

网站购物系统

**测试计划书**



北京航空航天大学

2016-12-20

版本变更历史

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 版本 | 提交日期 | 主要编制人 | 审核人 | 版本说明 | 贡献 |
| v1.0 | 2016-12-20 | 孙镜涵 | 嵇文麒，泰阳 | 初版 | 孙镜涵（主稿）嵇文麒，泰阳（审核） |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

目 录

1引言 1

1.1编写目的 1

1.2背景 1

1.3定义 1

2计划 2

2.1软件说明 2

2.2测试内容 2

2.3测试1（正确性测试） 5

2.3.1进度安排 5

2.3.2条件 5

2.3.3测试资料 5

2.3.4测试培训 5

2.4测试2（边界值测试） 5

2.4.1进度安排 5

2.4.2条件 6

2.4.3测试资料 6

2.4.4测试培训 6

2.5测试3（容错性测试） 6

2.5.1进度安排 6

2.5.2条件 6

2.5.3测试资料 6

2.5.4测试培训 7

2.6测试4（性能与效率测试） 7

2.6.1进度安排 7

2.6.2条件 7

2.6.3测试资料 7

2.6.4测试培训 7

2.7测试5（易用性测试） 7

2.7.1进度安排 8

2.7.2条件 8

2.7.3测试资料 8

2.7.4测试培训 8

2.8测试6（Javadoc文档测试） 8

2.8.1进度安排 8

2.8.2条件 8

2.8.3测试资料 9

2.8.4测试培训 9

3 测试计划说明 9

3.1测试1（正确性测试） 9

3.2测试2（边界值测试） 11

3.3测试3（容错性测试） 12

3.4测试4（性能与效率测试） 12

3.5测试5（易用性测试） 12

3.6测试6（Javadoc测试） 13

4评价准则 14

4.1范围 14

4.2数据整理 14

4.3尺度 15

**软件测试计划书**

1引言

1.1编写目的

测试文件计划将有助于我们发现软件潜在漏洞，并将相关问题反馈给开发人员，在软件开发中有剧组轻重的地位，详细测试计划的建立便于今后的开发以及重构工作。

读者对象：

1. 开发人员
2. 测试人员
3. 文档审核人员

1.2背景

本系统的一般特性是能够实现用户在本网站上购物的全过程。本系统面向用户，依照现实中网上购物的基本流程，满足用户的浏览商品，下单订货，退单，查询自己订单，提交评价的需求，依照用户浏览，添加到购物车，提交订单，支付，收货，评价的流程来进行设计。用户需注册，本系统负责保管用户信息，提供商品目录。

开发方：计算机学院软件工程实践小组。

支持机构：北京航空航天大学软件工程研究所。

相关文档：《项目开发计划》、《软件设计说明书》、《软件测试计划》

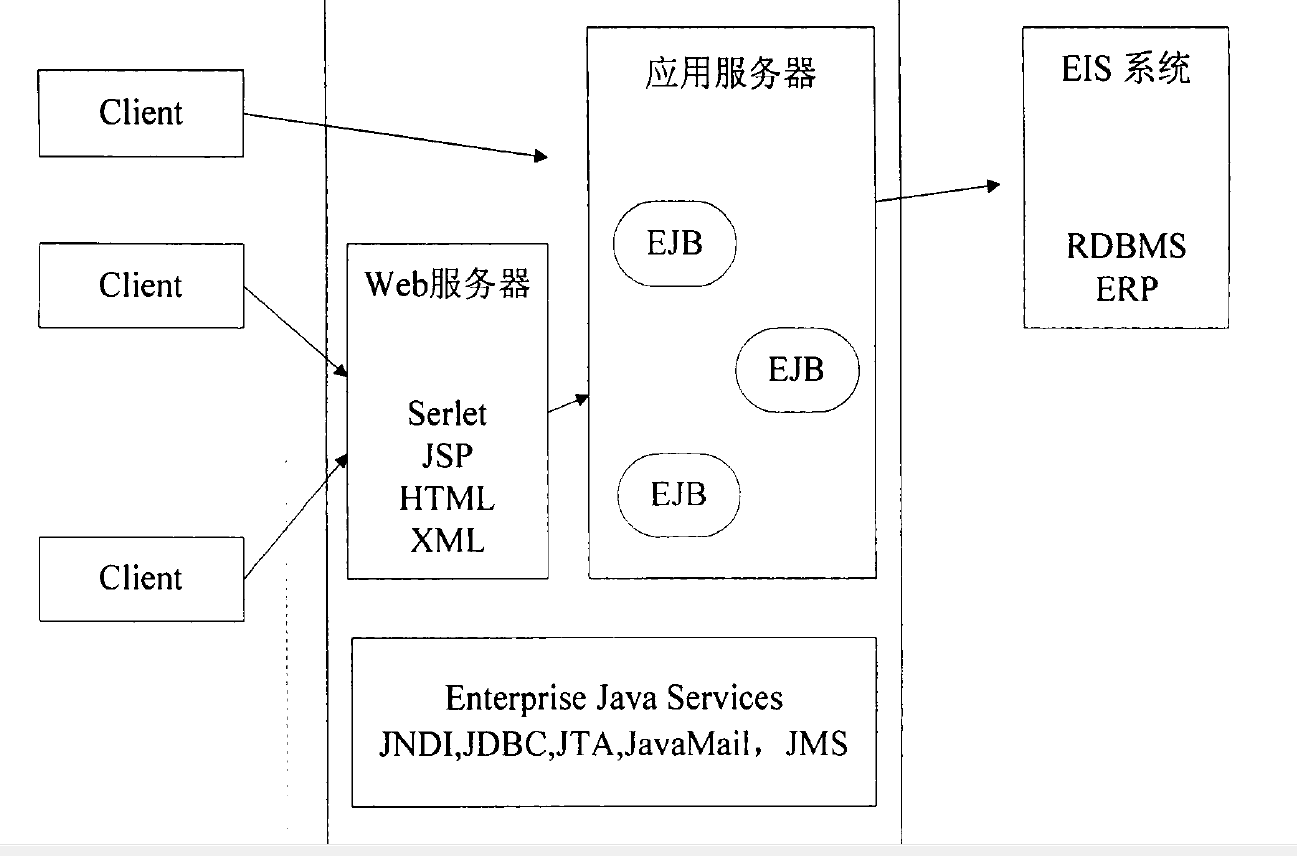
1.3定义

J2EE: J2EE（Java 2 Platform, Enterprise Edition）是一个为大企业主机级的计算类型而设计的Java平台。Sun微系统（与其工业伙伴一起，例如IBM）设计了J2EE，以此来简化在瘦客户级环境下的应用开发。由于创造了标准的可重用模块组件以及由于构建出能自动处理编程中多方面问题的等级结构，J2EE简化了应用程序的开发，也降低了对编程和对受训的程序员的要求。

2计划

2.1软件说明

系统架构，我们采用一个如下图所示的三层架构。三层架构在涉及到网络的开发中很常见，它将系统分割为自下而上的数据访问层（DAL）、业务逻辑层（BLL）和表示层（UI）。表现层即为展现给用户的UI界面，业务逻辑层是软件的核心了，包括处理用户请求和访问操作数据层等，数据访问层很直观了就是直接管理数据库的。在我们的项目开发中，我们将下图所画的三级结构表示为客户端层，模块层和数据层。



2.2测试内容

1.正确性测试

正确性测试又称功能测试，它检查软件的功能是否符合规格说明。由于正确性是软件最重要的质量因素，所以其测试也最重要。

基本的方法是构造一些合理输入，检查是否得到期望的输出。这是一种枚举方法。倘若枚举空间是无限的，那可惨了，还不如回家种土豆有盼头。测试人员一定要设法减少枚举的次数，否则没好日子过。关键在于寻找等价区间，因为在等价区间中，只需用任意值测试一次即可。等价区间的概念可表述如下：

记（A, B）是命题f(x) 的一个等价区间，在（A, B）中任意取x1进行测试。

如果f (x1) 错误，那么f (x) 在整个（A, B）区间都将出错。

如果f (x1) 正确，那么f (x) 在整个（A, B）区间都将正确。

2.边界值测试。即采用定义域或者等价区间的边界值进行测试。

例如测试 的一段程序。凭直觉等价区间应是（0, 1）和（1, +∞）。可取x=0.5以及x=2.0进行等价测试。再取 x=0以及x=1进行边界值测试。

有一些复杂的程序，我们难以凭直觉与经验找到等价区间和边界值，这时枚举测试就相当有难度。

在用“白盒测试”方式进行正确性测试时，有个额外的好处：如果测试发现了错误，测试者（开发人员）马上就能修改错误。越早改正错误，付出的代价就越低。所以大多数软件公司要求程序员在写完程序时，马上执行基于单步跟踪的“白盒测试”。

3.容错性测试

容错性测试是检查软件在异常条件下的行为。容错性好的软件能确保系统不发生无法意料的事故。

比较容错性测试通常构造一些不合理的输入来引诱软件出错，例如：

（1）输入错误的数据类型，如“猴”年“马”月。

（2）输入定义域之外的数值，上海人常说的“十三点”也算一种。

（3）输入错误的函数调用。

容错性测试俗称鲁棒性测试。

4.性能与效率测试

性能与效率测试主要是测试软件的运行速度和对资源的利用率。我们时而关心测试的“绝对值”，如数据送输速率是每秒多少比特，时而关心测试的“相对值”，如某个软件比另一个软件快多少倍。

在获取测试的“绝对值”时，我们要充分考虑并记录运行环境对测试的影响。例如计算机主频，总线结构和外部设备都可能影响软件的运行速度；若与多个计算机共享资源，软件运行可能慢得像蜗牛爬行。

在获取测试的“相对值”时，我们要确保被测试的几个软件运行于完全一致的环境中。硬件环境的一致性比较容易做到（用同一台计算机即可）。但软件环境的因素较多，除了操作系统，程序设计语言和编译系统对软件的性能也会产生较大的影响。如果是比较几个算法的性能，就要求编程语言和编译器也完全一致。

性能与效率测试中很重要的一项是极限测试，因为很多软件系统会在极限测试中崩溃。

5.易用性测试

易用性测试没有一个量化的指标，主观性较强。一般认为，如果用户不翻阅手册就能使用软件，那么表明这个软件具有较好的易用性。

6.文档测试

文档测试主要检查文档的正确性、完备性和可理解性。正确性是指不要把软件的功能和操作写错，也不允许文档内容前后矛盾。

2.3测试1（正确性测试）

参与单位：全体测试人员。

测试部位：程序正常使用功能

2.3.1进度安排

测试日期：12-21至12-22。

工作内容：针对每一类撰写单元测试，保证类功能的实现。

2.3.2条件

程序资源：系统实现程序，JUNIT4测试框架

人力资源：测试人员3人。

2.3.3测试资料

被测试程序：全部类。

输入输出举例：在setUp()中给测试类赋以初值，在随后的方法中进行测试。

2.3.4测试培训

JUNIT4的使用。

2.4测试2（边界值测试）

参与单位：全体测试人员。

测试部位：程序在数值上的极端使用功能。

2.4.1进度安排

测试日期：12-22至12-23。

工作内容：测试软件各类在数值上的极端使用功能是否正常。

2.4.2条件

程序资源：系统实现程序，JUNIT4测试框架

人力资源：测试人员3人。

2.4.3测试资料

被测试程序：全部类。

输入输出举例：在setUp()中给测试类赋以初值，在随后的方法中进行测试。

在测试时，需要考虑每一类的极端使用情况，进行边界值测试。

2.4.4测试培训

JUNIT4的使用。边界值的选取。

2.5测试3（容错性测试）

参与单位：全体测试人员。

测试部位：程序在错误使用时的鲁棒性。

2.5.1进度安排

测试日期：12-23至12-24。

工作内容：测试软件在极端条件下的运行情况。

2.5.2条件

程序资源：系统实现程序，JUNIT4测试框架，Jenkins持续集成测试环境

设备资源：云端测试服务器

人力资源：测试人员3人。

2.5.3测试资料

有关文件：SDP-项目测试计划。

被测试程序：所开发系统。

输入输出举例：用户名错误输入；报错。

2.5.4测试培训

Jenkins测试环境搭建培训。

2.6测试4（性能与效率测试）

参与单位：全体测试人员。

测试部位：软件执行时运行效率。

2.6.1进度安排

测试日期：12-22至12-26。

工作内容：测试软件执行时的运行时间并记录。

2.6.2条件

程序资源：系统实现程序，JUNIT4测试框架，Jenkins持续集成测试环境

设备资源：云端测试服务器

人力资源：测试人员3人。

2.6.3测试资料

被测试程序：所开发系统系统，核心算法代码。

输入输出举例：关键代码段执行前时间，执行后时间；总消耗时间

2.6.4测试培训

无测试培训。

2.7测试5（易用性测试）

参与单位：全体测试人员。

测试部位：网站购物系统。

2.7.1进度安排

测试日期：12-24至12-25。

工作内容：请志愿者仅根据文档使用本系统（或不根据文档）。

2.7.2条件

程序资源：系统实现程序

设备资源：个人电脑PC

人力资源：志愿者3人。

2.7.3测试资料

有关文件：SDP-项目测试计划。

被测试程序：所开发系统。

输入输出举例：用户使用的正常输入；程序正确执行。

2.7.4测试培训

志愿者管理培训。

2.8测试6（Javadoc文档测试）

参与单位：全体测试人员。

测试部位：网站购物系统。

2.8.1进度安排

测试日期：12-25至12-26。

工作内容：对文档语法进行检查，并检测文档是否正确的描述了程序功能。

2.8.2条件

设备资源：使用两台图书馆扫描仪器以及电脑，预定使用时间12-25至12-26。

程序资源：扫描仪器程序、图书馆管理系统。

人力资源：用户1人，测试人员4人，测试人员全员参与了软件开发过程，用户未接触过该软件。

2.8.3测试资料

程序资源：系统实现程序，Javadoc。

设备资源：个人电脑PC

人力资源：测试人员3人。

2.8.4测试培训

无。

3 测试计划说明

3.1测试1（正确性测试）

**3.1.1控制**：

初始化setUp()：

image = new BufferedImage(200, 200, BufferedImage.TYPE\_INT\_ARGB);

Graphics2D gra = image.createGraphics();

// transform shape to a coorniate x[-5,5], y[-5,5].

AffineTransform transform = new AffineTransform();

transform.scale(image.getWidth() / 10.0, image.getHeight() / 10.0);

transform.translate(0, 10.0);

transform.scale(1, -1);

transform.translate(5.0, 5.0);

gra.transform(transform);

drawerInit(gra);

**3.1.2输入**：

@Test

public void renderBoxShape() throws Exception {

drawer.draw(new Box2D(0, 0, 3, 4));

ImageUtils.assertImageSimilar(

ImageIO.read(SolidBox2DDrawerTest.class.getResourceAsStream("SolidBox2DDrawer\_base.png")),

image);

}

@Test

public void renderBoxOverflow() throws Exception {

drawer.draw(new Box2D(-10, -10, 8, 9));

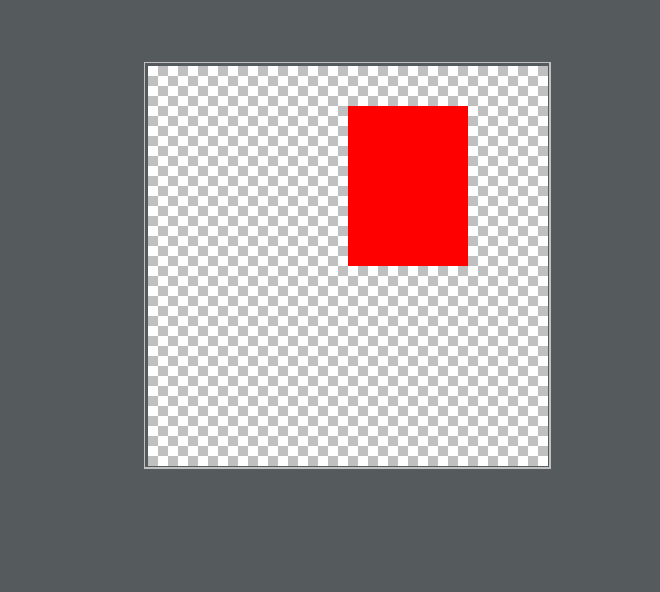
ImageUtils.assertImageSimilar(

ImageIO.read(SolidBox2DDrawerTest.class.getResourceAsStream("SolidBox2DDrawer\_base\_overflow.png")),

image);

}

**3.1.3输出（示例）**：

网站图标模型

**3.1.4过程**：

调用drawer类生成图标

3.2测试2（边界值测试）

**3.2.1控制**：

输入用户操作的可能边界值

**3.2.2输入**：

@Test

public void getShape() throws Exception {

}

@Test

public void determineNext() throws Exception {

baseAgent.determineNext();

System.out.println(baseAgent.toString());

}

**3.2.3输出**：

用户的可能行为

+++++++++++++++++++++++++++++++++++++@@+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

+++++++++++++++++++++++++++++++++++++@+@@@@@++++++++++++++++++++++++++++++++++++++++++++++++++++++++

+++++++++++++++++++++++++++++++++++@@+@+@@@+@+++++++++++++++++++++++++++++++++++++++++++++++++++++++

OOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOO+++OOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOO

OOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@+@OOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOO

OOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOO

++++++++++++++++++++++++++++++++OOOOOOOOOOOOOOOOO+++++++++++++++++++++++++++++++++++++++++++++++++++

++++++++++++++++++++++++++++++++OOOOOOOOOOOOOOOOO+++++++++++++++++++++++++++++++++++++++++++++++++++

**3.2.4过程**：

进入agent类生成行为

3.3测试3（容错性测试）

**3.3.1控制**：

输入错误的登录用户名

**3.3.2输入**：

sunjh1999

**3.3.3输出**：

label: 登陆错误

**3.3.4过程**：

进入login模块执行登录动作

3.4测试4（性能与效率测试）

**3.4.1控制**：

大批量登录，登出，资源占用操作

**3.4.2输入**：

登录 登出 页面访问 10000pps

**3.4.3输出**：

正常执行功能

**3.4.4过程**：

3.5测试5（易用性测试）

**3.5.1控制**：

志愿者在培训之后使用程序，并做检查如下：

用例1：符合标准和规范

最重要的用户界面要素是软件符合现行的标准和规范——或者有真正站得住脚的不符合的理由。

注意：如果测试在特定平台上运行的软件，就需要把该平台的标准和规范作为[产品说明书](http://baike.baidu.com/subview/550727/550727.htm)的补充内容。像对待产品说明书一样，根据它建立[测试用例](http://baike.baidu.com/view/106882.htm)。

这些标准和规范由软件易用性专家开发。它们是经由大量正规测试、使用、尝试和错误而设计出的方便用户的规则。

也并非要完全遵守准则，有时开发小组可能想对标准和规范有所提高。

平台也可能没有标准，也许测试的软件就是平台本身。

在这种情况下，设计小组可能成为软件易用性标准的创立者。

用例2：直观

用户界面是否洁净、不唐突、不拥挤？

UI的组织和布局合理吗？

有多余功能吗？

帮助系统有效吗？

用例3：一致

如果软件或者平台有一个标准，就要遵守它。如果没有，就要注意软件的特性，确保相似的操作以相似的方式进行。

快捷键和菜单选项

术语和命名

听众

诸如OK和Cancel按钮的位置。

用例4：灵活

多种视图的选择：

状态跳转

状态终止和跳过

数据输入和输出

用例5：舒适

软件使用起来应该舒适

3.6测试6（Javadoc测试）

**3.6.1控制**：

使用Teamcity集成测试环境对javadoc语法进行检查

**3.6.2输入**：

/\*\*

\* 代表一个场景参数的具体赋值。

\* 场景参数值的主要作用是存储一个特定类型的参数的赋值，

\* 并且将这个赋值应用于场景。

\*

\* @param <T> 内部表示值的类型。

\*/

**3.6.3输出**：

正确（T）

4评价准则

4.1范围

本测试计划的测试范围为：

* 类和接口所有功能的完整性和正确性；

登录功能，注册功能，购物车管理，商品评论管理，商品查询，购物功能，退货功能等。

* 系统的鲁棒性；
* 系统的易用性。
* 系统文档的正确性。
* 系统的性能与效率。

本测试计划的局限为：

* 没有分系统功能进行测试；

4.2数据整理

所有模块和文档正确性的测试结果均记录为正确与否。

压力和性能测试的结果由测试类输出结果到指定文件。

4.3尺度

|  |  |  |
| --- | --- | --- |
| 测试项目 | 输出结果与预期输出之间容许的偏差范围 | 允许中断或停机的最大次数 |
| 正确性测试 | 0 | 0 |
| 边界值测试 | 0 | 0 |
| 容错性测试 | 0 | 1 |
| 性能与效率测试 | 20~40% | 0 |
| 易用性测试 | 一般~易用 | - |
| Javadoc测试 | 0 | - |