电影评论系统设计

版本 2.0

小组成员：杨绮明，张依帆，张婷婷，敬效明，徐先荣

日期	修订号	描述
07-11	V1.0	问题陈述
07-12	V1.1	用例图、用例规约、活动图
07-12	V1.2	补充规约、术语表、系统架构设计、改进用例规约
07-12	V1.3	系统用例分析
07-13	V1.4	子系统设计、运行时部件设计
07-14	V2.0	系统分析部分细节修改完善
07-15	V3.0	软件设计技术，完善文档

目录

1.电影评论系统需求分析

1.1 问题陈述

越来越多的人会选择在闲暇时间去电影院观看电影来放松自己。但是我们如何才能找到适合自己口味的电影呢？为了方便广大影迷了解其他影迷对某部影片的感受，现开发一个电影评论系统。该系统不但可以查询相关电影信息，看到豆瓣、烂番茄、IMDB 等网站用户对该电影的点评，还可以在系统上评论、发帖与众多影迷一起讨论交流心得体会。

该系统设定了两个角色：管理员和用户。规定只有管理员和注册用户才可以登录。

管理员的主要功能是管理电影信息。在管理员执行职能前首先要登录系统，获得权限。管理电影信息首要工作就是定时更新上传新的电影的信息（导演、评分、主演、上映时间、剧情简介等），提交时系统会自动检测这些电影必要信息，检查有无重复或者疏漏。管理员也需要挑选一些好的评论放置在首页，以便让用户了解大部分网友的观点。

用户的功能是注册、登录、管理个人信息、管理自身评论。用户第一次使用本系统时，系统会自动提示用户进行注册。填写个人信息时，用户提交时系统会自动检测用户名的唯一性以及密码的规范与安全等级。登录后，用户进入主页，系统会在主页显示所有电影的封面以及评分等级；当用户想深入了解时，可点击该电影条目跳转至详细信息页面，系统会显示电影相关信息（导演、评分、主演、上映时间、剧情简介等）以及来自本系统注册用户的评论。当用户已观影完毕，用户登录后便可以在影片信息下方对该影片进行评论，给出自己心中的评级。

