Watir 自规范测试框架在一体化项目测试中的应用

徐 嫣,卢敬德
(珠江水利科学研究院,广东 广州 510611)

摘 要:广东省水利业务一体化管理系统是广东省水利数据中心工程(数据中心工程)的重要组成部分,项目具有集成度高、大型、复杂等特点。基于传统的手工测试方法在一体化系统测试工作中严重制约了测试乃至项目研发的效率,通过实践文中提出了一种基于 Watir 的自规范测试方法,在有效降低一体化系统测试成本、缩短项目开发周期、提高项目研发效率的基础上消除了自动化测试带来的 bug 重现率降低的问题。该方法具有高效、可靠、稳定、便于实现等优点。

关键词:自动化测试;Watir;bug 重现率低;自规范测试框架
中图分类号:TP39 **文献标识码**:A **文章编号**:1673-629X(2014)03-0238-04
doi:10.3969/j.issn.1673-629X.2014.03.059

Application of Watir Automation Test Framework in Integration Project Test

XU Yan,LU Jing-de
(Guangzhou Pearl River Water Conservancy Science Research Institute,Guangzhou 510611,China)

Abstract:Guangdong water business integrated management system is an important part of data center of Guangdong province water conservancy engineering (data center project) with high integration,large-scale,complexity and other characteristics. On the basis of the traditional manual testing method in the integrated system testing work seriously restricted the test and the efficiency of the project research and development,by practicing a specification of Watir-based self-testing method is presented. The problem that low bug reproduction rate caused by test automation can be eliminated on the basis of reducing costs of integrated system testing effectively,shortening project development cycle,improving the efficiency of research and development projects. The method is efficient,reliable,stable,easy to realize.

Key words:automation test;Watir;low bugs reproduction rate;test automation framework

0 引 言

广东省水利业务一体化管理系统(以下简称一体化系统)是广东省水利数据中心工程(数据中心工程)的重要组成部分。一体化系统集水利信息资源存储管理、共享交换、应用服务等功能于一体,采用 Oracle 数据库存储海量数据,在 B/S 架构上,通过 J2EE 框架进行开发。系统建设工作包括农村水利、水库移民、建设管理、农村机电、水政监察、行政许可、科技管理、人才管理、规划计划、发展评价等十多个业务应用子系统的开发与集成。

一体化系统的庞杂性带来了传统手工测试难以解决的效率问题,尤其是不断重复的回归测试严重制约了测试乃至项目研发工作的效率,因此必须使用有效、可靠、自动化的测试方法对系统进行测试。

1 Web 自动化测试技术

1.1 自动化测试简介

自动化测试是使用软件工具代替手工测试进行的一系列系统操作和相关记录[1]。自动化测试工具通常是一些脚本程序或者其他代码驱动的应用程序。自动化测试是通过用户界面操作或命令集操作完成的[2]。

Web 自动化测试工具一般由表单提交(Form Submit)、查询结果可视化(Results Visualization)、缺陷跟踪(Bug Tracking)等功能组成[3]。Web 自动化测试工具是基于脚本的测试框架开发平台,通过灵活集成多源测试组件,实现各种基于不同平台开发系统的测试,在此基础上,提供测试脚本管理、测试结果管理、测试实例管理等功能[4]。

收稿日期:2013-05-08 **修回日期**:2013-08-25 **网络出版时间**:2014-01-07
基金项目:广东省水利数据中心工程项目(粤发改农[2009]385 号,粤发改农[2010]95 号,粤发改农经函[2010]2122 号)
作者简介:徐 嫣(1981-),女,湖北武汉人,硕士,研究方向为水利信息化。
网络出版地址:http://www.cnki.net/kcms/detail/61.1450.TP.20140107.1528.025.html

1.2　Web 自动化测试工具

目前基于 Web 应用的自动化测试工具很多,其中比较具有代表性的是 Watir 和 Selenium。表 1 对 Watir 和 Selenium 的适用范围、支持语言和安装条件进行了对比[5-6]。

表 1　Selenium-Watir 对比

	Selenium	Watir
支持浏览器	IE, Firefox	IE, Firefox
支持平台	Windows, Linux	Windows, Linux
易用性	简单易用	需要开发
支持语言	Selenese, Ruby, Java	Ruby
支持录制功能	Firefox	不支持
需要安装	是	否

