行文件导入成绩。本系统使用了适配器模式对成绩导入功能进行扩展。WebUploader为原有的网页上传功能，它含有FileUploader的引用对其文件导入功能进行调用。文件导入类（FileUploader）调用了AdvanceUploader接口，此接口实现了两种格式的文件导入（Excel和Json），根据用户上传的文件格式，调用不同的接口实现。

过滤器模式（Filter Pattern）：