1.2 电影评论系统用例图

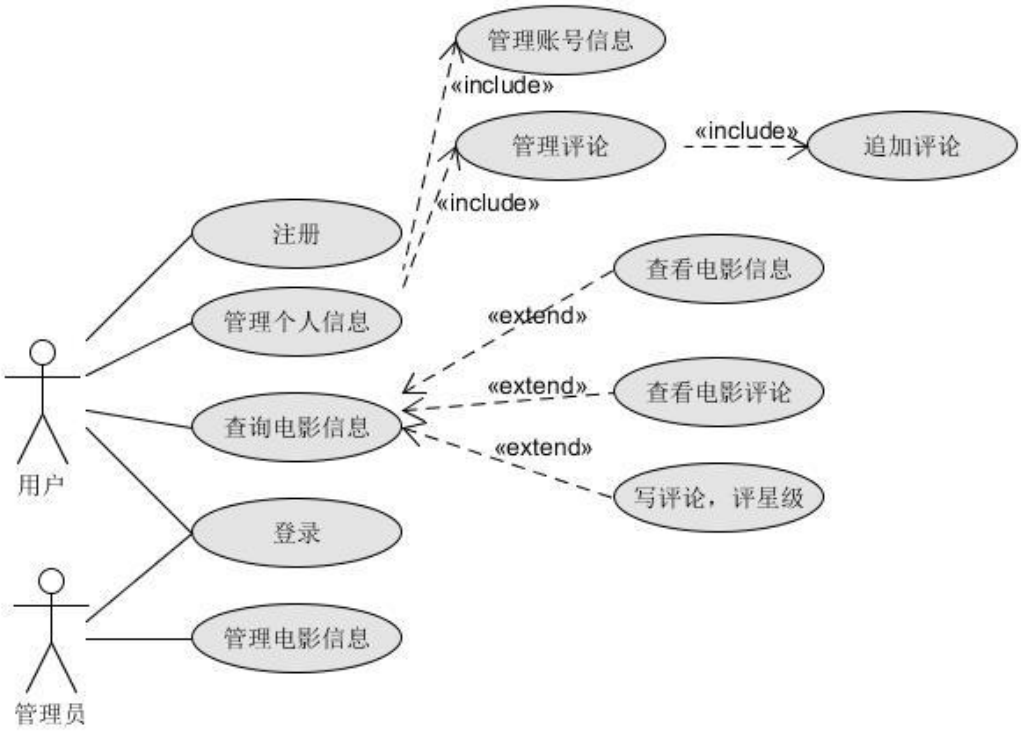

图 1：电影评论系统用例图

1.3 电影评论系统用例规约

用户注册

(1) 简要说明

本用例描述新用户如何注册到电影评论系统。

用户名合法性定义：用户名必须只能包含字母和数字并且长度超过 3 个字符。

密码合法性定义：密码的长度必须超过 6 个字符。

(2) 参与者

新用户

(3) 事件流

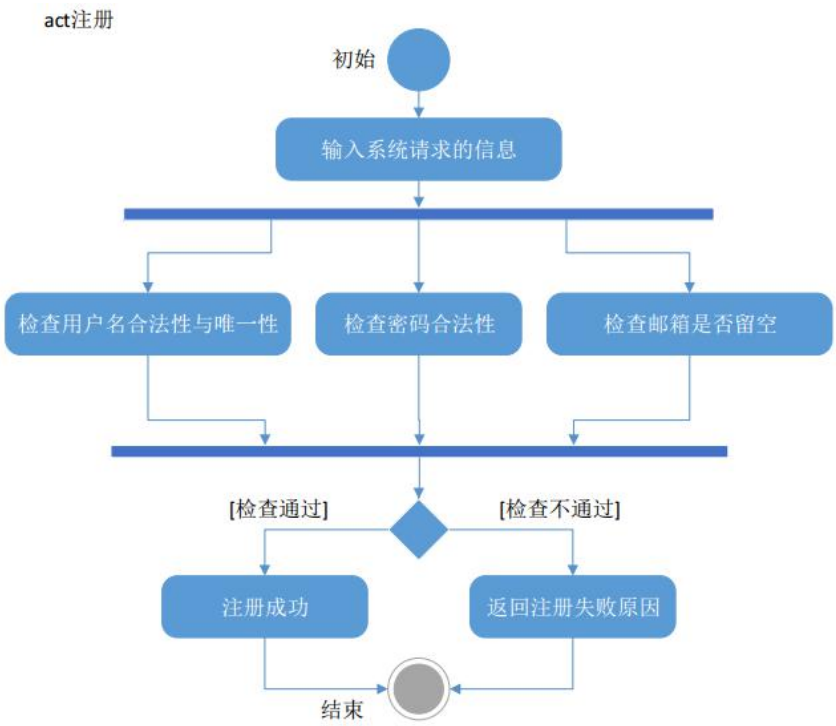

图 2：用户注册活动图

1) 基本事件流

本用例开始于新用户希望注册到电影评论系统。

Ⅰ系统请求注册用户输入用户名；

A1：用户名不合法

A2：用户名已存在

Ⅱ系统请求注册用户输入两次密码；

A3：密码不合法

A4：两次密码不一样

Ⅲ系统请求用户选择性别，输入电子邮件；

Ⅳ 系统把当前用户的信息增加到数据库中。

2) 后备事件流

A1. 用户名不合法

- ✧ 系统显示“用户名不合法”错误信息；
- ✧ 返回基本事件流第 i 步。

A2. 用户名已存在

- ✧ 系统显示“用户名已存在”错误信息；
- ✧ 返回基本事件流第 I 步。

A3. 密码不合法

- ✧ 系统显示“密码不合法”错误信息；
- ✧ 返回基本事件流第 II 步。

A4. 两次密码不一样

- ✧ 系统显示“两次密码不一样”错误信息；
- ✧ 返回基本事件流第 II 步。

(4) 特殊需求

密码输入框必须以密文方式呈现。

性别选择默认为“男”。

所有个人信息均不能留空。

(5) 前置条件

本用例开始前用户打开系统注册界面。

(6) 后置条件

如果用例成功，用户将注册成功，系统在数据库中增加一条注册用户的相关记录。

若失败，系统状态不改变。

用户登录

(1) 简要说明

本用例描述注册用户/管理员如何登录到电影评论系统。

(2) 参与者

注册用户、管理员

(3) 事件流

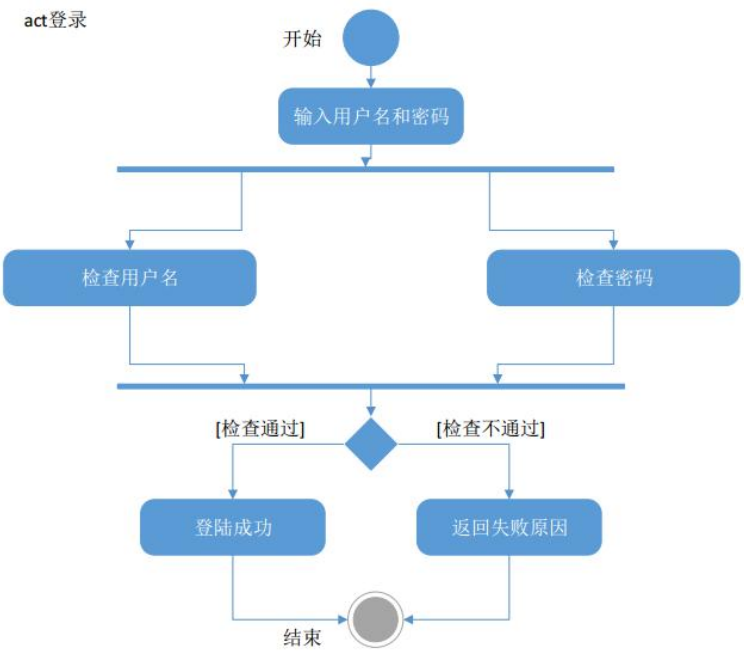

图 3：用户登录活动图

1) 基本事件流

本用例开始于注册用户/管理员希望登录到电影评论系统。

Ⅰ系统请求注册用户/管理员输入用户名和密码；

Ⅱ系统验证输入的用户名和密码；

A1：用户名不存在

A2：用户名对应密码不正确

Ⅲ注册用户/管理员成功登录到主界面，进行其他操作。

2) 后备事件流

A1. 用户名不存在

- ✧ 系统显示“用户名不存在”错误信息；
- ✧ 返回基本事件流第 I 步，且系统将提示用户注册。

A2. 用户名对应密码不正确

- ✧ 系统显示“密码不正确”错误信息；
- ✧ 返回基本事件流第 I 步。

(4) 特殊需求

密码输入框必须以密文方式呈现。

(5) 前置条件

本用例开始前用户/管理员已经打开对应的系统登录界面。

(6) 后置条件

如果用例成功，注册用户/管理员将成功登录系统，并赋予相应权限。若失败，系统状态不改变。

用户管理个人信息

(1) 简要说明

本用例允许注册用户对个人信息进行管理，提供修改密码、邮箱、电话的功能，但不允许修改用户名；可以查看用户个人已发表的评论或选择追加评论（包括修改星级）。

评论合法性定义：评论的长度不少于 10 个字符。

密码合法性定义：密码的长度必须超过 6 个字符。

(2) 参与者

注册用户

(3) 事件流

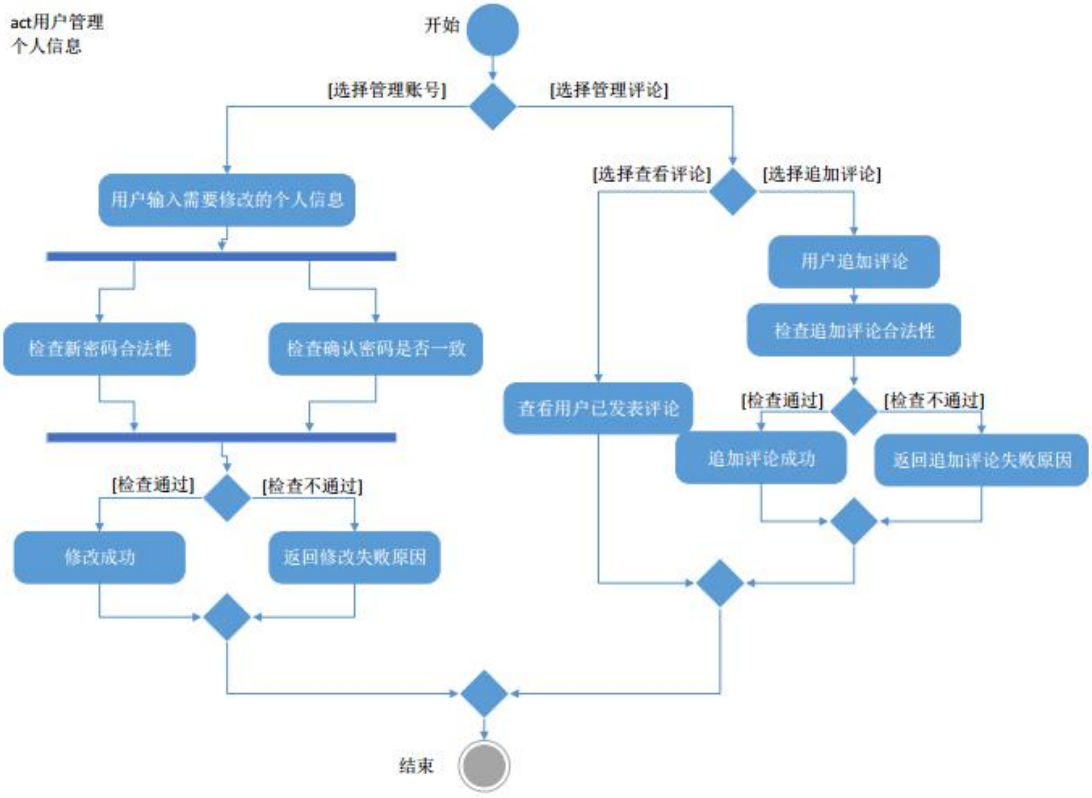

图 4：用户管理个人信息活动图

1) 基本事件流

本用例开始于注册用户希望查看或修改个人信息。

Ⅰ注册用户选择管理个人信息，则以下的其中一条子事件流将被执行：

如果选择的是“管理账户信息”，管理账号信息子事件流将被执行；

如果选择的是“管理评论”，管理评论子事件流将被执行。

Ⅰ.1 管理账号信息

a) 系统显示注册用户个人信息，包括用户名、密码、性别、电子邮箱及头像；

b) 注册用户选择需要修改的信息进行修改；

A1：新密码不合法

A2：确认密码与新密码不一致

c) 注册用户保存修改后的账户信息。

Ⅰ.2 管理评论

a) 系统显示注册用户过去曾发表过的电影评论；

b) 注册用户选择需要修改的评论进行追加并发表；

A3：追加评论不合法

c) 注册用户选择修改电影评级；

Ⅱ系统把当前用户的新信息（包括个人信息及评论信息）增加到数据库中。

2) 后备事件流

A1. 新密码不合法

- ✧ 系统显示“密码不合法”错误信息；
- ✧ 返回基本事件流第 I.1 步。

A2. 确认密码与新密码不一致

- ✧ 系统显示“两次密码不一样”错误信息；
- ✧ 返回基本事件流第 I.1 步。

A3. 追加评论不合法

- ✧ 系统显示“追加评论不合法”错误信息；
- ✧ 返回基本事件流第 I.2 步。

(4) 特殊需求

密码输入框必须以密文方式呈现。

用户名及电子邮箱注册后均不能修改。

(5) 前置条件

本用例开始前注册用户已经登录系统，并选择“管理个人信息”功能。

(6) 后置条件

如果用例成功，注册用户将成功修改个人信息。若失败，系统状态不改变。

用户查询电影信息

(1) 简要说明

本用例允许注册用户查询电影信息（包括电影介绍、原有评论及星级信息），并根据个人需要选择是否发表评论。

评论合法性定义：评论的长度不少于 10 个字符，星级不能为空。

(2) 参与者

注册用户

(3) 事件流

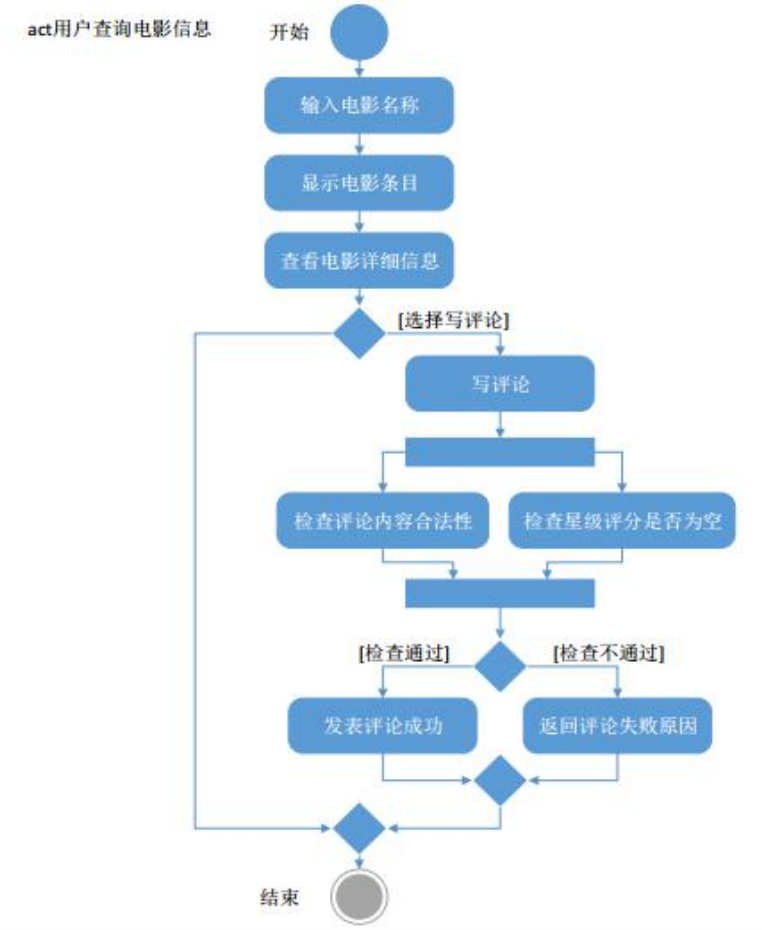

图 5：用户查询电影信息活动图

1) 基本事件流

本用例开始于注册用户希望查询电影信息。

外网评分：指来自豆瓣、IMDB、烂番茄等三个外网对该电影的评分。

点评：指本系统用户对该电影的评论与星级评分。

Ⅰ系统请求注册用户输入希望查询的电影相关词条。

A1：相关电影不存在

Ⅱ系统展示与该搜索词条相关的电影介绍条目，点击该条目，则跳转至该电影的详细信息页面，包括导演、外网评分、主演、上映时间、剧情简介的介绍，以及来自本系统用户的点评。

Ⅲ 注册用户选择对该电影新建评论，则以下子事件流将被执行：

Ⅲ.1 写评论

a) 系统请求注册用户书写评论。

A2：评论内容不合法

A3：注册用户没有选择星级评分

b) 注册用户发表评论。

Ⅳ系统把当前用户的新评论信息（如有新增）增加到数据库中。

2) 后备事件流

A1. 相关电影不存在

- ✧ 系统提示“相关电影不存在”错误信息；
- ✧ 返回基本事件流第 I 步。

A2. 评论内容不合法

- ✧ 系统提示“评论长度不少于 10 个字符”错误信息；
- ✧ 返回基本事件流第Ⅲ. 1 步。

A3. 注册用户没有选择星级评分

- ✧ 系统提示“星级评分不能为空”错误信息；
- ✧ 返回基本事件流第Ⅲ. 1 步。

(4) 特殊需求

无。

(5) 前置条件

本用例开始前注册用户已经登录系统，并选择“查询电影信息”功能。

(6) 后置条件

如果用例成功，注册用户将获得相关电影信息并发表评论与主题帖。若失败，系统状态不改变。

管理电影信息

(1) 简要说明

本用例描述管理员如何管理电影信息

(2) 参与者

管理员

(3) 事件流

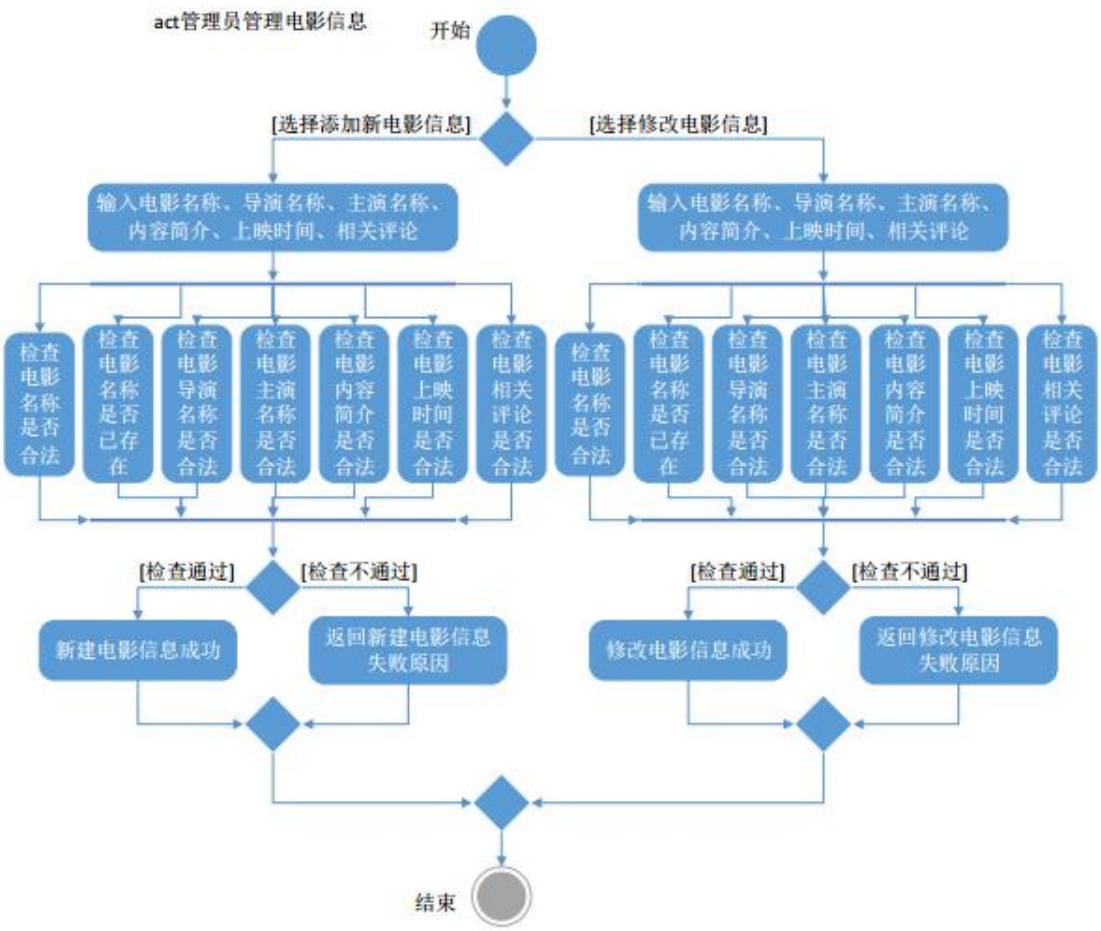

图 6：管理员管理电影信息活动图

1) 基本事件流

本用例开始于管理员希望添加新电影信息或者修改电影信息。

Ⅰ管理员选择添加管理电影信息，则以下的一条子事件流将被执行：

如果选择的是“添加新电影信息”，添加新电影信息子事件流将被执行：

如果选择的是“修改电影信息”，修改电影信息子事件流将被执行。

Ⅰ.1 添加新电影信息

a) 系统请求管理员输入电影名称、导演名称、主演名称、内容简介、上映时间、相关评论；

A1：电影名称不合法

A2：电影名称已存在

A3：电影导演名称不合法

A4：电影主演名称不合法

A5：电影内容简介不合法

A6：电影上映时间不合法

A7：电影相关评论不合法

b) 管理员新增电影信息

Ⅰ.2 修改电影信息

a) 系统请求管理员修改电影名称、导演名称、主演名称、内容简介、上映时间、相关评论的内容。

A8：电影名称不合法

A9：电影名称已存在

A10：电影导演名称不合法

A11：电影主演名称不合法

A12：电影内容简介不合法

A13：电影上映时间不合法

A14：电影相关评论不合法

b) 管理员修改电影信息

Ⅱ系统把当前管理员的新增或者修改的电影信息增加到数据库中。

2) 后备事件流

A1. 电影名称不合法

✧ 系统提示“电影名称不为空”错误信息；

✧ 返回基本事件流第Ⅰ.1 步。

A2. 电影名称已存在

✧ 系统提示“电影名称已存在，是否确认修改”错误信息；

✧ 选择“否”则返回基本事件流第Ⅰ.1 步，选择“是”则继续执行基本事件流第Ⅱ步。

A3. 电影导演名称不合法

✧ 系统提示“电影导演名称不为空”错误信息；

✧ 返回基本事件流第Ⅰ.1 步。

A4. 电影主演名称不合法

✧ 系统提示“电影主演名称不为空”错误信息；

✧ 返回基本事件流第Ⅰ.1 步。

A5. 电影内容简介不合法

✧ 系统提示“电影内容简介长度不少于 10 个字符”错误信息；

✧ 返回基本事件流第Ⅰ.1 步。

A6. 电影上映时间不合法

✧ 系统提示“电影上映时间格式不正确”错误信息；

✧ 返回基本事件流第Ⅰ.1 步。

A7. 电影相关评论不合法

✧ 系统提示“电影相关评论长度不少于 10 个字符”错误信息；

✧ 返回基本事件流第Ⅰ.1 步。

A8. 电影名称不合法

✧ 系统提示“电影名称不合法”错误信息；

✧ 返回基本事件流第Ⅰ.2 步。

A9. 电影名称已存在

✧ 系统提示“电影名称已存在”错误信息；

✧ 返回基本事件流第Ⅰ.2 步。

A10. 电影导演名称不合法

✧ 系统提示“电影导演名称不为空”错误信息；

✧ 返回基本事件流第Ⅰ.2 步。

A11. 电影主演名称不合法

- ✧ 系统提示“电影主演名称不为空”错误信息；
- ✧ 返回基本事件流第Ⅰ.2 步。

A12. 电影内容简介不合法

- ✧ 系统提示“电影内容简介长度不少于 10 个字符”错误信息；
- ✧ 返回基本事件流第Ⅰ.2 步。

A13. 电影上映时间不合法

- ✧ 系统提示“电影上映时间格式不正确”错误信息；
- ✧ 返回基本事件流第Ⅰ.2 步。

A14. 电影相关评论不合法

- ✧ 系统提示“电影相关评论长度不少于 10 个字符”错误信息；
- ✧ 返回基本事件流第Ⅰ.2 步。

⑷ 特殊需求

无。

⑸ 前置条件

本用例开始前管理员已经登录系统，并选择“管理电影信息”功能。

⑹ 后置条件

如果用例成功，管理员将新增电影信息或者修改电影信息。若失败，系统状态不改变。

1.4 电影评论系统补充规约

1.目标

本文档的目的是定义电影评论系统的需求。本补充规约列出了不便于在用例模型的用例中获取的系统需求。

补充规约和用例模型一起记录关于系统的一整套需求。

2.范围

(1)本补充规约适用于电影评论系统。

(2)本规约除定义了在许多用例中所共有的功能性需求以外，还定义了系统的非功能性需求，例如：可用性、可靠性、性能和可支持性等。

3.参考

无

4.可用性

桌面用户界面应与 Windows7/8/10 兼容。

5.可靠性

电影评论系统能够提供 7×24 小时不间断服务。

6.性能

(1)用户在搜索电影的时候，系统的相应时间应小于 2 秒；

(2)用户在更新个人信息的时候，系统的相应时间应小于 2 秒。

7.可支持性

无

8.安全性

(1)只有管理员才能够更改电影的信息。

(2)只有本人能够修改自己的评论。

(3)登录系统时,须输入用户名和密码进行身份验证。

9.设计约束

本系统通过共享数据库方式获取电影评论系统的电影信息。

1.5 术语表

本部分内容包括与本系统开发相关的关键概念定义。

⑴电影评论系统

用户可以查询电影，查看对应的评论以及评分，撰写自己对该电影的评论并做出评分。管理员管理电影信息。下文所称的本系统均指电影评论系统。

(2)注册用户

拥有撰写电影评论，给出电影评分权限的人。

(3)管理员

由系统设定，负责更新电影信息的人。

(4)用户名

注册用户和管理员在系统中的唯一标识。

(5)电影信息

包括电影的名称、导演、主演、上映时间、内容简介、相关评论和评分等信息。

(6)电影名称

电影在系统中的唯一标识。

2.电影评论系统架构设计

2.1 架构描述

本电影评论系统是基于 Flask 框架用 Python 语言开发实现，使用 MVC 层次结构将本系统进行分层，将整体划分为模型层，视图层和控制层，使得每个层次便于管理。

2.2 架构图

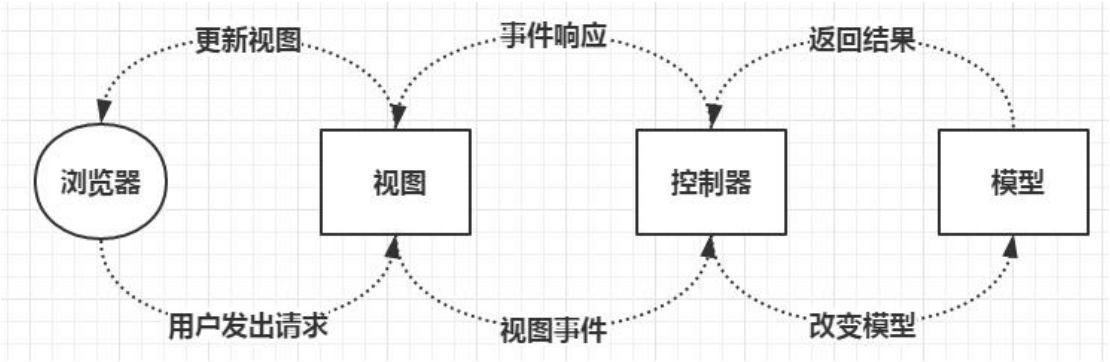

图 7：电影评论系统架构图

1.视图层

视图层是用户与系统交互的界面。用户在浏览器中发出请求，视图层获取请求后触发视图事件，交由控制器层处理。在最后视图层获得返回的改变事件更新视图。本系统视图层的模块包括：用户登录，用户注册，用户管理个人信息，查询电影，电影详细信息，管理电影信息。

2.控制器

控制器是对业务逻辑代码进行的抽象和封装。接收视图层的事件，调用模型层的方法进行相应的操作，将业务结果返回给视图层更新视图。本系统控制器的模块包括：设置，查询，发布，页面逻辑控制。

3.模型层

模型层包含系统的实体对象。本系统模型层的模块包括：用户，电影，管理员，点评。

2.3 系统关键抽象

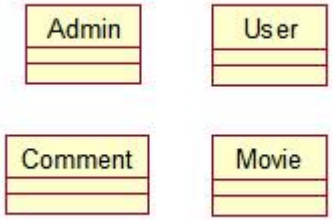

图 8：本系统关键抽象

3.电影评论系统软件设计技术

3.1 Object-Oriented Programming

面向对象编程的设计思想在整个系统中都有所体现，因为我们使用的是 Python 语言。Python 语言本身就是基于面向对象设计的，创建一个类和对象是很容易的，所以面向对象编程在源代码中的位置可以理解为整个项目的代码，最主要是在 forms.py、models.py 以及 views.py 文件中的所有代码。

3.2 Aspect-Oriented Programming

面向切面编程是对业务处理过程的切面进行提取，面对处理过程的某个步骤或阶段，降低逻辑代码各个部分之间的耦合性。根据上网查得的资料得知，Python 中使用的装饰器可以算作面向切面的编程，因此面向切面编程在项目中的体现主要在在 views.py 中的每个方法前使用的装饰器@app.route()。

Views.py line 35

Views.py line 41

Views.py line 65

Views.py line 88

Views.py line 96

Views.py line 105

Views.py line 131

Views.py line 211

Views.py line 238

Views.py line 265

Views.py line 271

Views.py line 291

3.3 Service-Oriented Architecture

面向服务架构要求一个服务在创建后能够用于多个应用和业务流程，并且该服务是独立、自包含的请求，这样能够使得整个服务更加灵活。在源代码中的体现主要是在/static/css 文件夹下的各个 css 代码文件，在 css 中每个样式设置都能够被标签使用，因此可以灵活地搭配从而渲染出更好的页面效果。

例如 login.css line 20 以及 Login.css line 24 用在了多个标签上。

3.4 Design Patterns

在此项目中使用了一些设计模式，有模板方法。模板方法是定义了一个操作的骨架，而将一些步骤延迟到子类中，使得子类在不改变父类的结构就可以重定义特定步骤。在此项目中主要体现在 Layout.html 全部代码与其他 html 文件上，其他页面都是继承此页面，然后根据自己的特定内容进行拓展