Selenium 是一个用于 Web 应用程序测试的工具。Selenium 测试直接运行在浏览器中并支持自动录制动作和自动生成。Selenium 基于录制的测试方法在一些小型系统的快速自动化测试应用中十分便捷却无法在大型系统的测试中准确完成高复杂度的用例测试;Watir 是一个 Ruby 语言编写的开源类库[7],提供了轻量级的自动化测试程序框架和丰富的开源库,通过 Watir 可以十分灵活地构建任何基于 Web 应用的自动化测试用例,因此在一体化系统自动化测试框架的构建上测试小组使用了 Watir。

1.3　Watir 特点

Watir 是通过 DOM(Document Object Model)技术来获取 Web 页面元素的。其工作原理如图 1 所示[8]。

图 1　Watir 工作原理图

Watir 的主要特性及优势如下:

1)Watir 是一款基于 Ruby 语言的测试框架,具有面向对象、功能强大、简单易用等特性。程序无须编译可直接解释运行,具有良好的跨平台性。

2)Watir 提供了丰富的开发库,封装了包括 iframe,frame,table,div,input,menu,td,button,img,map,li,option,select 等绝大多数 HTML 对象,测试工程师可以方便调用开发库内封装的对象快速构建自动化测试程序。

3)Ruby 支持 IRB(Interactive Ruby Shell)交互式命令方式进行调试,交互式命令调试支持以完整程序、代码段、代码句等对象为单元的多尺度交互调试。

4)Ruby 支持对象嵌套功能,一个测试用例可以单独测试也可以嵌套在别的测试用例中测试,在测试用例之上允许建立测试用例集,通过用例嵌套和测试用例集功能,用户可以灵活组织测试结构。

5)提供断言机制(Assertion)辅助完成复杂测试逻辑。

6)Watir 能够很好地兼容其他脚本语言如 Perl,Python,Shell 等。

7)Watir 不要求独占网页资源,用户可以进行并发浏览测试[9]。

2　基于 Watir 的自动化测试框架应用

2.1　搭建 Watir 开发运行环境

下载并安装 Ruby 包和 Watir 包,安装成功之后在浏览器上安装 html 查看插件。在 Eclipse 中安装 Ruby 插件[10]。

Watir 是一个 Ruby 开源库,阅读其源代码不仅可以了解 Watir API 的具体实现,同时它本身也是编写 Ruby 和 Watir 的很好的代码示例。Watir 源码位于{Ruby 安装目录}\lib\ruby\gems\1.8\gems\watir-version,其中 unittests 目录下是对 Watir 所支持的页面对象类型以及主要的 API 进行单元测试的 Ruby 程序,一体化系统测试小组在编写 Watir 自动化程序时需要先查看和了解 Watir 源代码[11]。

2.2　创建 Watir 自动化测试程序

Watir 开发环境搭建好后,在 Eclipse 中创建一个 Watir 自动化测试程序项目如下:

1)在 Eclipse 中创建一个 Ruby 项目。

2)搭建自动化测试程序框架,一体化系统测试小组延续了 IBM 的三层自动化体系结构,从下到上分别是公共对象(Object),任务(Task),测试用例和测试用例集(Test Case)。

3)编写 Watir 程序。使用 IE Explorer Developer Toolbar 去抓取 Web 页面对象属性。

4)通过继承 Test::Unit 框架编写测试用例和测试用例集。

5)通过断言(Assertion)为测试用例添加测试结果验证,并通过日志及截屏(类 Watir::ScreenCapture 提供了对截屏的支持)记录测试结果。

6)在 YAML 或 Excel 中准备测试数据。

7)启动 IRB(Interact Ruby Shell)来调试 Watir 代码片段。

8)运行 Ruby 程序。

9)查看运行日志,分析测试结果。

3　Watir 自动化测试导致 bug 重现率降低

测试小组在一体化系统自动测试的过程中发现,使用基于原生 Watir 框架生成的自动测试程序降低了系统中 bug 的重现率(见表 2)。在系统测试中,bug 的重现率越低,测试的置信度也就越低、软件中的缺陷越容易被忽略并最终影响交付系统的质量[12]。通过数据分析,测试小组发现造成一体化系统自动测试 bug

重现率低的原因是不同测次数据环境的各异性。

表 2 手工测试-自动测试 bug 重现率比对表

子系统	测试方式	bug 重现率/%
数据共享与交换	手工测试	98
	自动化测试	67
综合信息展示	手工测试	99
	自动化测试	99
数据发掘与分析	手工测试	89
	自动化测试	93
数据发布	手工测试	97
	自动化测试	62
专题服务	手工测试	92
	自动化测试	58
业务一体化管理	手工测试	86
	自动化测试	61
信息交流平台	手工测试	98
	自动化测试	72
系统管理	手工测试	99
	自动化测试	90

