**中国矿业大学**

**计算机科学与技术学院**

**2019级本科生实验报告**

课程名称 软件工程实践

开课学期 2021学年第一学期

报告时间 2022年1月1日

学生姓名 胡钧耀

学 号 06192081

班 级 2019-4班

专 业 计算机科学与技术

任课教师 刘迎春

# 软件需求分析

## 结构化需求分析

### 简介

小组依托阿里云《政务数字参谋产品白皮书》等材料，对**城市智能政务中台**进行结构化方法的软件需求分析。城市智能政务中台的适用对象主要包括**各个政府机构的业务管理人员（审批人）与需要办理业务与进行业务评价的市民（办理人）**。该中台系统**对接各个政府机构的业务系统**，对数据与业务进行**处理分析**，**追踪业务进度**，优化业务流程，对数据与业务进行**可视化**处理。

### 用户分析

**审批人（机构总管、业务工作人员）：**根据国家规定，公务员需要大学专科及以上文化程度，可认为对电脑操作有一定基础，同时又经过部门内部业务培训，对政务中台界面以及业务流程有一定了解，能够比较熟练地使用软件。由于机构总管与业务管理人员每日处理工作量大，使用频率高，软件布局界面要尽量简单，操作要便捷，以提高处理效率。

**办理人（自然人、企业）：**文化程度不一致，不会都非常熟悉软件的使用，使用软件的频率也不会很高，只有需要办理事务的时候才会使用软件，每次使用都需要熟悉一次界面，因此使用流程也需要设计的尽量简单，界面要简洁易懂，不能比现场办理的流程更加复杂。

### 需求分析

根据小组讨论与资料整合，制作了审批人管理系统（Web端）和受理人办理系统（APP）。

对于Web端，将该政务数据中台分为政务协同分析、政务服务分析、政务移动分析三大板块，其中不同板块又分为四个小板块。各自内容详见下表。

表 1 审批人管理系统需求分析

|  |  |  |
| --- | --- | --- |
| 大板块 | 小板块 | 主要需求 |
| 政务协同分析 | 全局概览 | 办公总体核心指标的展示 |
| 活跃分析 | 政务钉钉使用活跃情况 |
| 会议分析 | 在线会议相关数据分析 |
| 工作台分析 | 软件不同板块使用分析 |
| 政务服务分析 | 实施概况 | 审批任务的核心指标展示 |
| 实施统计 | 按照地区或部门查看数据 |
| 实施查询 | 流程进度分析与异常情况分析 |
| 事项办理 | 对事务进行审批或打回 |
| 政务移动分析 | 流量分析 | 查看各页面访问情况 |
| 用户分析 | 查看用户活跃分布情况 |
| 推广分析 | 查看政务服务业务推广情况 |
| 应用分析 | 查看热点应用及热点事项 |

对于APP，主要设计了受理人办事流程一览、办事流程详解、办事进度查询、办事好差评四个板块。各自内容详见下表。

表 2 受理人办理系统需求分析

|  |  |
| --- | --- |
| 板块 | 主要需求 |
| 办事流程一览 | 检索需要办理的事务 |
| 办事流程详解 | 展示需要办理的事务流程解读 |
| 办事进度查询 | 展示当前办事进度 |
| 办事好差评 | 对已办理的事务进行评价 |

### 非功能需求

**性能要求与时间特性要求**：非并发大批量数据交换少于2000秒/百万条，响应时间毫秒级延迟即可；数据的转换和界面更新传送时间等要求不高。简单点可视化报表查询的响应时间小于五秒。

**数据管理能力要求**：需要管理的文卷和记录的个数单位在百万以上、表和文卷的大小规模较大，政务部门会议与文件较多，倡导无纸化办公，数据管理要求高。

**安全保密性要求**：系统所应具备的故障处理能力很高、处理方式及故障后的系统恢复、数据恢复等要求，需要及时在云服务器进行备份，防止机密数据被非法侵入、修改及丢失，可十分钟备份一次。

**其他专门要求**：政府机构单位要求使用方便，市民使用方便，平台界面友好，具有可维护性、可补充性、易读性、可靠性。软件系统需要能全年内稳定连续运行，故障时间不超过千分之一，导致业务连续停止时间不超过6小时。

### 数据流图

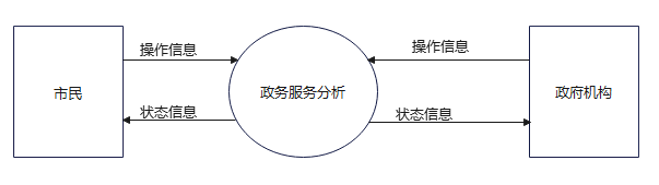


图 1 顶层数据流图

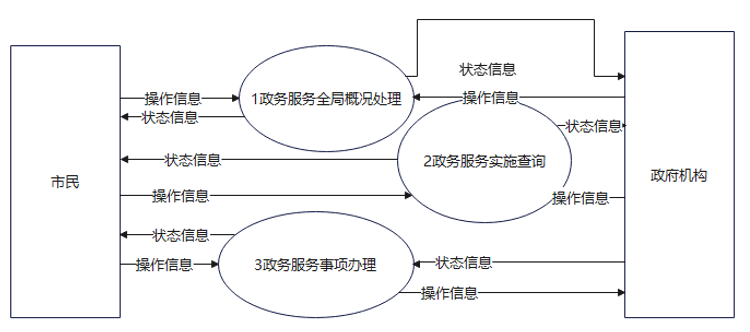


图 2 第一层数据流图

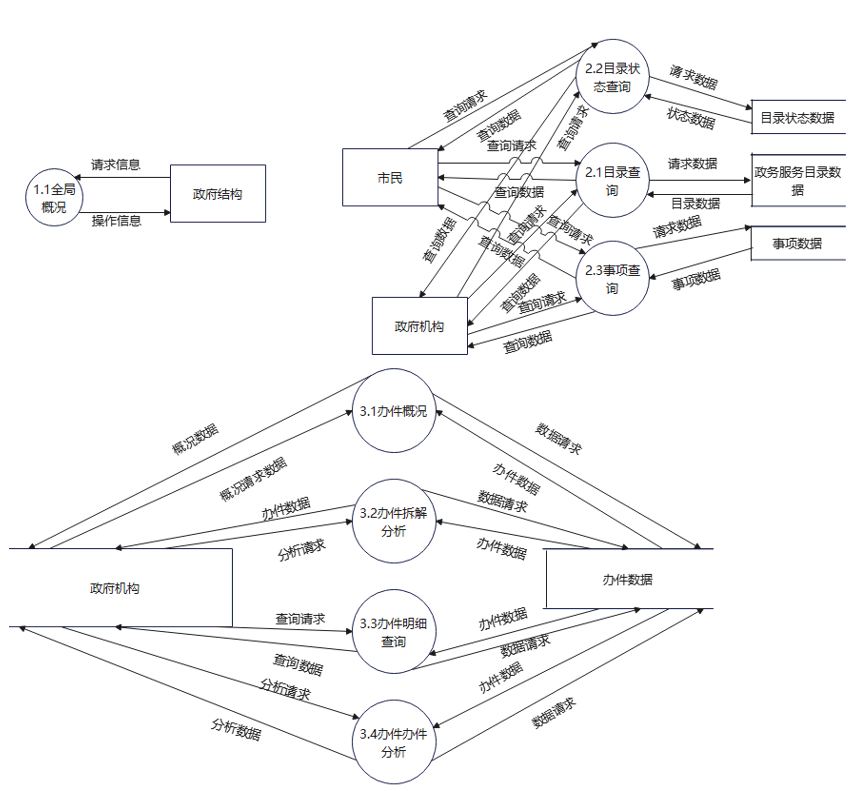
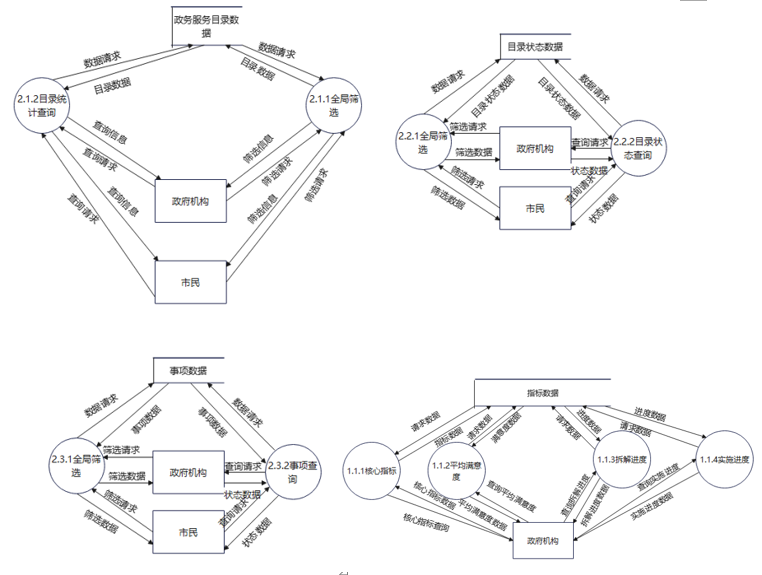


图 3 第二层数据流图



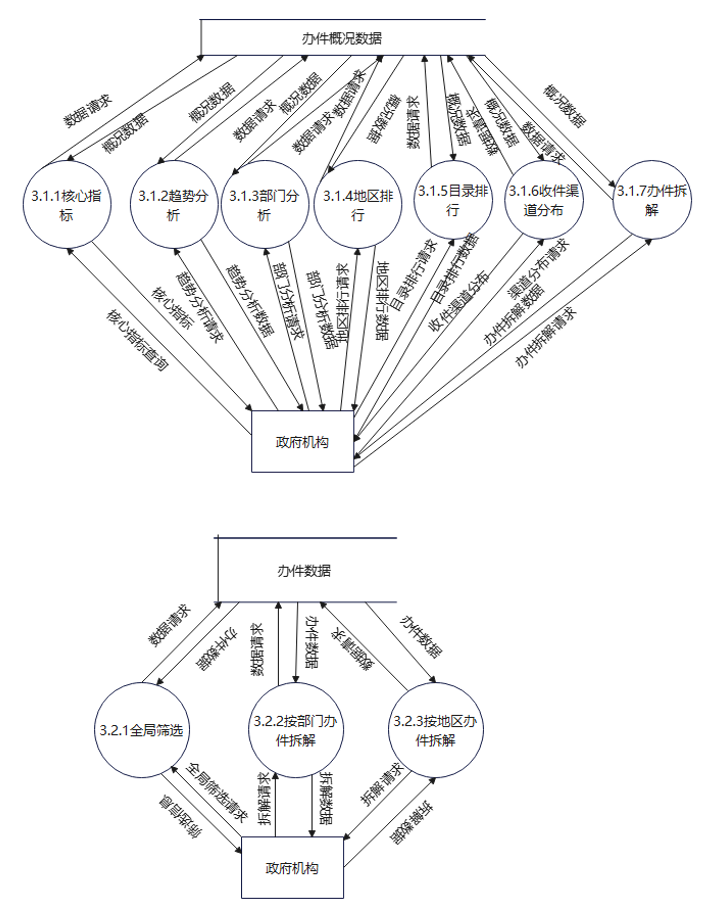
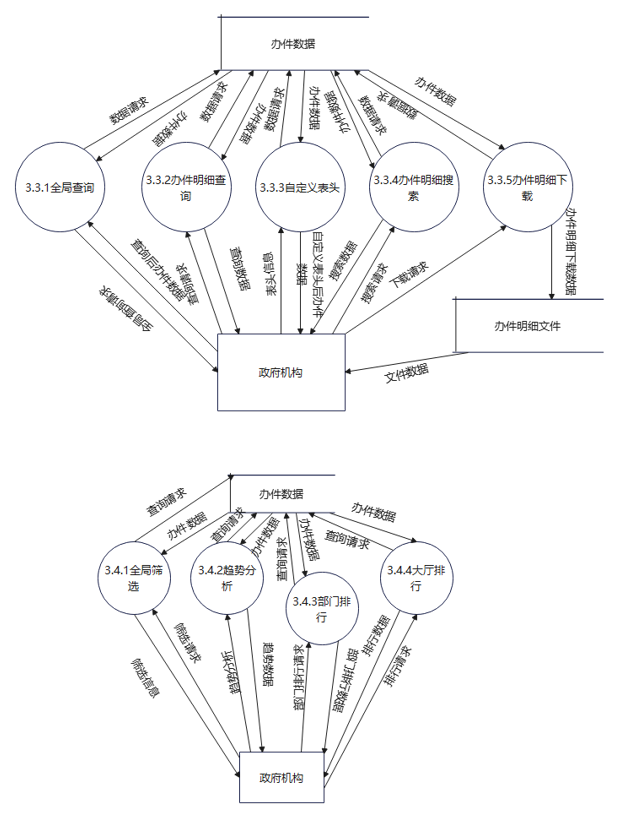
 

图 4 第三层数据流图（2）

## 快速原型工具

### 系统原型

原型使用 Axure RP 9 进行制作，制作工程链接：<https://yk5h0j.axshare.com>

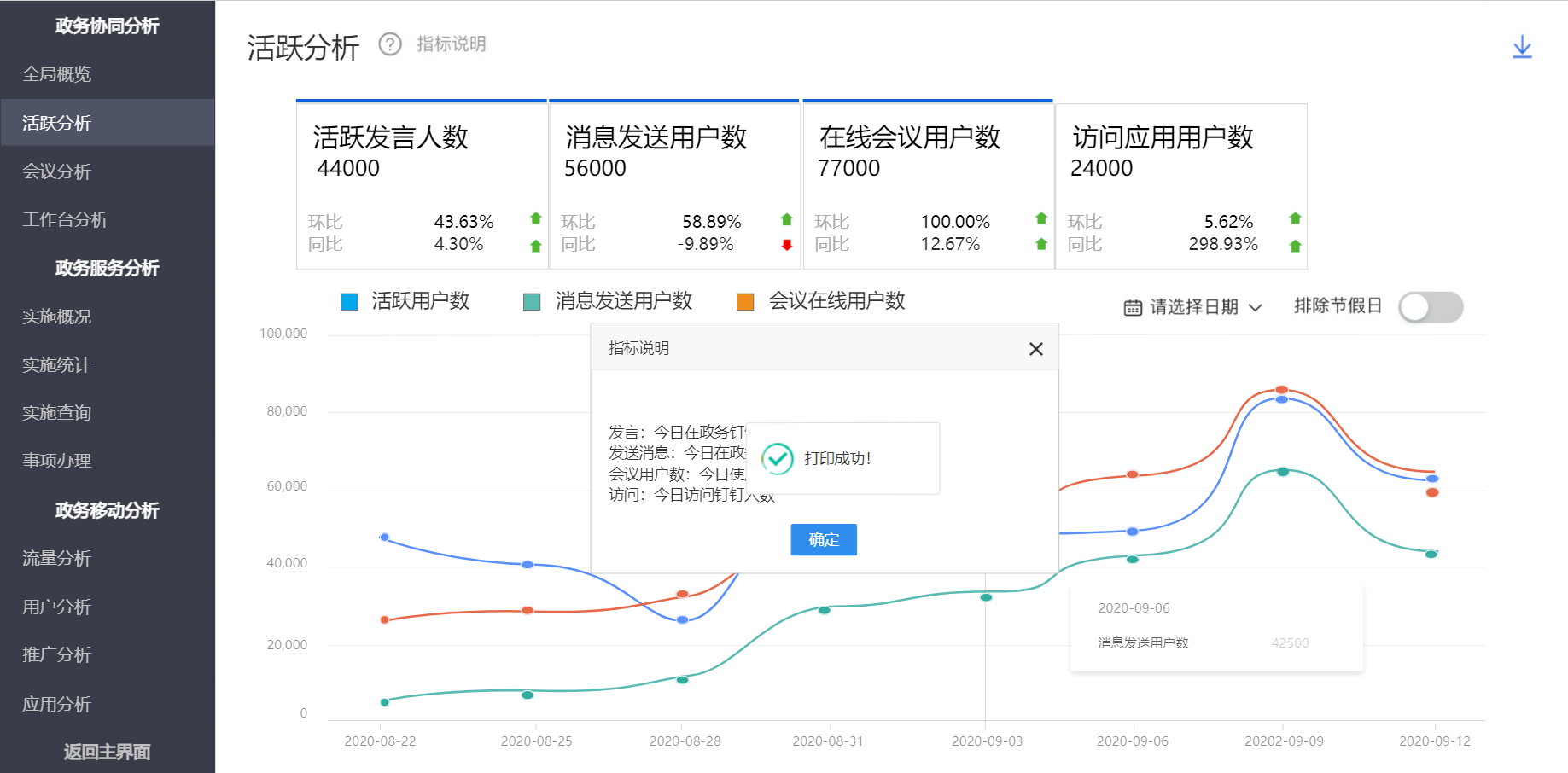
 

图 5 政务数据中台软件原型界面概览（详见上述链接）

### 典型代码

Axure 可以导出 HTML 文件，导出文件夹中内容包括数据、文档、图片、插件、资源、以及最终 HTML 文件，现摘录部分典型代码如下。

如事项办理中的一件事项的展示：

<div id="u996-1" class="" style="width: 1178px; height: 71px;">

<!-- Unnamed (矩形) -->

<div id="u997-1" class="ax\_default box\_1 u997" style="width: 1178px; height: 71px; left: 0px; top: 0px;visibility: inherit">

<div id="u997-1\_div" class="u997\_div" style="width: 1178px; height: 71px;visibility: inherit"></div>

<div id="u997-1\_text" class="text u997\_text" style="display:none; visibility: hidden">

<p></p>

</div>

</div>

<!-- title (矩形) -->

<div id="u998-1" class="ax\_default heading\_11 u998" data-label="title" style="width: 517px; height: 32px; left: 184px; top: 21px;visibility: inherit">

<div id="u998-1\_div" class="u998\_div" style="width: 517px; height: 32px;visibility: inherit"></div>

<div id="u998-1\_text" class="text u998\_text" style="visibility: inherit">

<p><span>xx民办高中筹设</span></p>

</div>

</div>

<!-- Unnamed (矩形) -->

<div id="u999-1" class="ax\_default button u999" style="width: 140px; height: 40px; left: 1006px; top: 17px; visibility: inherit; cursor: pointer;">

<div id="u999-1\_div" class="u999\_div" style="width: 140px; height: 40px;visibility: inherit"></div>

<div id="u999-1\_text" class="text u999\_text" style="visibility: inherit">

<p><span>查看</span></p>

</div>

</div>

<!-- Unnamed (复选框) -->

<div id="u1000-1" class="ax\_default checkbox u1000 selected" style="width: 41px; height: 40px; left: 24px; top: 16px;visibility: inherit">

<label id="u1000\_input\_label" for="u1000-1\_input" style="position: absolute; left: 0px;">

<img id="u1000-1\_img" class="img u1000\_img selected" src="images/事项办理/u1000\_selected.svg">

<div id="u1000-1\_text" class="text u1000\_text" style="display:none; visibility: hidden">

<p id="cache0"></p>

</div>

</label>

<input id="u1000-1\_input" type="checkbox" value="checkbox" class="u1000\_input selected">

</div>

<!-- type\_bg (矩形) -->

<div id="u1001-1" class="ax\_default box\_2 u1001" data-label="type\_bg" style="width: 79px; height: 42px; left: 86px; top: 15px;visibility: inherit">

<div id="u1001-1\_div" class="u1001\_div" style="width: 79px; height: 42px;visibility: inherit"></div>

<div id="u1001-1\_text" class="text u1001\_text" style="display:none; visibility: hidden">

<p></p>

</div>

</div>

<!-- type (矩形) -->

<div id="u1002-1" class="ax\_default heading\_21 u1002" data-label="type" style="width: 79px; height: 28px; left: 86px; top: 22px;visibility: inherit">

<div id="u1002-1\_div" class="u1002\_div" style="width: 79px; height: 28px;visibility: inherit"></div>

<div id="u1002-1\_text" class="text u1002\_text" style="visibility: inherit">

<p><span>审批</span></p>

</div>

</div>

</div>

## 实验总结

我主要负责的是**政务协同分析**（全局概览、活跃分析、会议分析、工作台分析）**板块的流程图初稿绘制和Web端原型各个板块的界面布局优化、跳转逻辑设置与整合**。在小组合作过程中，起初进展很慢，大家都觉得项目太大了，根本完成不了，但是在老师的指点下，不断划分任务，每个人有比较明确的分工，这样一步一步走，最终还是完成了整个项目。

此外，由于思维的限制和经验能力不足，我们无法对有更特殊的业务需求，例如对企业机构进行分析，因而本软件没有考虑企业在政府机构进行的审批、申诉等流程，希望在今后的过程中加以了解，并合理迁移应用所学的知识。

软件工程是一门应用性较强的课程，绝非纸上谈兵，“纸上得来终觉浅，绝知此事要躬行”，理论学的再好，不会应用也是不可行的，只有做出可以落地的项目，才是真正地懂得了软件工程课程的意义所在。

# 面向对象的软件分析与设计

## UML 软件工具

利用现有的 **21 点游戏小程序**，使用面向对象方法做需求分析，画用例模型，掌握面向对象分析与设计方法。使用 Star UML 对类图进行绘制，并完成双向工程，检查制作的优劣程度。

### 绘制用例模型

在 21 点游戏小程序中，进行交互的 Actor 有：**庄家、闲家、服务器、硬件设备**。

庄家使用游戏系统完成游戏。

闲家使用游戏系统完成游戏。

庄家与闲家从游戏系统中执行各种操作（下注、确认、加倍、保险、拿牌、听牌、重开等），需要传递操作命令。

发牌人给庄家与闲家发牌，操作信息，传递操作命令。

服务器需要读取、存储、修改游戏产生的玩家和出牌信息。

硬件设备接收系统发生事件时的通知，向玩家发送当前游戏结果。

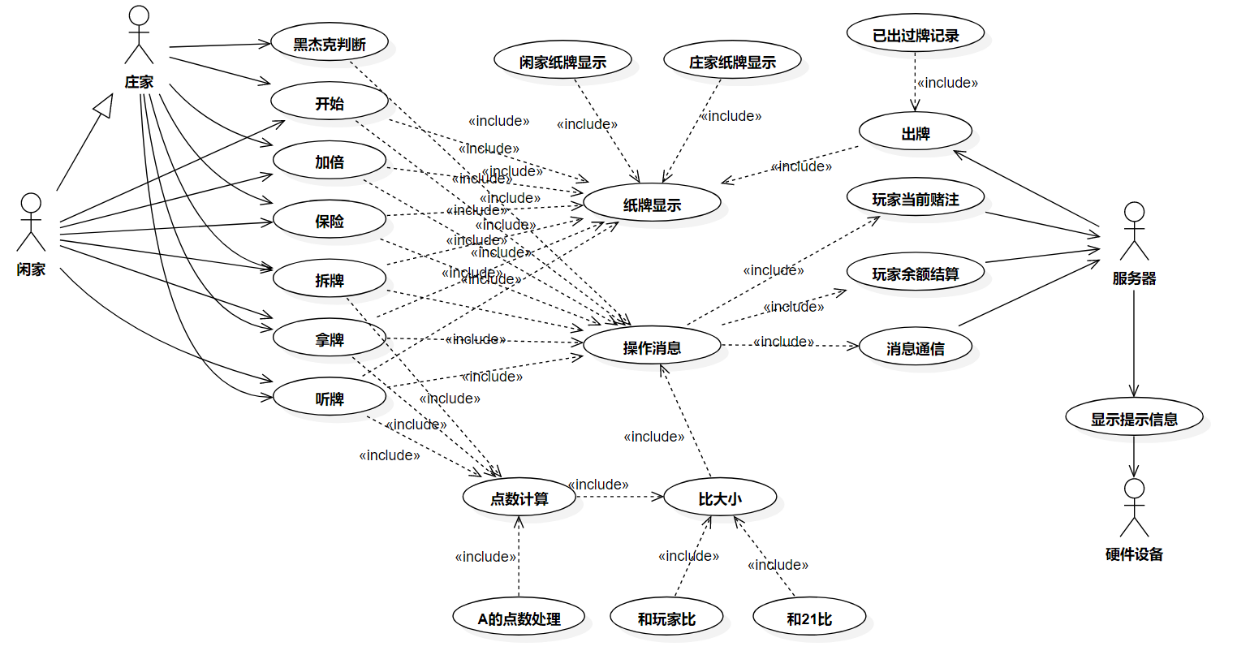


图 6 21 点小游戏用例模型

### 绘制类图

根据用例模型图进行分析与优化，利用Star UML 绘制出如下类图，由于第一次绘制进行双向工程的效果不好，将中文修改为英文，并重新对箭头方向和类型进行了检查，绘制了新的类图。

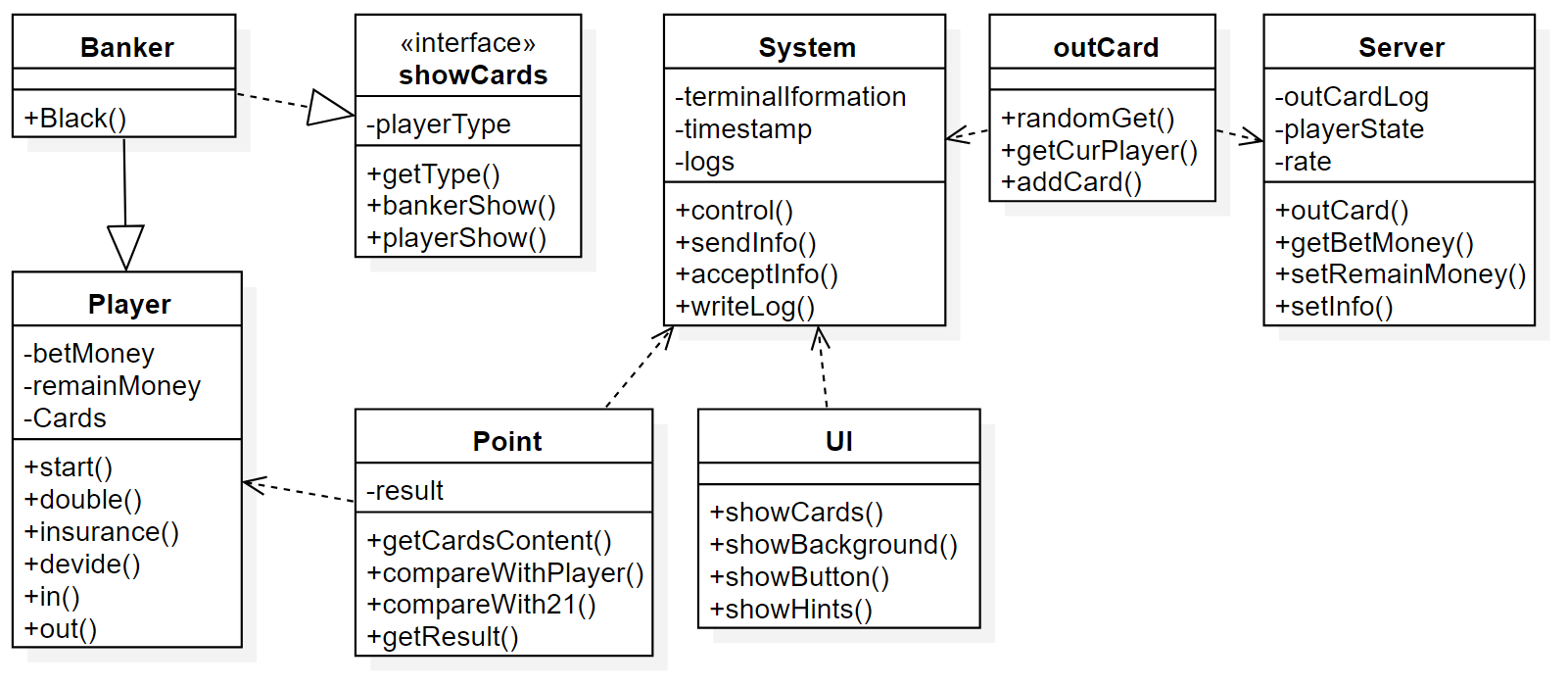


图 7 21 点小游戏类图

## UML双向工程

选择【Tools > Java > Generate Code】，选择所绘制的类图文件，随后生成了一系列java文件。打开其中的【Player.java】，【Banker.java】和【Server.java】，可以看到所生成的详细代码如下图所示。

// Player.java

import java.util.\*;  
public interface showCards {  
private void *playerType*;  
public void getType();  
 public void bankerShow();  
public void playerShow();  
}

// Banker.java

import java.util.\*;public class Banker extends Player implements showCards {  
 public Banker() {  
 }  
public void Black() {  
 *// TODO implement here* }  
public void getType() {  
 *// TODO implement here* }  
public void bankerShow() {  
 *// TODO implement here* }  
public void playerShow() {  
 *// TODO implement here* }  
public void getType() {  
 *// TODO implement here* }  
public void bankerShow() {  
 *// TODO implement here* }  
public void playerShow() {  
 *// TODO implement here* }  
}

// Server.java

import java.util.\*;  
public class Server extends outCard {  
public Server() {  
 }  
private void outCardLog;  
private void playerState;  
public void 游戏规则;  
private void rate;  
public void outCard() {  
 *// TODO implement here* }public void getBetMoney() {  
 *// TODO implement here* }public void setRemainMoney() {  
 *// TODO implement here* }  
public void setInfo() {  
 *// TODO implement here* }  
}

为了验证 UML 的逆向工程，首先尝试将正向工程中生成的代码逆向转为类 图。选择【Tools > Java > Reverse Code】后，选择刚才存放 .java 文件的文件夹， 进行逆向工程，可以看到，Start UML 可以自动为代码生成类图。

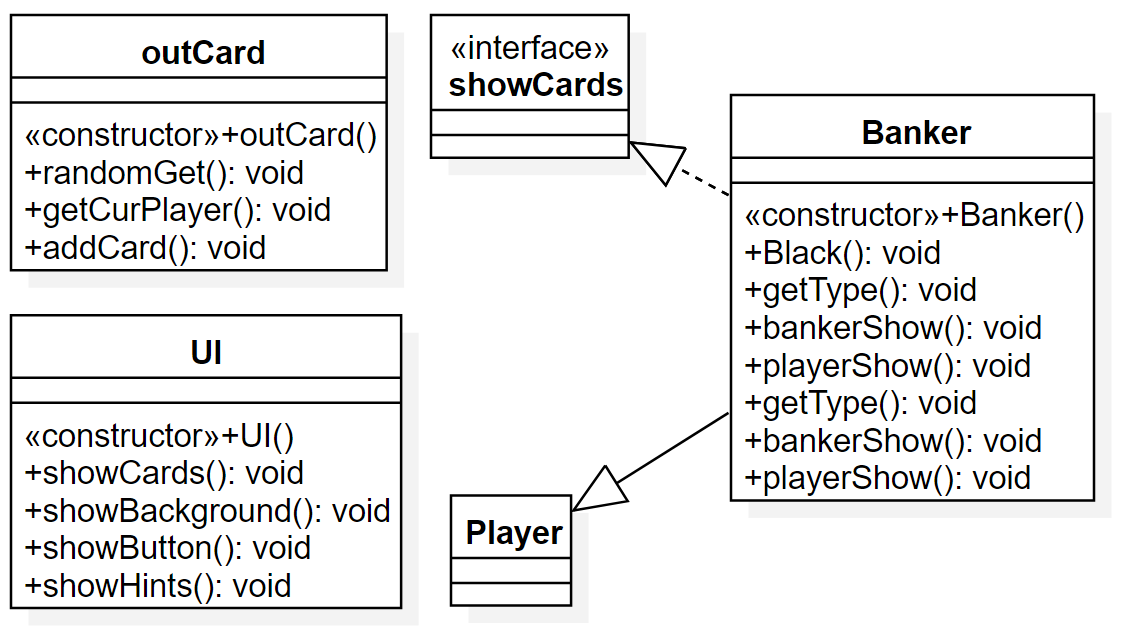


图 8 21 点小游戏逆向工程得到的类图

然而，比对自己绘制的 UML 图和逆向工程生成的 UML 图，可以发现类图在经过正向工程之后再反向，所生成的类图与原来的类图相比较有较多的缺失。分析这一现象，有可能是自己设计的类不够规范，不够全面导致的，效果并不是很完美，同时也没有考虑函数的返回值和参数，值得继续改进。

## 实验总结

这次实验，主要是学习了面向对象分析建模方法的步骤。结合着课堂上所学，通过不断分析，逐步完成对于问题的建模。同时我也感受到了，编写用例图不是简简单单随便写上几个方法和角色就行，图形和代码的规范，让我切身的感受到了编写一份规范的易懂又便于维护的代码非常重要，软件开发不仅仅是是敲代码，不是一件凭灵感肆意运行的事情。在软件方面，学习了对 Star UML 建模工具绘制类图的基础操作，感受了通过工具进行面向对象分析的流程。

本次实验涉及的操作较为简单，以后的过程中还要多加练习，增加自己对问题进行面向对象分析的熟练度。未来将学以致用，将软件面向对象的分析方法真正的在实际项目中加以运用，同时在平常学习中，逐步培养起规范编写代码的习惯，更好的开发软件。

# 集成编程环境与编码规范

## Python 语言集成开发环境调试方法与自动化工具模块

选择的代码如下，主要实现了基于 OpenCV 的对编码规则为 DICT\_7X7\_1000的 AcuroMaker 二维码进行图像信息识别。

*# -\*- coding:utf-8 \_\*-*import cv2  
import cv2.aruco as aruco  
import numpy as np  
import matplotlib.pyplot as plt  
'''检测图片中的编码点，并在图片上画出结果'''  
dict1 = aruco.getPredefinedDictionary(aruco.DICT\_7X7\_1000) *# 编码点的类型,与生成的时候对应*font = cv2.FONT\_HERSHEY\_SIMPLEX *# 使用默认字体*'''载入图片'''  
Img = cv2.imread('testBigImg\_new.png')  
h = Img.shape[0] *# 图片的尺寸*w = Img.shape[1]  
'''检测编码点，并在图片上显示检测结果'''  
corners, ids, rejectedImgPoints = aruco.detectMarkers(Img, dict1)  
c = np.array(corners) *# 检测出来的marker*id1 = np.array(ids).flatten() *# 对应marker的编码信息*ImgDetection = Img  
for i in range(0, c.shape[0]): *# 遍历每一个marker* for j in range(0, c[i][0].shape[0]): *# 遍历marker的4个角点* ImgDetection = cv2.circle(Img, (int(c[i][0][j][0]), int(c[i][0][j][1])), 30, (55, 255, 155), 10) *# 圈出角点* ImgDetection = cv2.putText(ImgDetection, str(id1[i]),  
 (int(c[i][0][0][0]),  
 int(c[i][0][0][1])),  
 font, 3, (255, 0, 0), 10) *# 写上id信息*plt.imshow(ImgDetection, 'gray')  
plt.show()  
cv2.imwrite('BigImg\_detect.png', ImgDetection)

PyCharm 调试可以点击，即可进入调试界面，效果如下。右侧是科学计算窗口，便于查看矩阵，方便更直观地观察图像处理过程，利于提高调试效率。

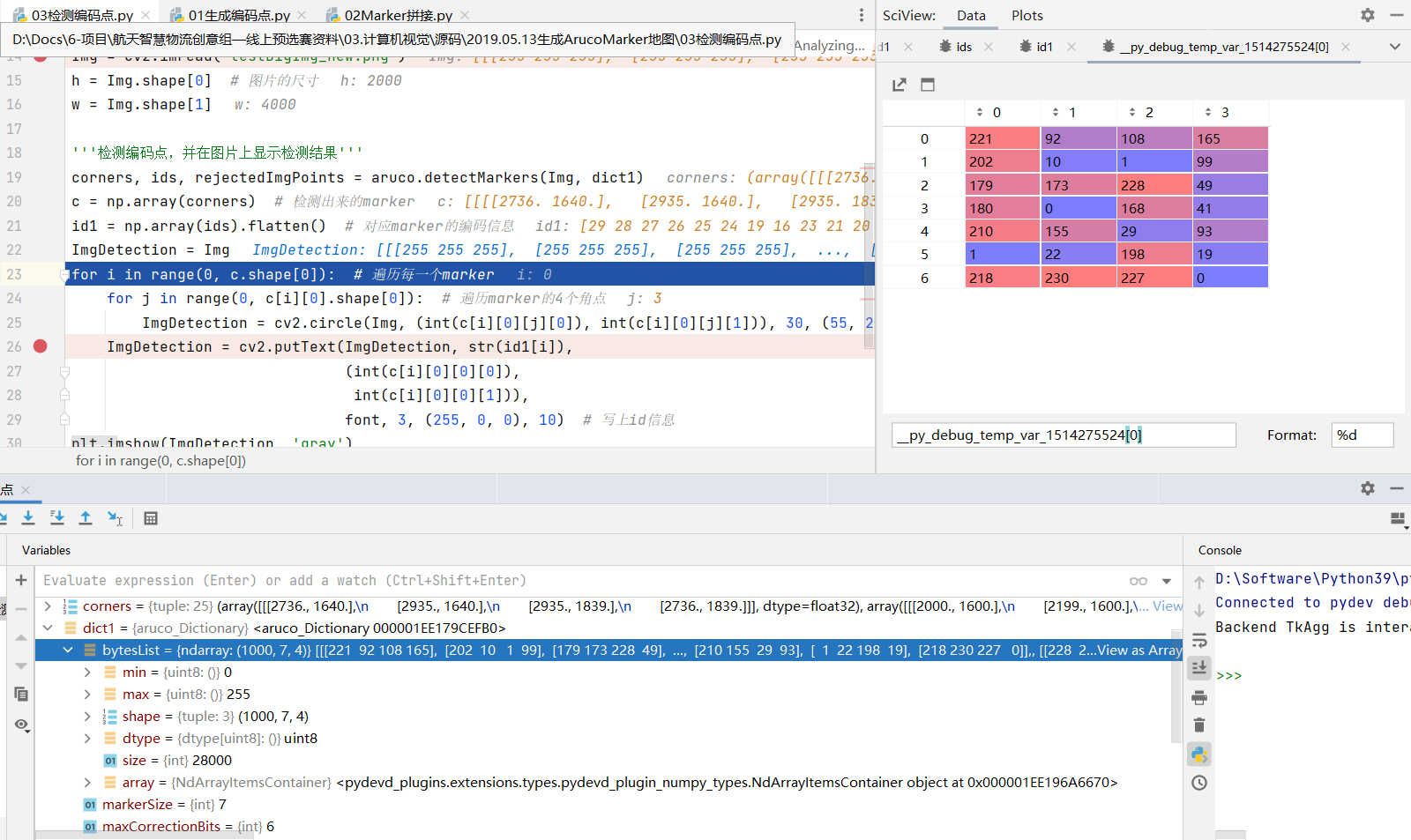


图 9 PyCharm 调试界面

调试情况及使用自动化工具进行分析、设计等工作。

## PHP 语言集成开发环境调试方法与自动化工具模块

### PhpStrom 调试情况

选择的代码如下，主要实现了键值对的输出与九九乘法表的输出。

<?php  
$x=array(1=>"Google", 2=>"Runoob", 3=>"Taobao");  
foreach ($x as $key => $value)  
{  
 echo "key 为 " . $key . "，对应的 value 为 ". $value . *PHP\_EOL*;  
}  
  
$j = 1;  
while($j<=9){  
 $i = 1;  
 while($i<=$j){  
 echo "{$i}x$j=".($i\*$j)." ";  
 $i++;  
 }  
 echo "<br />";  
 $j++;  
}

PhpStrom 调试可以点击，即可进入调试界面，效果如下。

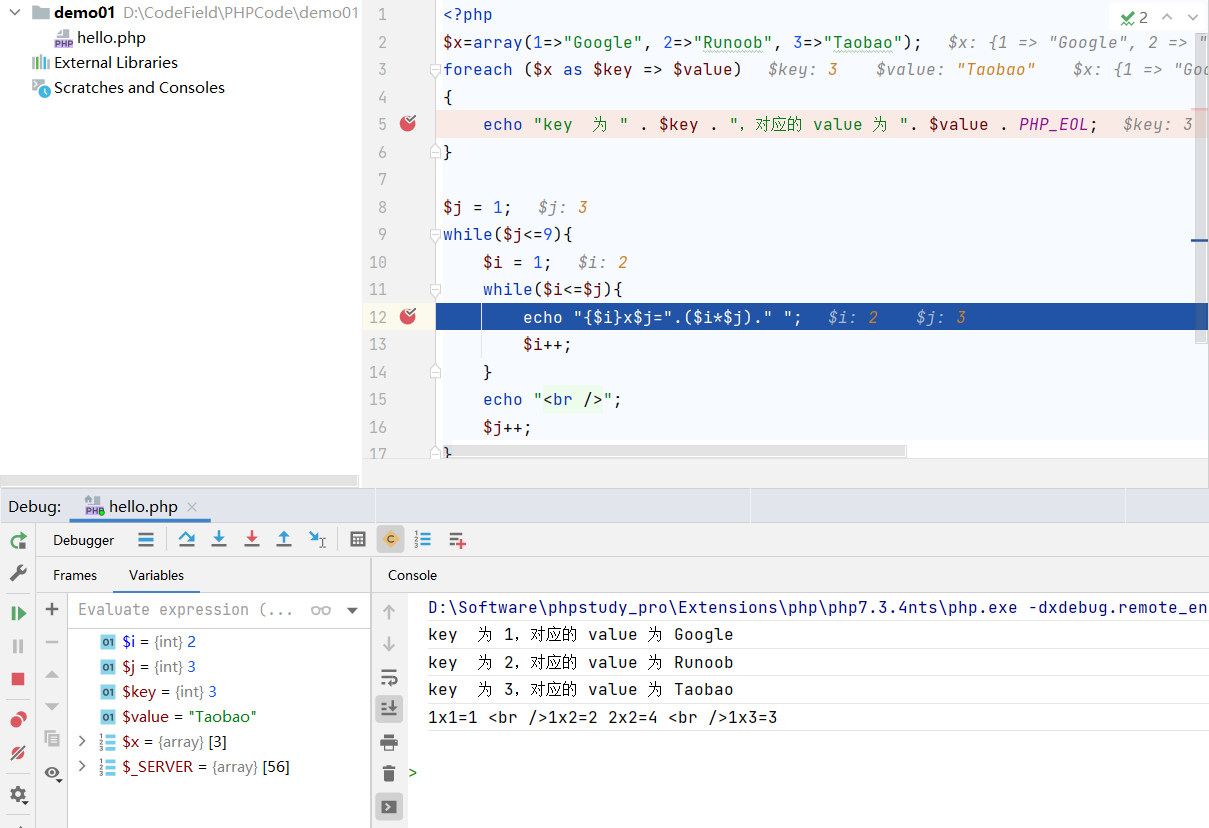


图 10 PhpStrom调试界面

使用自动化工具进行分析、设计等工作。？？？？

## Java 语言集成开发环境调试方法与自动化工具模块

选择的代码如下，主要实现了模拟多人过独木桥的模拟。独木桥每次只能通过一个人，每个人通过木桥的时间为1秒，随机生成10个人过此独木桥，显示一下每次通过独木桥人的姓名。

import java.util.HashSet;  
import java.util.Set;  
class People implements Runnable{  
 @Override  
 public void run() {  
 synchronized(this) {  
 System.*out*.println(Thread.*currentThread*().getName()+"正在过桥");  
 try {  
 Thread.*sleep*(1000);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 System.*out*.println(Thread.*currentThread*().getName()+"已通过桥");  
 }  
 }  
  
}  
public class demoThread {  
 public static void main(String[] args) {  
 String[] name= {"一","二","三","四","五","六","七","八","九","十"};  
 Set<Integer> set=new HashSet<Integer>();  
 while(set.size()<10) {  
 set.add((int)(Math.*random*()\*10));  
 }  
 People people =new People();  
 for(int a:set) {  
 Thread thread =new Thread(people,name[a]);  
 thread.start();  
 }  
 }  
}

IDEA 调试可以点击，即可进入调试界面，效果如下。

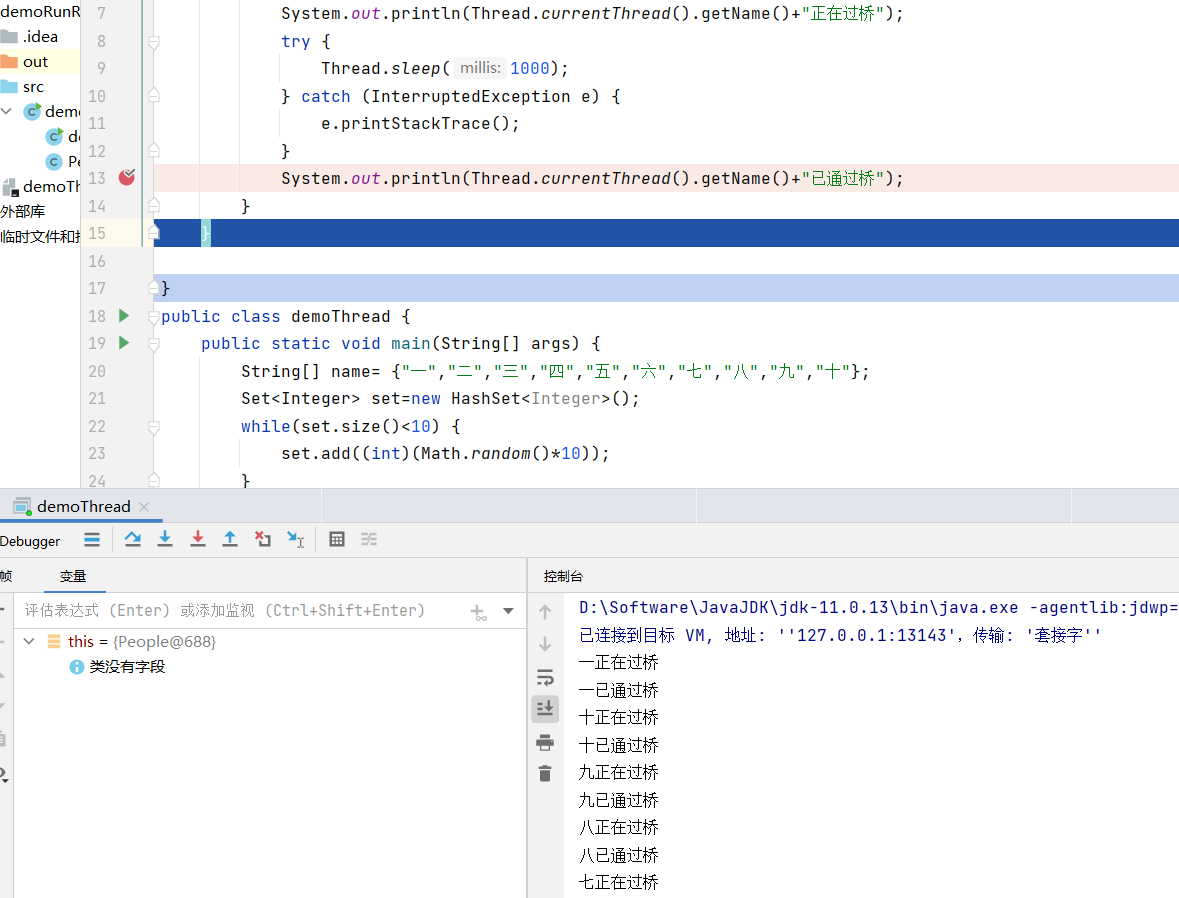


图 11 IDEA调试界面

调试情况及使用自动化工具进行分析、设计等工作。

## Docker 虚拟化工具

部署情况。

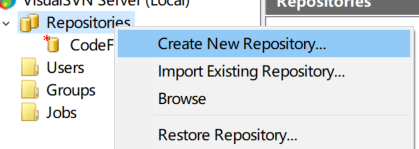
## 实验总结

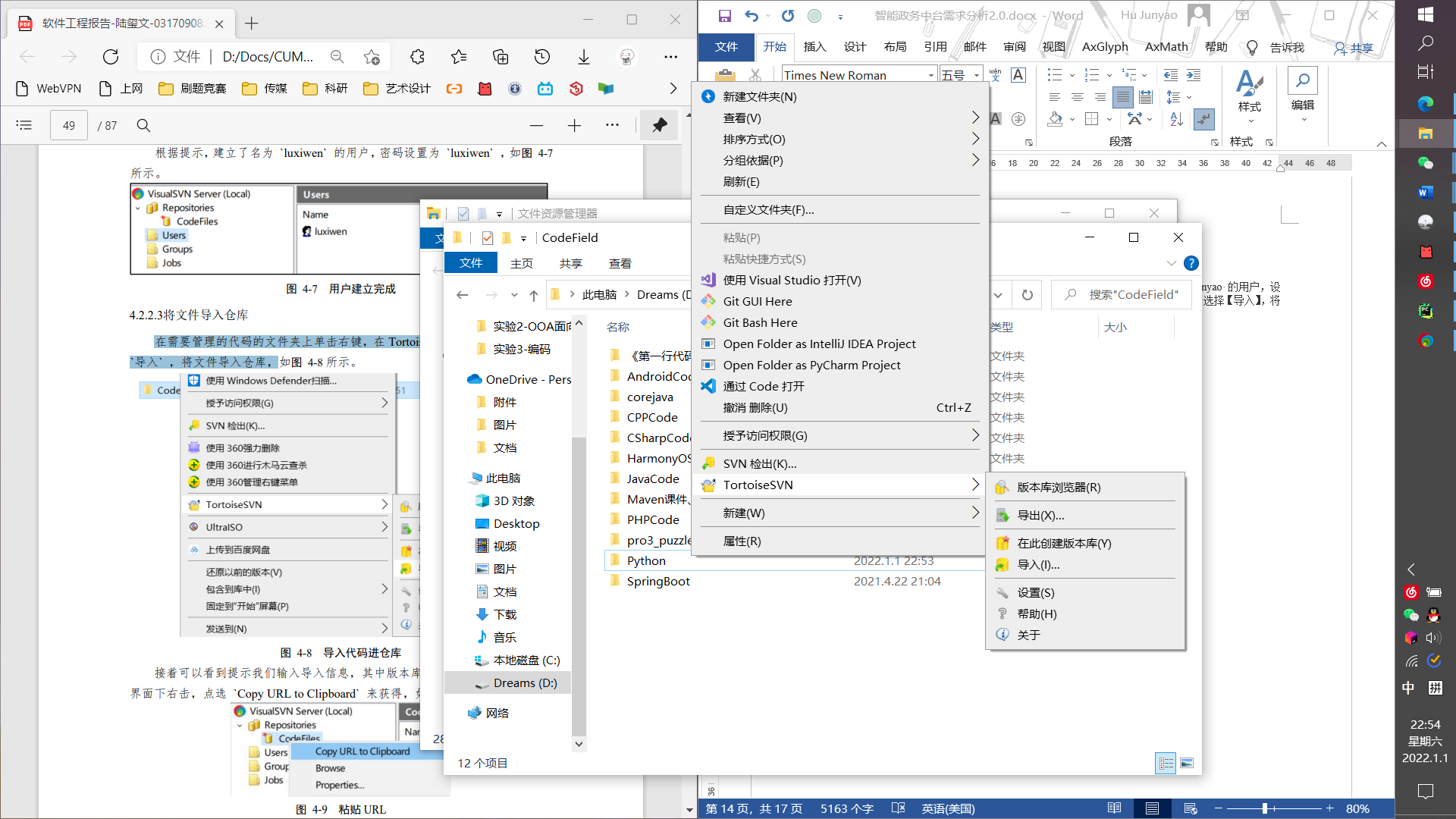
这次实验，学习 PHP 、 Java、 Python 集成编程环境，熟悉各语言的编码规范。学习了 JetBrains 全家桶中的 PhpStrom 、 IDEA 、 PyCharm，结合着课堂上所学，更进一步的明白了代码编写不能仅凭个人志趣，要符合规范。此外，实验中在阅读代码规范的过程中，发现了许多自己以前不注意的点。其中，很大一部分的原因在于以往所编写的代码太小不够完整。对比规范，让我切身的感受到了编写易懂又便于维护的代码是多么的重要又不容易，让我不再想当然的觉得软件开发只是敲代码，只是一件凭灵感肆意运行的事情。未来将学以致用，将软件面向对象的分析方法真正的在实际项目中加以运用，同时在平常学习中，逐步培养起规范编写代码的习惯，更好的开发软件。此外，我觉得 JetBrains 系列软件的界面风格非常统一，学会了一个开发IDE就能很快迁移到另外的IDE上面，同时界面交互非常友好，这也是我们设计软件UI的时候需要注意的点。

# 源代码版本管理

## 独立的源代码版本管理工具（SVN）

首先在 VisualSVN Sever 建立自己的空代码仓库 CodeField， 创建名为 hujunyao 的用户，设置好密码，在需要管理的代码的文件夹【Python】上单击右键，在 TortoiseSVN 的子菜单下，选择【导入】，选择将文件导入仓库。仓库 URL 可以通过在 VisualSVN Sever 服务器界面下右击，【Copy URL to Clipboard】来获得，并填写好导入信息，便于管理。点击【确定】，即可完成文件的导入。



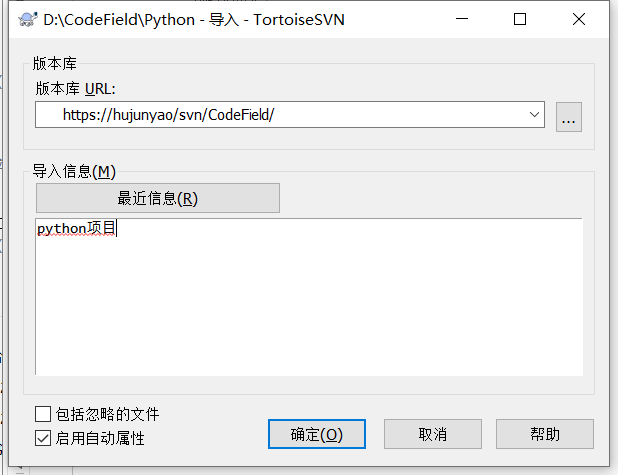
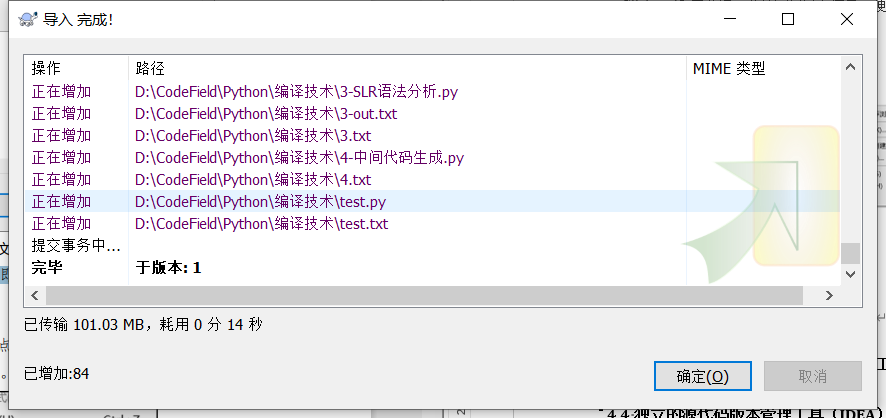
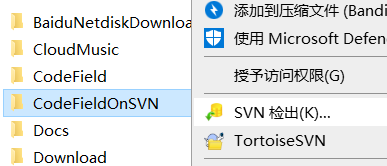
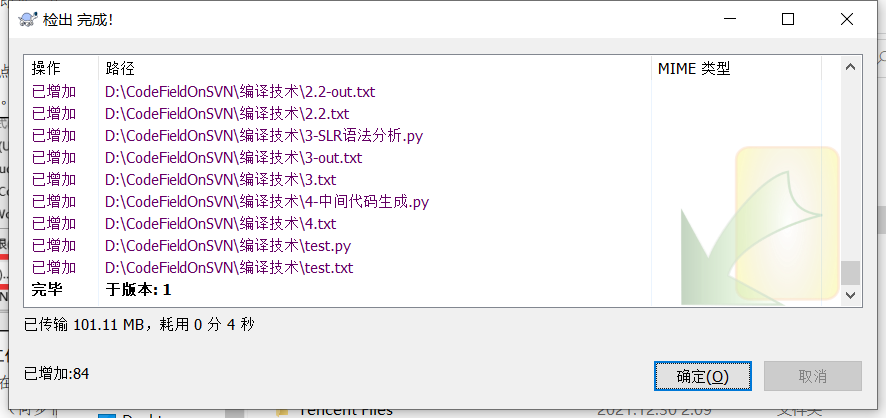
 

图 12 将文件导入仓库的流程

在本地的【CodeFieldOnSVN】文件夹上，右击点选【SVN 检出】，设置好服务器和本地目录即可对代码文件进行检出（同步）。选择确定之后，即可进行代码的同步，在同步完成后，可以看到相应的完毕提示界面。同步完成后的文件夹及其子文件夹的图标上都将有绿色对号出现，表明已同步。

部署情况。

## 独立的源代码版本管理工具（Git）

## 集成开发环境中的源代码版本管理工具（VS）

部署情况。

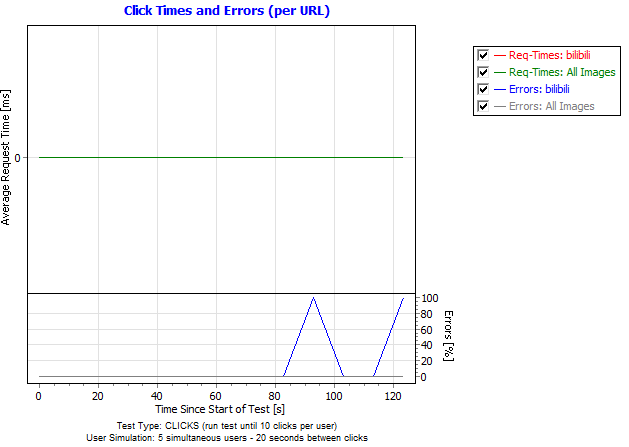
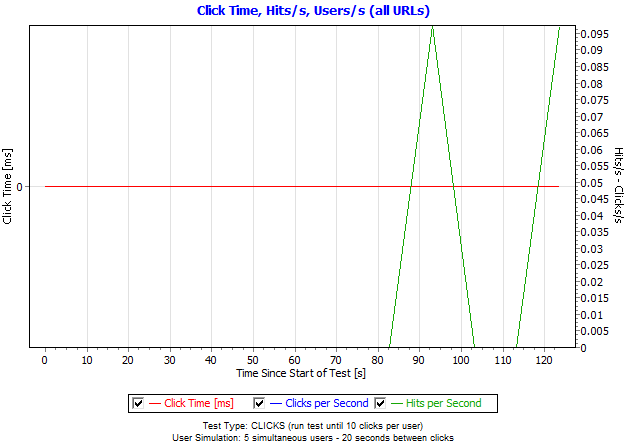
## 独立的源代码版本管理工具（IDEA）

## 实验总结

# 自动软件测试

* 1. **Webstress 性能测试工具**

使用 Webstress 测试 Bilibili 视频网。测试完成后可以打开相应的 HTML 格式报告或是 Word 格式报告。在 HTML 报告中，可以查看点击次数和错误、点击时间速度、请求和数据传输、转移数据和系统存储等，数据如下。

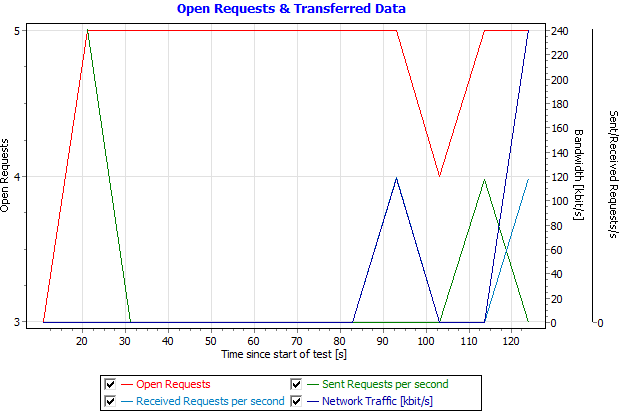
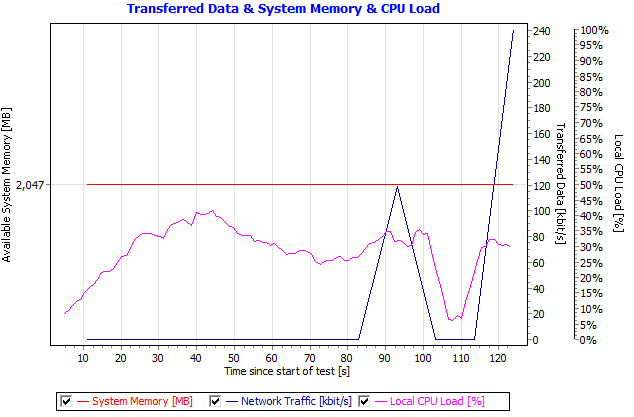
 

图 13 Webstress 测试 B 站压力部分数据结果

## 独立的自动测试软件 NUnit / JUnit

部署及测试情况。

## 华为 DevCloud 云测模块

测试情况。

## PHP 集成开发环境测试模块

测试情况。

## Python 集成开发环境测试模块

测试情况。

## Java 集成开发环境测试模块

在 IDEA 中有两种构造测试模块的方法。

一种是自己创建类，例如在 src 下创建一个包命名为 testjunit，包内创建类。在 src 的同级目录下创建一个文件夹名为 test，右键选择 Test Sources Root，于文件夹中创建与 testjunit 包名称一致的包，并创建测试类。在TestDemo.class 中创建测试方法：单元测试方法时候，方法命名规则：public void 方法名() {}。方法名不能有参数。使用注解方式运行测试方法，在方法的上面：@Test；如无注解，则不能运行。

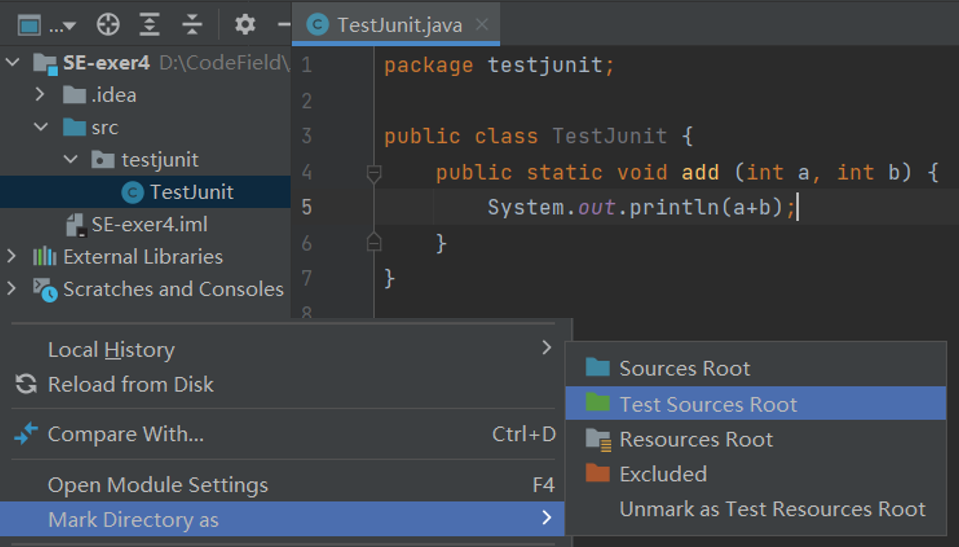
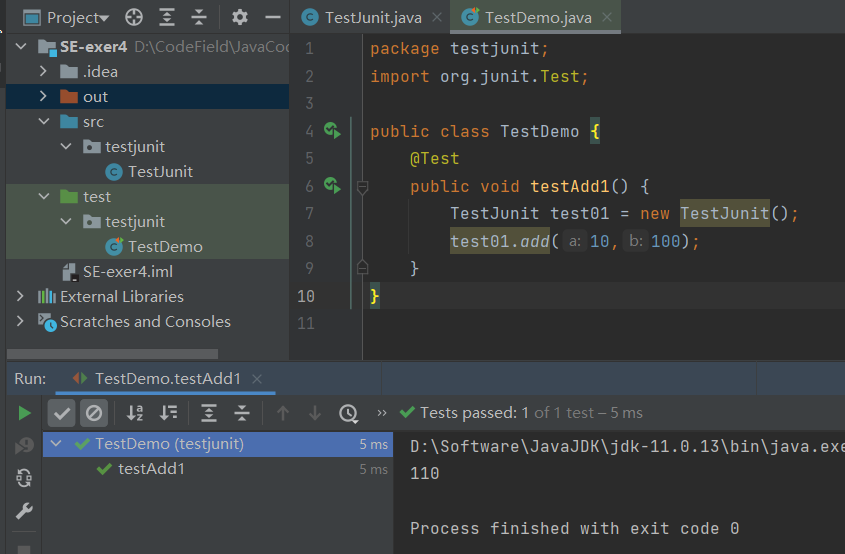
 

图 14 手动构建测试类的方法及真出口测试结果

再写一个 mul 函数，使用自动生成方法生成测试类，快捷键是Ctrl+Shift+T。编写测试代码，测试错误的显示情况如下。

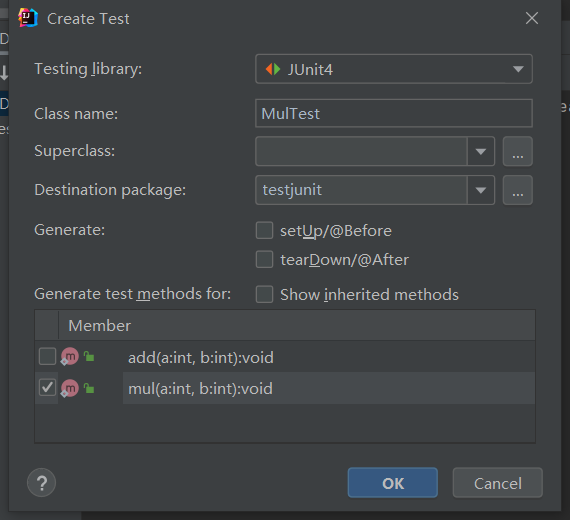
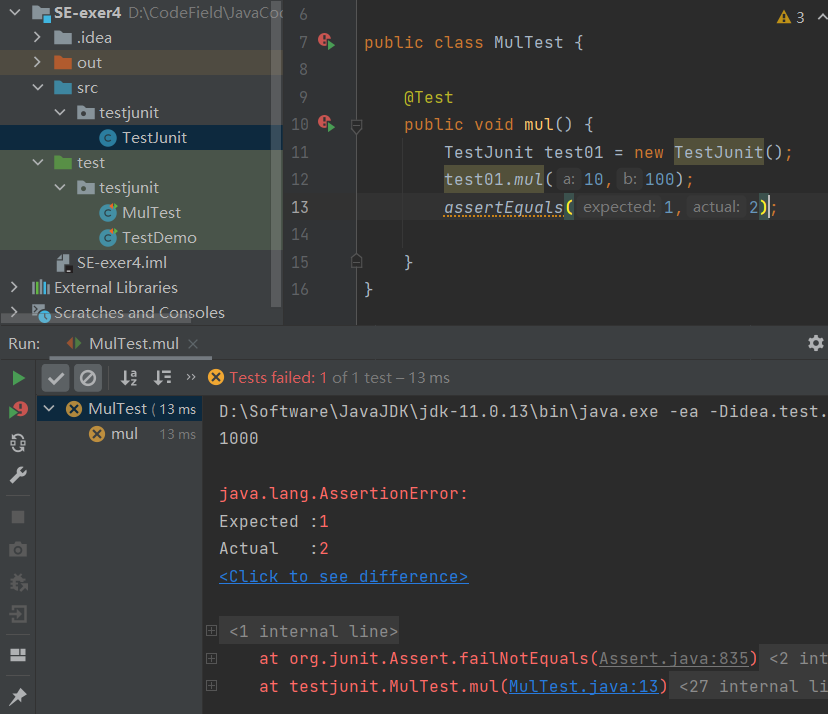
 

图 15 使用自动生成测试类的方法及假出口测试结果

## 实验总结

# PowerDesigner软件设计

## Power Designer设计

设计情况。

## 实验总结