Layout.html line 16

Layout.html line 70

Layout.html line 86

以上三个部分的代码都是可以进行拓展的部分。

4.电影评论系统划分模块

将电影评论系统按照使用者对象划分了三个主要模块，管理员模块、用户模块和电影模块。

4.1 管理员模块设计

View(视图层)包括的类有：

Login Logout Home

Controller(控制器)包括的类有：

AddMovie ModifyMovie PersonalSetting ModifyComment Search

Model(模型层)包括的类有：

User Comment

4.2 用户模块设计

View(视图层)包括的类有：

Login Logout Home MyComment

Controller(控制器)包括的类有：

PersonalSetting ModifyComment Regist Search

Model(模型层)包括的类有：

User Comment

4.3 电影模块设计

View(视图层)包括的类有：

Information MovieList

Controller(控制器)包括的类有：

ModifyMovie AddMovie Search

Model(模型层)包括的类有：

Movie Comment

5.电影评论系统用例分析

5.1 用户管理个人信息用例分析

5.1.1 用户管理个人信息用例功能描述

用户可以使用这个功能进行修改个人信息或评论（包括星级）的操作。

5.1.2 用户管理个人信息用例交互过程

1．用户看到个人信息以及对全部电影的评论。

2．用户根据自己的需求输入新信息后进行更改个人信息操作。

3．管理个人信息页面调用函数对用户的输入进行合法性判断。如果输入合法，则数据库保存用户对个人信息以及评论的修改；否则提示用户输入不合法。

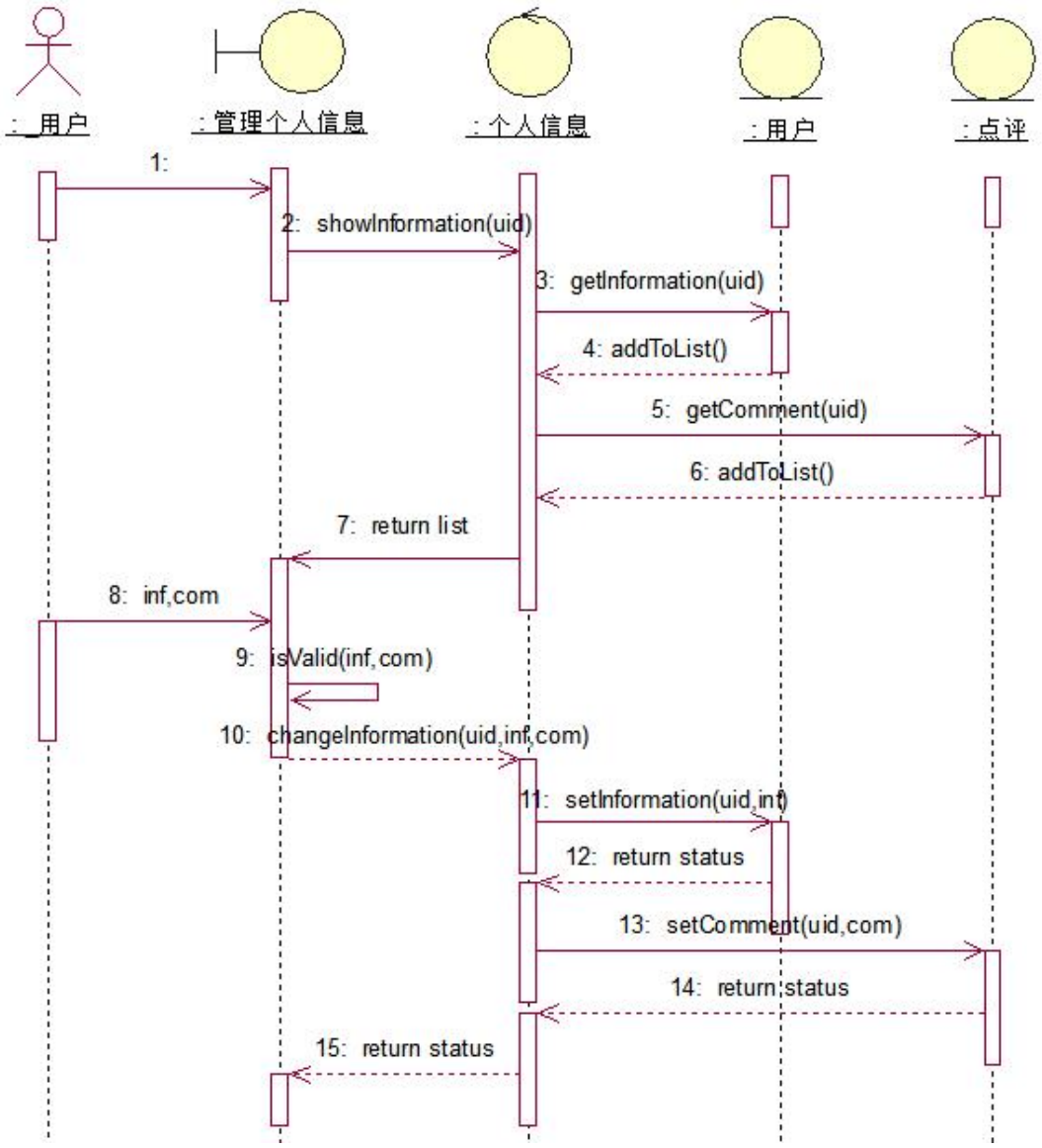

图 9：用户管理个人信息时序图

5.1.3 个人信息管理用例的类的分析与设计

边界类：在本用例中，边界类为用户管理个人信息。该页面有多个输入框给用户输入修改信息或追加评论（包括星级）。边界类的属性以及方法如图 10 所示；

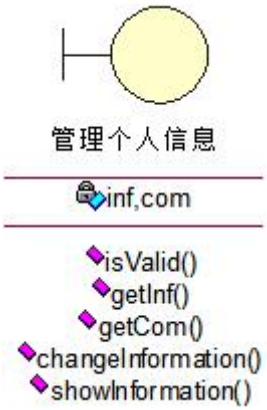

图 10：边界类的属性以及方法

控制类：在本用例中，控制类为个人信息。首先是通过 getInformation()方法取得该用户此前的个人设置及点评。控制类的属性以及方法如图 11 所示；

图 11：控制类的属性以及方法

实体类：在本用例中，实体类有用户和点评。其中用户实体类存储着用户的名字、邮箱等基本信息。而点评实体类包括该用户所有的对电影的点评。实体类的属性和方法如图 12 所示。

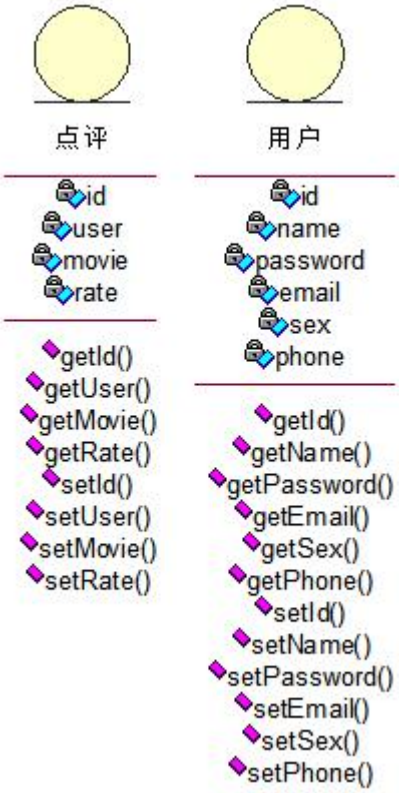

图 12：实体类的属性以及方法

类与类之间的关系总体类图如图 13 所示

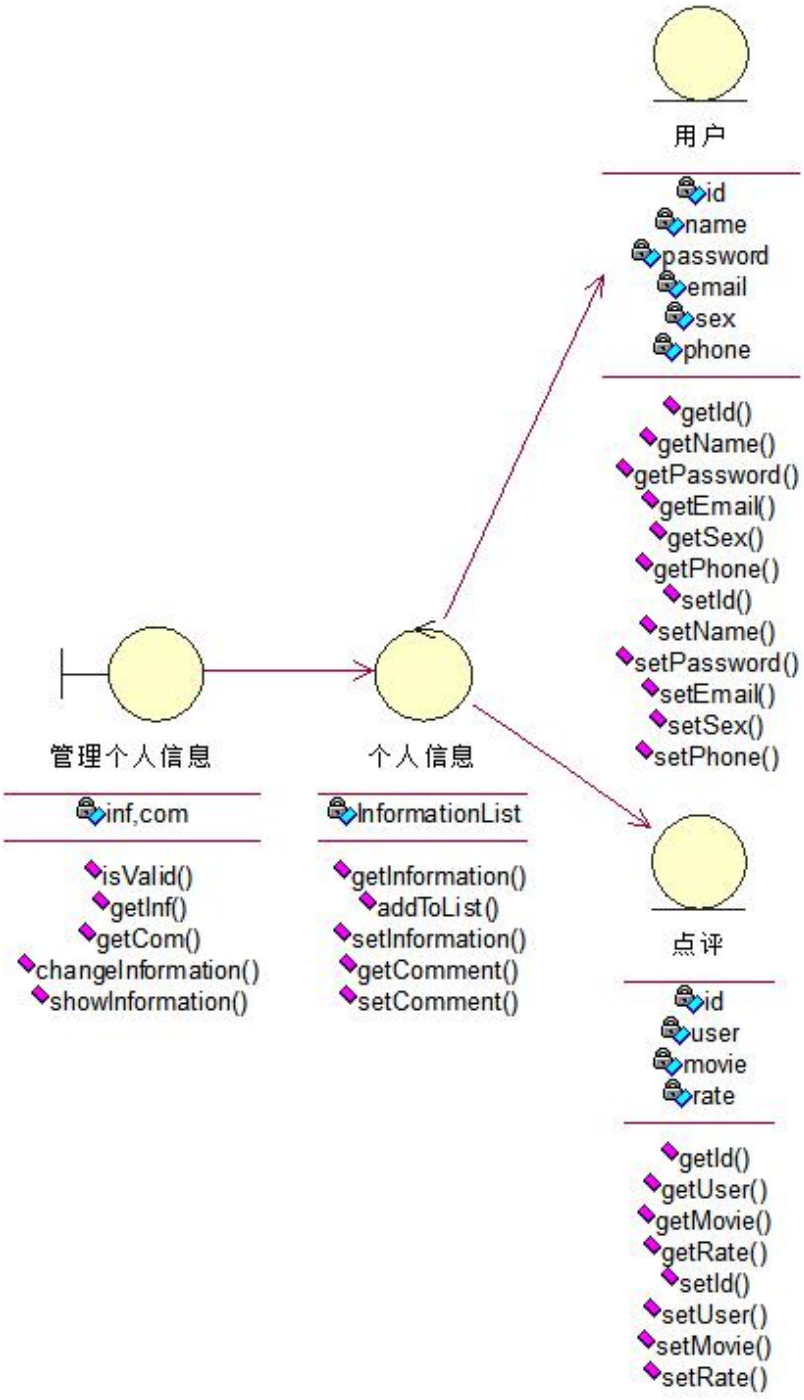

图 13：类的关系

5.2 用户查询电影用例分析

5.2.1 用户查询电影用例功能描述

用户可以使用这个功能进行查询电影信息的操作。

5.2.2 用户查询电影用例交互过程

1．用户在查询电影界面中点击查询输入框，输入电影名称后点击查询按钮。

2．查询电影界面调用校验函数检查电影名称是否合法，合法则允许用户查询；否则返回电影名称错误的信息。若该电影信息不存在，则返回“相关电影不存在”错误信息。

3．电影查询控制类根据用户的输入在数据库中查找电影。若成功搜索则将电影信息回然后继续查找与该电影相关的点评，返回点评。若该电影信息不存在，则返回“相关电影不存在”错误信息。

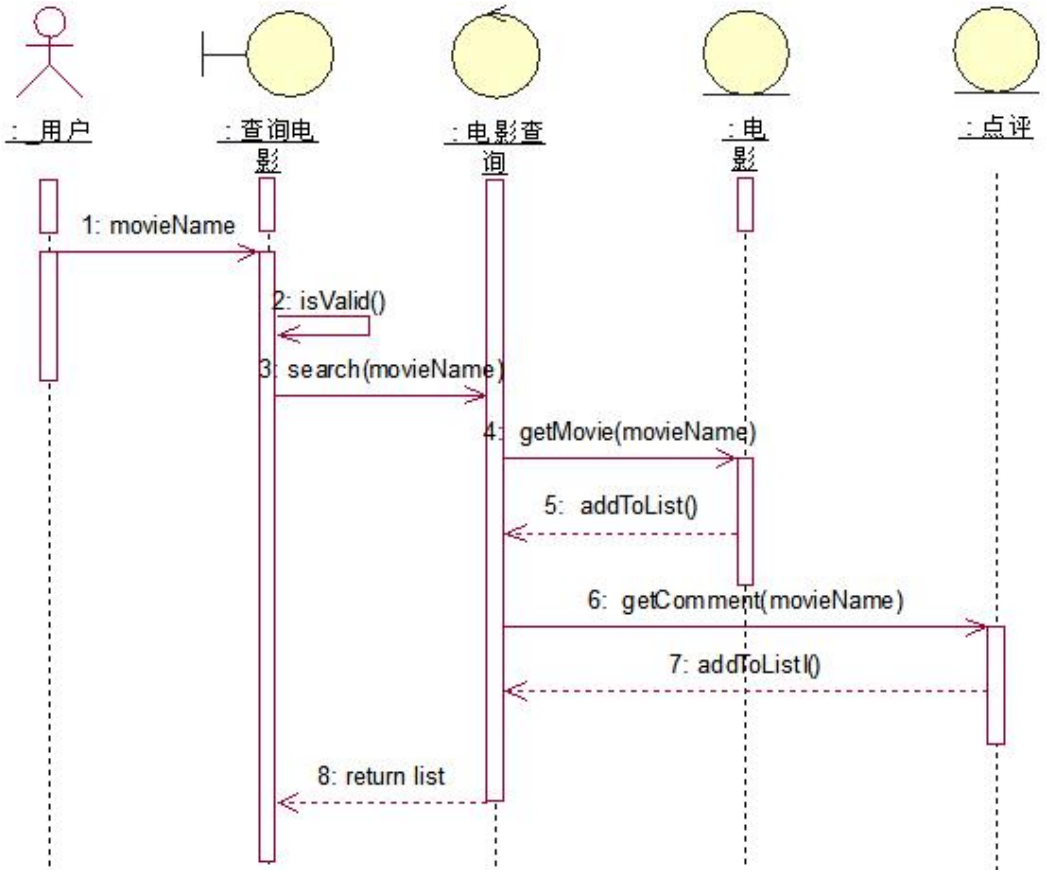

图 14：用户查询电影时序图

5.2.3 用户查询电影的类的分析与设计

边界类：在本用例中，边界类为查询电影界面。该页面有一个输入框，获取用户输入的信息。边界类的属性及方法如图 15 所示。

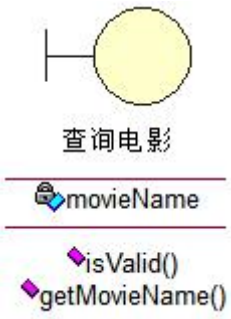

图 15：边界类的属性与方法

控制类：在本用例中，控制类为电影查询。通过调用方法 search()，对数据库中所有电影实例进行 getMovie()操作，如果与参数 movieName 相匹配，则返回结果并将结果加入到一个列表中。控制类的属性与方法如图 16 所示。

图 16：控制类的属性与方法

实体类：在本用例中，实体类为电影和点评。电影实体类包含电影名称、导演、主演、外网评分、上映日期、简介等信息。点评是本系统注册用户对某电影的评论和评分。实体类的属性与方法如图 17 所示。

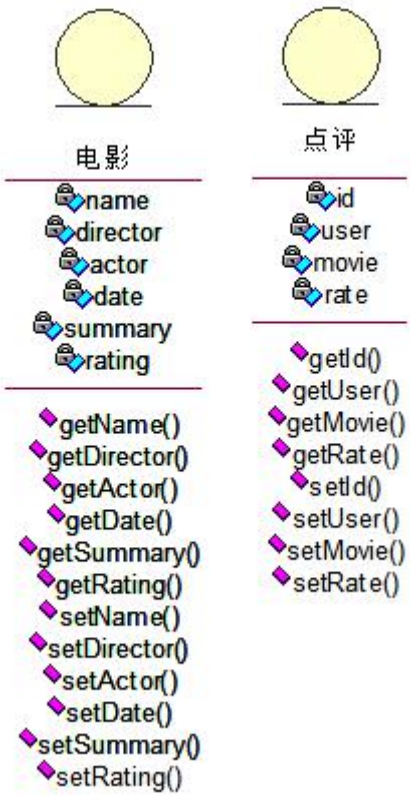

图 17：实体类的属性与方法

类与类之间的关系总体类图如图 18 所示

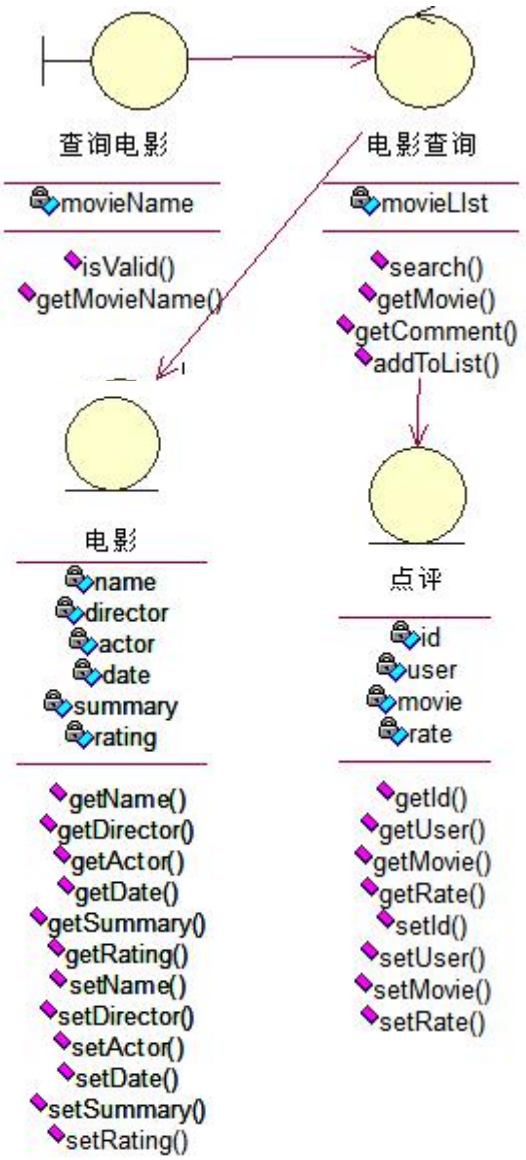

图 18：类的关系

5.3 管理电影信息用例分析

5.3.1 管理电影用例功能描述

管理员可以使用这个功能进行更新电影信息的操作。

5.3.2 管理电影用例交互过程

管理员输入电影名称、导演、主演、外网评分、上映时间、简介等信息。

管理电影界面调用方法校验管理员的输入是否合法。如果输入合法，允许管理员进行电影信息更新操作；否则提示输入不正确，要求管理员重新输入。

更新电影类根据管理员的输入判断数据库中是否存在该电影，如电影已存在，则提示管

理员已存在该电影，是否确认修改；否则在数据库中新建该电影信息。

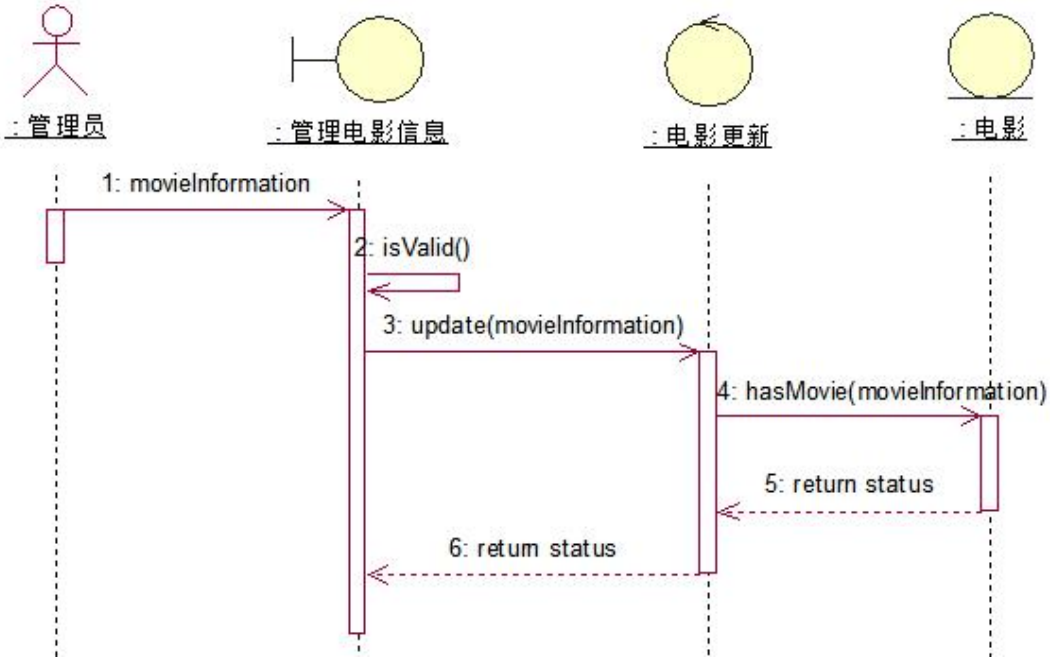

图 19：管理电影信息时序图

5.3.3 管理电影信息用例的类分析与设计

边界类：在本用例中，边界类为管理电影信息。该页面有多个输入框，让管理员输入电影信息。边界类的属性及方法如图 20 所示。

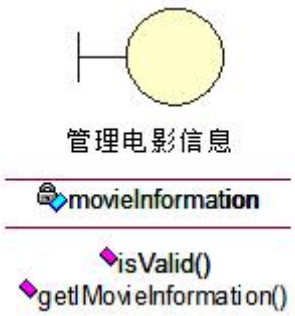

图 20：边界类的属性及方法

控制类：在本用例中，控制类为电影更新。该类通过方法 hasMovie()的调用，查询数据库中是否存在对应的电影信息。如果不存在，在数据库中新增一条信息，同时返回添加成功的状态信息。否则的话返回添加失败的状态信息提示管理员是否确认更新电影信息。控制类的属性与方法如图 21 所示。

图 21：控制类的属性及方法

实体类：为电影实体类。电影实体类包含电影名称、导演、主演、上映日期、简介等信息。实体类的属性及方法如图 22 所示。

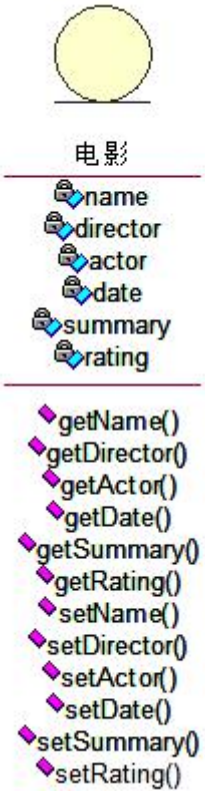

图 22：实体类的属性及方法

类与类之间的关系总体类图如图 23 所示

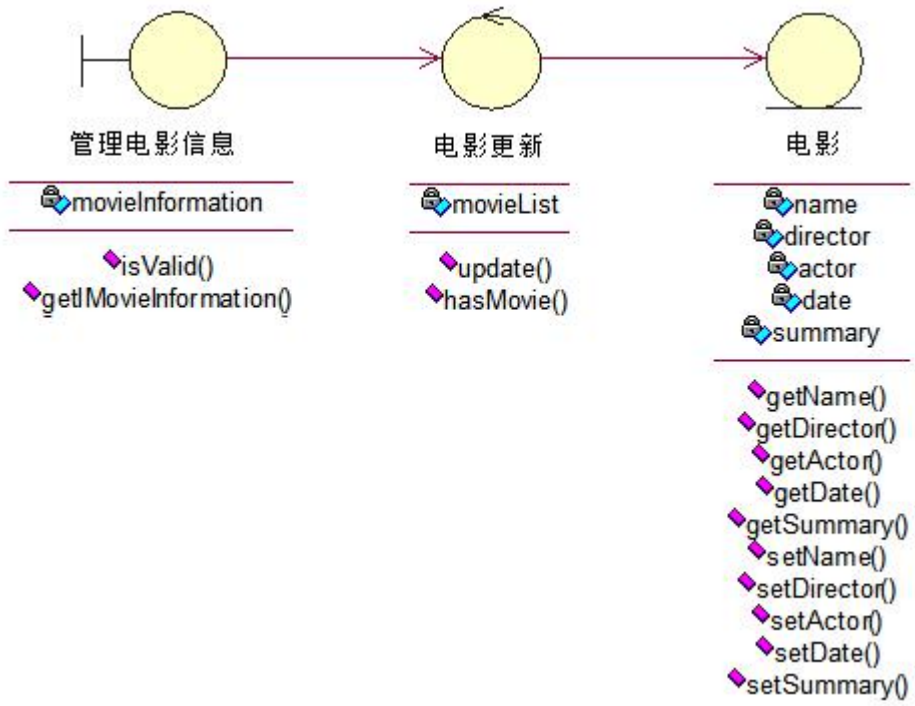

图 23：类的关系

5.4 电影评论系统类的设计

综合以上用例分析，可以得到本系统的类图，如图 24 所示。

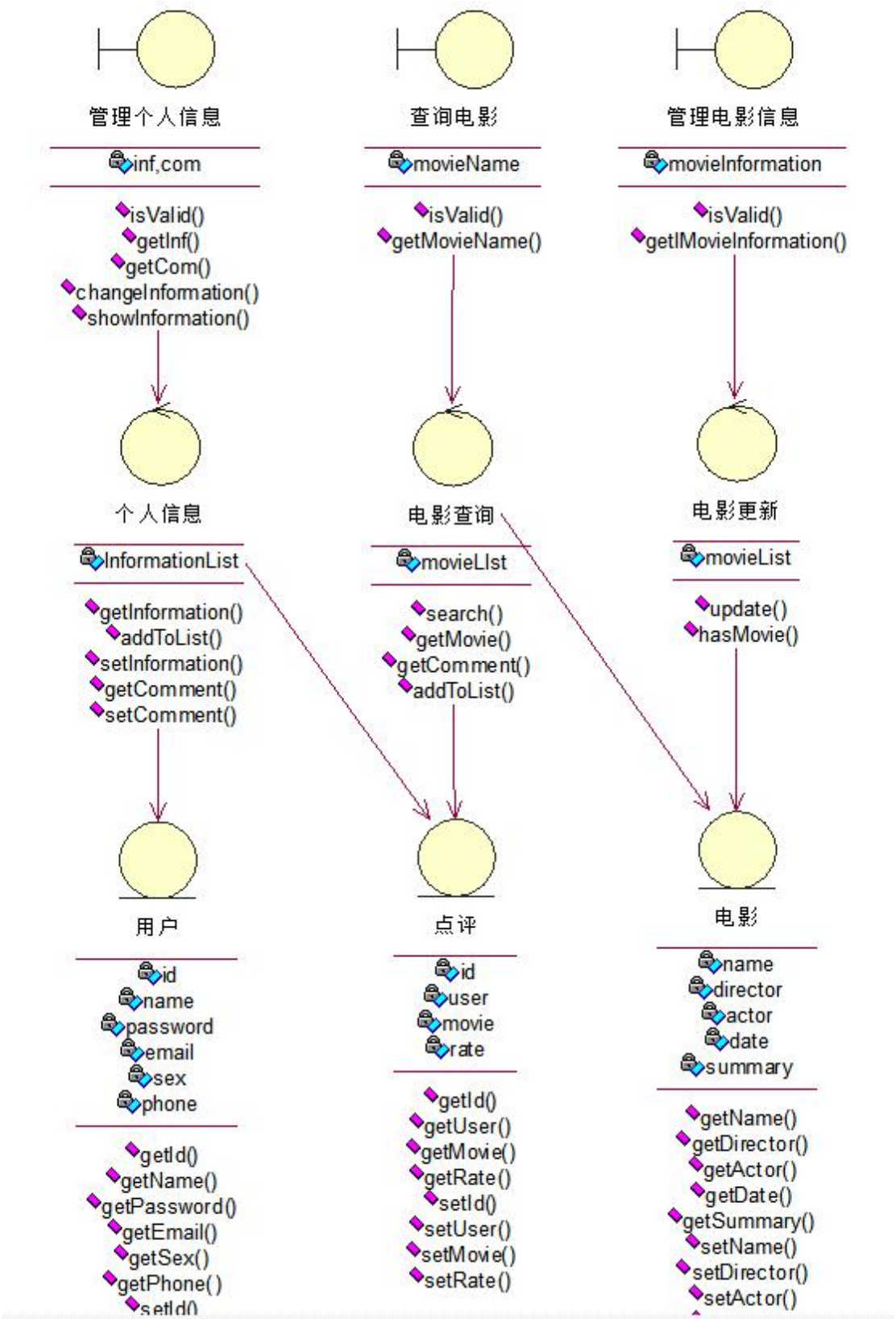

图 24 电影评论系统类图

5.5 电影评论系统分析机制

分析类到分析机制的映射如表 1 所示

表 1 分析类到分析机制的映射

分析类（analysis classes）	分析机制（analysis mechanism）
登录	图形化
注册	图形化
查询电影	图形化
管理个人信息	图形化
管理电影	图形化
个人信息	资源管理
电影更新	资源管理
电影查询	资源管理
用户	持久化，安全性
电影	持久化，安全性
点评	持久化，安全性
帖子	持久化，安全性
管理员	持久化，安全性

6.电影评论系统子系统及其接口设计

6.1 子系统及其接口设计

电影评论系统包括两个子系统：电影查询子系统，个人信息管理子系统。

两个子系统向主程序提供获取电影信息和个人信息的接口服务，主系统通过调用子系统提供的接口满足用户的需求。

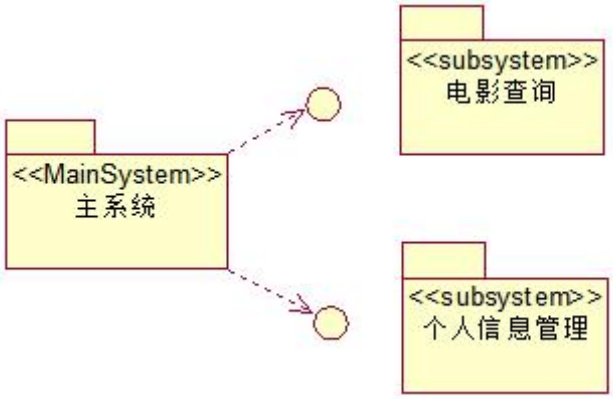

图 25：电影评论系统接口设计

6.2 子系统主要流程

系统主要流程包括：用户输入电影名称，主程序调用电影查询子系统，获取电影详细信息及相关评论并显示在页面上。主程序调用个人信息管理子系统，显示已保存的个人信息；用户输入需要更改的个人信息后，主系统继续调用子系统修改个人信息。

图 26 表示调用电影查询子系统的流程。用户点击查询电影，主程序向电影管理子系统发出打开查询电影页面请求。

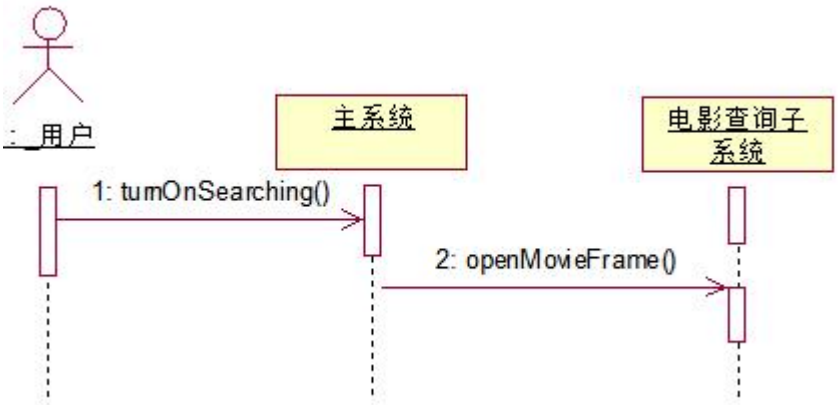

图 26：电影查询模块顺序图

图 27 表示调用个人信息管理子系统的流程。用户点击更改个人信息按钮，主程序向个人信息管理子系统发出打开个人信息管理界面请求。

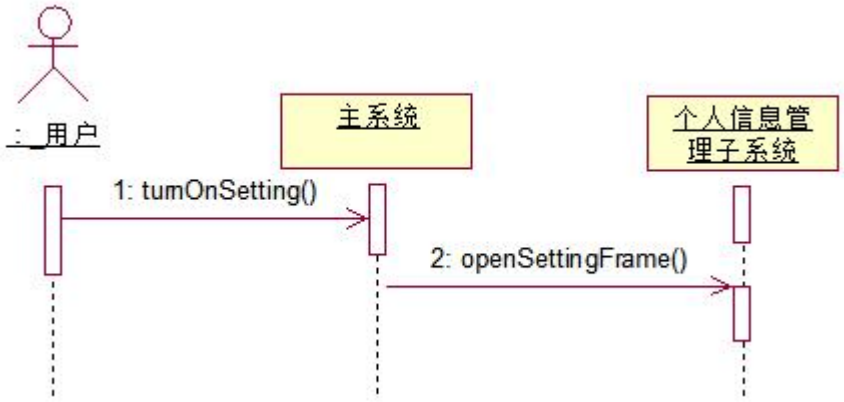

图 27：个人信息管理模块顺序图

6.3 子系统内部设计

6.3.1 电影查询子系统内部设计

电影查询子系统依然使用 MVC 三层架构模式，分为视图层，控制层以及模型层，如图 28 所示。

1．视图层

由电影查询窗口和输入框等组成。

2．控制层

控制层将用户的输入作为参数，调用方法对在底层的模型层进行查询操作，里面包含 search()，getMovie()的方法。

3．模型层

包括电影、点评这两个实体类。

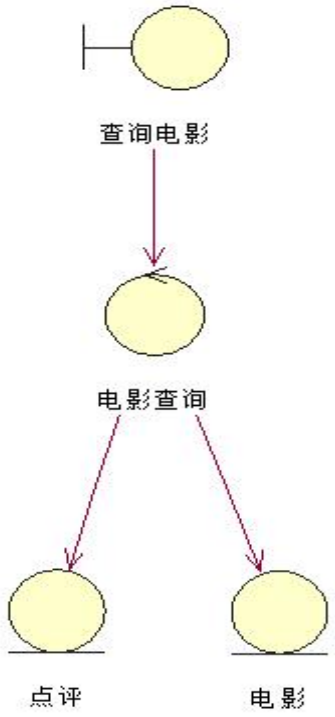

图 28：电影查询子系统架构

6.3.2 个人信息管理子系统内部设计

个人信息管理子系统依然使用 MVC 三层架构模式，分为视图层，控制层以及模块层，如图 29 所示。

1．视图层

由个人信息窗口和输入框等组成。

2．控制层

控制层提供页面跳转、设置个人信息服务。首先通过方法访问数据库，在页面上显示当前的个人信息。根据用户的输入调用方法来设置个人信息。

3．模型层

包括用户、点评这两个实体类。

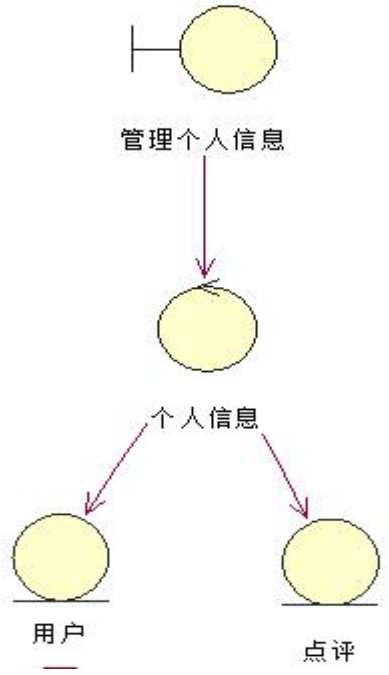

图 29：个人信息管理子系统架构

7.电影评论系统构件设计

7.1 分析电影评论系统并发需求

本项目需联网使用，实时更新电影信息更改及电影点评数据库，故当出现多机多用户共同查询某部电影和发表点评时，需要执行多线程操作。

对本项目的析取模块的并发情况进行研究后，认为存在并发，进行构件设计如下。

对构件设计的具体说明在 5.2。

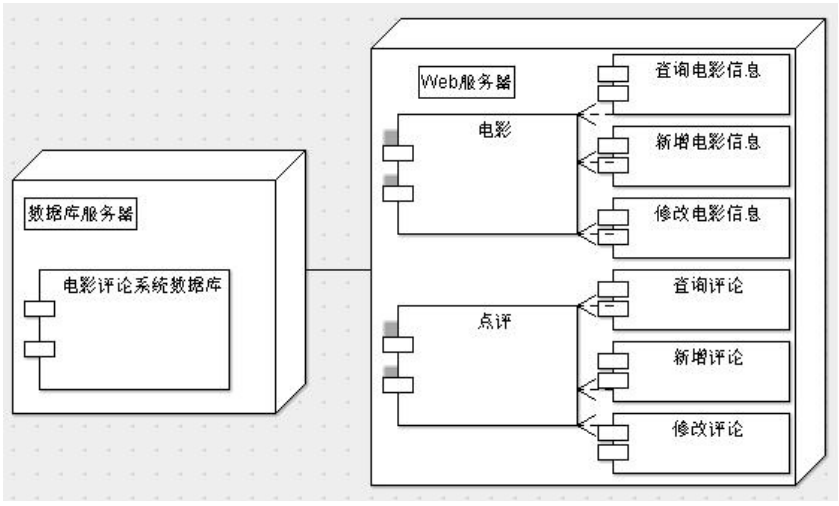

图 30：电影评论系统构件设计

7.2 电影评论系统运行时并发设计

7.2.1 电影查询子系统运行时并发设计

电影查询子系统需使用电影信息构件，涉及查询信息，新增信息和修改信息等功能。这些功能可能出现以下并发情况：

A1: 多个用户同时查询某一部电影信息；

A2: 单个或多个用户进行查询时管理员同时修改该电影信息；

A3: 单个或多个用户进行查询时管理员同时新增该电影信息。

当 A1 发生时，电影构件向所有用户同时提供最新的该部电影信息；

当 A2 发生时，用户查询到的是管理员修改前的电影信息；管理员修改完后，电影构件更新，当用户下次查询时，查询到的即更新数据后的电影信息；

当 A3 发生时，用户查询到的电影信息为空；管理员新增完后，电影构件更新，当用户下次查询时，查询到的即新增数据后的电影信息。

7.2.2 个人信息管理子系统运行时并发设计

个人信息管理子系统需使用点评构件，涉及查询评论（包括星级），新增评论（包括星级）和修改评论（包括星级）等功能。这些功能可能出现以下并发情况：

A1: 多个用户同时查询某一部电影的评论；

A2: 多个用户同时对某一部电影新增评论；

A3: 某一用户修改评论同时其他用户在查询该电影的信息。

当 A1 发生时，点评构件向所有用户同时提供最新的该部电影的评论信息；

当 A2 发生时，用户界面看到的是新增评论前的电影点评信息；用户新增完后，点评构件更新，当用户下次打开评论界面时，看到的即更新数据后的电影点评信息；

当 A3 发生时，用户界面看到的是修改评论前的电影点评信息；用户修改完后，点评构件更新，当用户下次打开评论界面时，看到的即更新数据后的电影点评信息。