下面介绍一个一体化系统自动测试过程中经过碰到的 bug 不可重现案例:

1)测试员 A 使用测试程序 p1 对水文站 ST1 的历史洪水过程相似度进行了测试,得出了测试数据{a},从这次测试中,p1 发现{a}与测试用例中指定的预期输出数据不一致,于是 p1 生成相应错误报告 e1,并将 e1 发送至开发人员。

2)在开发人员对 e1 进行处理之前,测试员 B 使用测试程序 p2 对水文站 ST1 的历史数据进行了修改测试,测试成功,没有发送错误。

3)开发人员在查看 e1 后希望重现 e1 中描述的 bug,并重复相同操作以重现错误,由于在此之前 p2 修改了水文站 ST1 的历史数据,因此开发人员不能在这里重现出 e1 中的错误,e1 最终被退回。

通过上面的案例可以看出,由于自动测试过程中数据环境发生了改变,从而造成了 bug 不可重现的问题。

4 自动测试用例的自规范化

4.1 测试用例格式规范

只要保证不同测次的测试数据环境一致,bug 重现率低的问题就可以避免[11]。通过一体化系统自动测试的实践,文中提出了一个简单实用的自动测试改进办法,在这个方法中,首先要对用例的规范化进行定义:

1)一个或多个测试用例块可包含在同一个测试用例文件中。

2)每个测试用例块应包含必要的注析。注析内容应酌情包含用例名称、描述、创建者、创建时间、备注等信息,提高测试用例的易解性。

3)测试用例块具有严格的格式,保证测试用例块可单独进行测试。测试用例块格式有:

· 开始标识符:<TestBlock>。

表示测试用例块开始,在测试用例中必须与</TestBlock>成对出现。

· 前置条件:<precondition>。

指定测试用例运行时需要设置的运行环境条件,主要是测试数据库的数据环境。设置前置条件的目的是保证测试的独立性和可重现性。前置条件由一个或多个影响数据库或表状态的属性值构成。

· 预期结果。

指明测试运行后的预期结果。预期结果允许是简单的属性值;对于复杂的测试块而言需要定义预期结果结构体。程序对比测试结果和预期结果,当测试结果不在预期结果之中时对测试结果进行输出。

· 测试内容。

测试内容则是测试用例的主体。包含所有用于测试的语句。

· 后置处理。

后置处理相当于测试工作的后勤部,负责对测试后的现场进行还原以便后续测试获得与之前测试相同的数据环境:对于只含数据库读取操作的测试用例块无需后置处理。对于包含数据库读写操作的测试用例块,必须在此定义测试结束后的后置处理动作,对系统中的脏数据进行有效清理。

· 结束标识符:</TestBlock>。

表示测试用例块结束,在测试用例中与<TestBlock>成对出现。

4)前置条件应该包含以下几个方面:

· 清除无效数据,保证测试数据的有效性;

· 创建测试用户账号;

· 按照测试角色分配给测试用户账号相应的权限;

· 建立数据库连接;

· 创建测试相关数据表;

· 输入测试数据。

5)后置处理应该包含以下几个方面:

· 删除测试中插入表的数据;

· 删除临时表;

· 删除测试用户;

· 断开连接。

按照以上定义的规范编写的测试用例,提高了测试用例的可阅读性、可重用性,降低了 bug 的不可重现率。但由于一体化系统项目中已有大量不规范的测试

用例,于是文中提出了用例行为分析—自规范算法,用于自动规范已有的测试用例。

4.2　自动化测试用例

由于在一体化系统项目中已经存在大量格式不规范的测试用例,所以文中提出一种算法——用例行为分析—自规范算法。使用该算法实现测试用例的规范化。

如前所述,测试过程中数据的前后一致性对于测试结果的准确性、合理性有着重要意义。为了得到文中要求的数据一致性,需要程序自动跟踪分析测试用例测试过程中的行为。

用例行为分析—自规范算法分步进行:第一步是对程序语句语意及语句中包含的 SQL 进行分析;第二步是根据分析结果,对改变了测试前后数据库数据环境一致性的语句进行后处理,使经过后处理后的数据库数据环境与测试前保持一致。SQL 语句可分为查询语句和更新语句,具体划分如下:

1)查询语句:用于查询数据表或视图的语句、用于查询数据库系统设置的语句、用于查询统计值的语句。

2)更新语句:

· 用于创建用户的语句;

· 用于分配用户权限的语句;

· 用于创建数据库连接的语句;

· 用于创建数据库、表空间、表、视图、完整性约束、索引的语句;

· 用于修改数据库、表空间、表、视图、完整性约束、索引的语句;

· 用于删除数据库、表空间、表、视图、完整性约束、索引的语句;

· 用于修改表数据的插入语句;

· 用于修改表数据的修改语句;

· 用于修改表数据的删除语句。

在分析测试用例过程中,如果该语句属于查询语句,则跳过并继续检查;如果属于更新语句,则向行为分析文件中加入对应记录。记录内容包括:序号,行为名称,操作对象,完整的行为语句。

当读取到<TestBlock>时,根据行为分析记录文件重现完整测试用例在此阶段的数据库状态并将其写入到<precondition>的相应属性中,以保持数据库环境的一致。当遇到</TestBlock>时,应根据过程行为记录,执行相应的后置处理,以清除脏数据,保持数据库状态前后一致。自此,一个自规范格式的测试用例构建完毕。

4.3　Watir 测试用例自规范化实例

根据前文提出的算法,实现 Watir 测试用例自规范化测试框架——Watir Test Framework。

一体化系统中共有测试用例 14 307 个,其中大多数测试用例格式较不规范。如:多个测试块共用前置条件、缺少后置处理部分等。为了解决这种用例不规范带来的问题,文中根据前面提出的格式规范,并结合 Watir 测试用例的特点,制定 Watir 测试用例规范。

根据上面提出的 Watir 测试用例格式规范和用例行为分析—自规范算法,使用 Java 编写了基于 Watir 的自规范测试框架 TestCaseFormation 工具,通过该工具完成了一体化系统测试用例格式自动规范化的工作,并使用规范化后的测试用例进行系统的自动测试,实践表明规范化后的测试用例很好地保障了 bug 的重现率,同时也提高了测试的整体效率。

5　结束语

一体化系统在开发阶段由于存在需求的变更、bug 的修复、代码优化以及平台移植等问题,需要不断对模块代码进行修改,根据国家及行业信息化对软件测试的规范要求,模块代码一经更改,无论之前测试与否,均需对修改的模块进行回归测试以确认修改没有引入新的错误或导致其他代码产生错误。由于一体化系统庞大、复杂度高,大量手工重复的回归测试严重制约了项目研发效率。基于回归测试的重复性、流程固定性以及测试项目重复率高等特点,测试小组通过基于 Watir 建立一体化系统自动化测试程序,并进行系统自动化测试,有效降低了一体化系统的测试成本、缩短了项目的开发周期,提高了项目研发的效率。在使用基于 Watir 建立的自动化测试程序系统测试过程中,遇到了 bug 重现率降低的问题,测试小组通过用例行为分析—自规范算法修正了数据环境不一致的问题,使 bug 重现率达到理想状态,达到了提高测试效率的同时保障 bug 重现率的目的。

参考文献:

[1]　黄梦薇,黄大庆,周　未. 基于 Watir 的 WEB 自动化回归测试框架[J]. 电子设计工程,2012,20(21):34-36.

[2]　Ash L. Web 测试指南[M]. 北京:机械工业出版社,2004.

[3]　朱少民. 全程软件测试[M]. 北京:电子工业出版社,2007.

[4]　杜吉伟. 自动化测试的未来[J]. 电子技术,2005(10):56-58.

[5]　陆　璐,王柏勇. 软件自动化测试技术[M]. 北京:清华大学出版社;北京交通大学出版社,2006:10-11.

[6]　路小丽,葛　玮,龚晓庆. 软件测试技术[M]. 北京:机械工业出版社,2007:23-25.

[7]　Huberty D,马　博,赵云龙. 软件质量和软件测试[M]. 北京:清华大学出版社,2003:86-88.

(下转第 245 页)

消息到达 AAA(为了便于描述假设消息是基于 Radius 协议的码流);

表 1　Radius 向 Diameter 的消息转换配置

Radius AVP	Diameter AVP	说明
User-Name	[User-Name]	NAI
无	[Origin-State-Id]	AAA 主动生成
Event-Timestamp	[Event-Timestamp]	Event-Timestamp
Calling-Station-ID	*[Subscription-Id] {Subscription-Id-Type} {Subscription-Id-Data}	Calling-Station-ID = MDN
Release-Indicator	[Termination-Cause]	AAA 转换
PPAQ	*[Used-Service-Unit]	
QuotaId	无	Diameter 中无对应参数,不需要转换
Update-Reason	[Reporting-Reason]	
无	[Tariff-Change-Usage]	Diameter 中无对应参数,不需要转换
DurationQuota	[CC-Time]	
VolumeQuota	[CC-Total-Octets]	

步骤 3. AAA 协议适配与转换模块根据静态的配置数据,选择适用的协议栈进行消息码流处理;或者如果没有静态数据配置策略,协议适配与转换模块根据消息码流的格式,自动选择 Radius 或者 Diameter 协议进行处理;

步骤 4. Radius 协议栈处理模块对码流进行编解码等处理,处理后的码流送达业务处理模块;

步骤 5. 业务处理模块按照一定的业务逻辑,完成对消息的业务层处理,并判断消息下一步到达的系统设备所支持的协议类型是否也是基于 Radius 协议的?

如果支持,则不需要进行协议转换与翻译,业务处理后的消息经协议栈模块重新编解码后,通过协议适配与转发模块发送回去,流程结束;

如果下一步到达的系统设备所支持的协议类型为 Diameter,则流程转步骤 6,消息码流送达协议适配与转换模块;

步骤 6. 协议适配与转换模块对消息按照配置模块的转换规则进行转换,完成后,消息经协议栈模块重新编解码后,流程转步骤 7;

步骤 7. 转换为基于 Diameter 协议的消息码流送达协议适配与转发模块,流程转步骤 8;

步骤 8. 协议适配与转发模块经底层通讯以及监听模块发出转换后的 Diameter 消息,流程结束。

4　结束语

文中设计和实现的基于双栈方式的下一代 AAA 服务器,同时支持 Radius/Diameter 协议的处理,且可配置地支持不同协议系统设备间的消息转换,具备很高的应用性和灵活性,在实际应用部署中,能够带来显著的经济价值,是下一代网络中 AAA 服务器应用的一个发展方向。但由于篇幅所限,文中并没有探讨下一代 AAA 服务器的容灾方式,安全特性,高性能架构以及基于云服务模式的业务处理等内容,这些可以作为进一步研究和应用的方向。

参考文献:

[1] Rigney C, Rubens A C, Simpson W A, et al. Remote Authentication Dial In User Service (Radius)[S]. RFC 2865, 2000.

[2] 朱海龙,张国清. 基于 DIAMETER 的 AAA 技术及其在 Mobile IP 中的应用[J]. 计算机工程与应用,2003(21):159-163.

[3] Fajardo V, Arkko J, Loughney J, et al. Diameter base protocol [S]. RFC 6733, 2012.

[4] 李斌祥. 采用 Radius 协议的 AAA 系统的研究[J]. 重庆邮电学院学报(自然科学版),2006(Sup):193-196.

[5] 蔡　磊,陈　越,王　娜,等. DIAMETER 协议和 Radius 协议的比较[J]. 微计算机信息,2006,22(5-3):244-246.

[6] 李　倩. AAA 认证协议的分析[J]. 北京工商大学学报(自然科学版),2006,24(4):45-47.

[7] 张　琪,喻占武,李　锐,等. 基于 AAA 服务的协议分析与比较[J]. 计算机应用研究,2007(2):296-298.

[8] 黄永锋,王　滨,许晓东. RADIUS 在 802.1x 中的应用[J]. 计算机工程与设计,2006,27(5):798-801.

[9] 3GPP TS23.402. Architecture enhancements for non-3GPP accesses Version10.0.0[S]. 2010.

[10] 3GPP TS29.273. Evolved Packet System (EPS);3GPP EPS AAA interfaces[S]. 2010.

[11] 3GPP TS23.234. 3GPP system to Wireless Local Area Network Version10.0.0 (WLAN) interworking;System description[S]. 2010.

[12] 陈能干,裘姝平. 基于 Diameter 的 AAA 服务器的设计与实现[J]. 计算机工程与设计,2004,25(12):2274-2276.

(上接第 241 页)

[8] 王明兰,叶东升. 测试用例描述语言研究[J]. 计算机工程与设计,2006,27(22):4281-4284.

[9] Aurum A, Peterson H, William C. State-of-the art: Software inspectiors after 25 years[J]. Software testing, verification and reliability, 2002, 12(3):133-154.

[10] Myers G J. The art of software testing[M]. New Jersey: John Wiley & Sors, Inc., 2004.

[11] Patton R. Software testing[M]. 2nd Revised ed. [s. l.]: Sams Publishing, 2005.

[12] Dustin E, Garrett T, