

应用软件中自动化测试的实现

贾小云¹, 韩 薇²

(1 陕西科技大学 电气与信息工程学院, 陕西 西安 710021; 2 西安电力电子技术研究所, 陕西 西安 710061)

摘 要: 在介绍了自动化测试的重要性的基础上, 提出了如何实现自动化测试, 结合实例重点讨论了实现单元测试和功能测试自动化的方法, 最后讨论了使用 Jmeter 实现性能测试自动化。这些方法在基于 J2EE 企业应用软件的测试中取得了良好的效果。

关键词: 自动化测试; Cactus; 测试组; Jmeter

中图分类号: TP393.41

文献标识码: A

文章编号: 1000-7180(2010)09-0165-03

Realize Tests Automatically in Application Software

JIA Xiao-yun¹, HAN Wei²

(1 College of Electrical and Information Engineering, Shaanxi University of Science and Technology, Xi'an 710021, China; 2 Xi'an Power Electronics Research Institute, Xi'an 710061, China)

Abstract: This paper introduces the importance of the automatic test firstly. Based on which, the paper proposes how to implement automation testing. Combining with the examples, the paper discusses emphatically some methods to achieve automated unit test & function test. In the end, this paper discusses the way for implementing automated performance test by Jmeter. These methods have gained good effect in J2EE-based enterprise application software testing.

Key words: automatic test; Cactus; test suit; Jmeter

1 引言

软件测试是为了发现错误而执行程序的过程。一个成功的系统与完整的、及时的测试密不可分。实现测试的自动化, 在每次代码修改后都重新运行一遍测试用例, 如果有错误, 系统自动给出出错原因, 使得系统发现缺陷改正缺陷非常便捷。

测试分为单元测试、功能测试、性能测试、集成测试和确认测试。其中单元测试属于白盒测试, 其它测试都属于黑盒测试^[1]。文中主要讨论如何自动进行单元测试、功能测试和性能测试。

2 单元测试

2.1 方法的选择

单元测试目的为验证一个或若干个类是否按所设计那样正常工作与表现, 简单的说单元测试就是比较预期结果是否与实际执行结果一致。单元测试

是一种白盒测试, 一般由开发人员完成。

基于 J2EE 平台的应用程序, 出现了如 Junit、Cunit 等开放源码的自动化测试框架, 使得单元测试可以自动化进行。程序代码先要部署到服务器容器 (以下简称容器) 中, 运行时代码在容器中由容器支持、控制和运行, 结果由服务器容器发给客户端。程序代码不能脱离容器单独运行, 运行结果不能从容器中直接得到, 必须由容器发给客户端才能得到。如果直接使用 Junit, 则只能在客户端进行测试, 这会使运行与测试位置不一致^[2]。

因此, 在 J2EE 应用程序中, 采用客户端测试不能使单元测试达到准确、彻底、尽可能多地发现错误。而 Cactus 是一种集成的、通用的、开放源码的计算科学和工程问题解决环境。用它进行 J2EE 应用程序单元测试可以避免以上所有问题。

2.2 使用 Cactus 实现自动化单元测试

Cactus 是一套简单, 易于使用的服务器端测试框

架,可以使开发人员很轻松的测试服务器端的程序. Cactus 是 Junit 的一个扩展,但是它又和 Junit 有一些不同. Cactus 的测试分为三种不同的测试类别, JspTestCase, ServletTestCase 和 FilterTestCase, 而不是像 Junit 就一种 TestCase. Cactus 的测试代码有服务器端和客户端两个部分,它们协同工作,其基本原理见图 1.

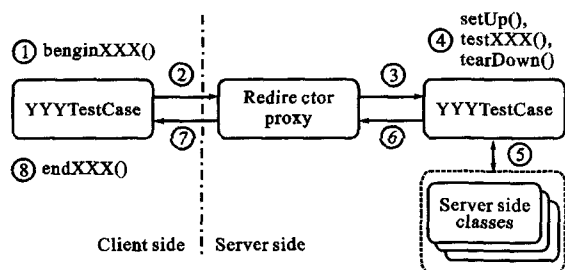


图 1 Cactus 测试原理图

Cactus 与 Junit 的测试代码的结构形式基本相同. 按照从下到上的顺序, 测试代码依次为 Test(测试)、TestCase(测试用例)、TestSuite(测试组), 他们之间为包含关系. 即一个 TestSuite 中可以包含多个 TestCase, 一个 TestCase 中可以包含多个 Test. 图 1 中 XXX 为测试名称, YYY 为测试用例名称, 它可以是 JspTestCase, ServletTestCase 和 FilterTestCase 三个类中任意一个的子类, 分别用于测试 Jsp, Servlet 和 Filter. TestCase 类必须在客户端和服务端都存放一份, 并且相同.

从图 1 可以看出, 使用 Cactus 测试框架时, 具体测试 XXX 的初始化和结束操作在客户端完成, 而测试用例的初始化和结束在服务器端完成. 服务器和客户端的 TestCase 完全相同, 它们通过 Cactus 提供的 Redirector Proxy(重定向代理)进行通讯, 保证了 Cactus 可以对 J2EE 容器中的服务器端类进行单元测试, 同时也保证了测试结果的正确性^[3].

3 功能测试

3.1 自动化功能测试

功能测试主要检测系统是否完成了需求所提出的功能. 如按资料作者查询资料的用例, 测试时应该测试查询出的资料作者是否是指定的作者. 功能测试在测试中占有非常重要的位置.

如何使这样的测试自动化, 而不需要每次测试时输入作者名, 再提交, 然后看看作者栏是否都是输入的作者. 输入的作者名是存放在请求对象中的, 可以通过在 request 对象中设置参数来模拟输入作者名. 服务器给客户端应答中的作者栏, 一般都是

table 中的一列, 可以通过判断这列的内容是否与输入的作者名相同, 完成人工查看的工作.

HttpUnit 给开发者提供了实现基于 WEB 应用系统功能测试自动化所必须的这些条件. HttpUnit 提供了 WebConversation、WebResponse、WebRequest 三个类, 从 WebConversation 可以得到 WebResponse、WebRequest 对象分别用于代表 HTTP 协议的请求和应答. 从 WebResponse 对象中可以得到指定位置的内容、链接等^[4].

下面以查询指定作者的资料这一功能的测试用例说明.

```
package com.yymis.dmi;
import junit.framework.TestCase;
import com.meterware.httpunit.*;
public class HTTPTest extends TestCase
{ private WebResponse response;
  private WebConversation conversation;
  private WebRequest request;
  public HTTPTest(String arg0)
  { super(arg0);
  }
  protected void setUp() throws Exception
  { conversation = new WebConversation();
    request = new GetMethodWebRequest
    (http://localhost:8080/listDoc.jsp&author='u-
    nauthor');
    response = conversation.getResponse(request);
  }
  public void testCell() throws Exception
  { WebTable salesReport = response.getTables()[0];
    TableCell cell = salesReport.getTableCell(2,3);
    assertTrue("作者", cell.asText().startsWith("u-
    nauthor"));
  }
}
```

在上面的代码中, 测试了系统是否完成了查询指定作者资料的功能. 首先用 GetMethodWebRequest() 向 listDoc.jsp 发送请求, 并对 author 参数赋值, 然后通过 WebResponse 类对象 conversation 得到应答, 最后测试应答中表的指定单元是否与请求的参数相同.

3.2 TestSuite 编写及测试运行

单元测试用例与功能测试用例编写完毕后, 就需要将所有的 TestCase 组成 TestSuite 并运行, 因为单元测试和功能测试框架都是在 Junit 基础上建立的, 因此所有测试用例可以组成一个 TestSuite, 并一

次运行.代码如下:

```
package com. yymis. dm;
import junit. framework. Test;
import junit. framework. TestSuite;
public class AllTests
{
    public static void main(String[] args) {
        junit. textui. TestRunner. run(suite());
    }
    public static Test suite() {
        TestSuite suite = new TestSuite("Test for com.
        xymis. dm package");
        suite. addTest(new TestSuite(Javabeantest.
        class));
        suite. addTest(new TestSuite(ServerletTest.
        class));
        suite. addTest(new TestSuite(FilterTest. class));
        suite. addTest(new TestSuite(HTTPTest. class));
        return suite;
    }
}
```

这样每次只需运行 AllTests 类,即可实现对全部测试用例的执行,实现了功能测试自动化。

4 性能测试

性能测试主要检测系统是否完成了需求所提出的强度、速度、精度等性能方面的要求.性能测试工具我们选用了开放源代码 apache 组织的 Jmeter.

Jmeter 测试 HTTP 页面、Servlets、JDBC、JNDI、和 FTP 服务器性能时,不需要任何代码.只需要在 Jmeter 中进行简单的设置,如设置目标地址,并发访问的线程数量^[4]等.详细的设置可以参看 Jmeter 的在线帮助文档.对于 Java Objects 的测试需要按一定的要求编写代码.下面说明 Java Objects 性能测试的方法.

测试类必须声明 org. apache. jmeter. protocol. java. sampler. JavaSamplerClient 和 java. io. Serializable 接口. JavaSamplerClient 接口有三个接口方法 setupTest()、teardownTest()、runTest(),分别用于被测测试框架在测试出始化、测试中止化和测试进行时调用. runTest()中必须返回 SampleResult 类的实例(result),而且必须对返回的 result 调用 setTime()方法^[5].下面以测试的对 StringBuffer 类调用其 1000 次“加”来说明,代码如下:

```
package com. yymis. dm;
import java. io. Serializable;
```

```
import java. util. HashMap;
import org. apache. jmeter. protocol. java. sampler.
JavaSamplerClient;
import org. apache. jmeter. samplers. SampleResult;
public class LoadTest implements JavaSamplerClient, Seri-
alizable
{public void setupTest(HashMap arg0)
{
}
public void teardownTest(HashMap arg0)
{
}
public SampleResult runTest(HashMap arg0)
{ long start = System. currentTimeMillis();
  SampleResult result= new SampleResult();
  StringBuffer a= new StringBuffer();
  for (int i=0; i<1000; i++ )
  { a. append(i);
  }
  System. out. println(a);
  long end= System. currentTimeMillis();
  result. setTime(end - start);
  result. setSuccessful(true);
  result. setSampleLabel("loadtest" + (start - end));
  return result;
}
}
```

在 runTest 中用 System. currentTimeMillis()得到现在的时间,分别记录测试前和测试后的时间,并给返回的 result 设置运行时间为 end - start.

将上面的代码用 Javac 编译,放到 Jmeter 的测试目录下,在 Jmeter 中选择 Java Obejct 测试,就可以看到 Jmeter 在测试目标中多了一项 com. xymis. dm. LoadTest.然后设置测试次数为 100 次,一个并发线程.得到性能测试结果为平均 0.09ms,如果将 StringBuffer 换成 String 得到的性能测试结果为平均 0.21ms,性能相差了 20 多倍.可以看出 Jmeter 为系统在性能优化时提供了可靠的依据.

5 结束语

文中用实例介绍了系统的测试过程,在单元测试和功能测试时引入 Junit、Cactus、HttpUnit 这三个测试框架后,实现了单元测试和功能测试的自动化.最后介绍了用 Jmeter 进行系统的性能测试方法,为编写高质量的应用软件奠定了基础.

(下转第 171 页)

model;

②4 show clustering result.

在本算法中,密度聚类的效果由 δ, ξ 加以控制,用密度聚类对样本进行聚类并计算出 EM 算法的初始值后再用高斯混合模型对整幅图像进行聚类分析,聚类结果较稳定,并具有良好的语义效果.

4 实验结果和性能评估

为了比较 Kmeans 法和近似密度法对 EM 算法初始化的效果,分别将两种方法初始化后的参数应用于人体腹部医学图像的高斯混合模型聚类.文中重点对人体肿瘤发病率较高的肝脏和肾脏两个腹部器官进行聚类分析首先对图像进行预处理并抽取 2500 个图像样本.表 1 为使用不同方法进行 EM 初始化后用于聚类的两个参数比较.

表 1 使用不同方法的比较

初始化方法	脏器	正确率	误判率
Kmeans	肝脏	82.46	2.93
	肾脏	75.76	0.74
近似密度	肝脏	90.66	1.98
	肾脏	92.56	0.45

用两种初始化 EM 的方法对一幅人湾腹部医学图像进行混合模型聚类,并得到了一组正确率和误判率的参数比较.从实验结果可以看出,用近似密度法初始化 EM 得到的正确率要大于 Kmeans 法,错误率小于 Kmeans 方法,达到了更好的聚类效果.

5 结束语

文中针对人体腹部医学 CT 图像,提出了用近似密度估计法来初始化 EM,并将初始化结果应用于高斯混合模型聚类分析,由于近似密度估计能较好地表达医学图像的内容,因而得到了较好的聚类结果.采用近似密度方法初始化 EM,提高了高斯混合模型聚类分析的准确性,有深刻的临床意义.

参考文献:

[1] 罗述谦,周果宏.医学图像处理与分析[M].北京:科学出版社,2003.

[2] Fraley C, Raftery A E. How many clusters? Which clustering method? - Answers via model - based cluster analysis[J]. The Computer Journal,1998(41):578 - 588.

[3] Jeff C F WU. On the convergence properties of the EM algorithm[J]. The Annals of Statistics, 1983,11(1):95 - 103.

[4] Christophe Biernacki. Initializing EM using the properites of its trajectories in gaussian mixtures[J]. Statistics and Computing, 2005,14(3):267 - 269.

[5] Biernacki C, Celeux G, Govaert G. Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate gaussian mixture models[J]. Computational Statistics and Data analysis, 2002,24(5):603 - 619.

[6] Hinneburg A, Keim D. An efficient approach to clustering in large multimedia databases with noise[C]// Proc of the 1998 4th Int,l Conf on Knowledge Discovery and Data mining. New York: AAAI Press,1998:58 - 65.

[7]李艳玲,王加俊.基于高斯混合模型的纹理图像的分割[J].微电子学与计算机,2004,21(4):16 - 19.

[8] 苏锦旗,薛惠锋,詹海亮.基于划分的 K - 均值初始聚类中心优化算法[J].微电子学与计算机,2009,26(1):8 - 11.

[9] 宋余庆.医学图像数据挖掘若干技术研究[D].镇江:江苏大学,2007.

[10] 谢从华,朱峰,王立军,等.基于密度聚类的医学图像分割[J]. 计算机应用研究,2007(2):167 - 169.

作者简介:

张树伟 男,(1982 -),硕士研究生.研究方向为医学图像处理.

宋余庆 男,(1959 -),博士,教授.研究方向为数据挖掘、知识发现、图像数据库系统、模式识别等.

陈健美 女,(1962 -),博士,教授.研究方向为数据挖掘、医学图像数据库.

谢从华 男,(1978 -),博士研究生,讲师.研究方向为医学图像处理、数据挖掘、模式识别.

(上接第 167 页)

参考文献:

[1] 郑人杰.软件工程:高级[M].北京:清华大学出版社,2002.

[2] 称华军.J2EE 构件企业级应用解决方案 [M].北京:清华大学出版社,2008.

[3] 贾小云,陈桦,奚宗悦.企业应用程序中会话 Bean 的设计[J].微电子学与计算机,2004,21(8):52 - 55.

[4] 胡滨.自动化测试工具的研究[J].现代电子技术,2007(8):97 - 99.

[5] Roman ED. Mastering EJB[M]. New York, Wiley Longmen, 2002.

作者简介:

贾小云 女,(1971 -),教师,副教授.研究方向为软件工程.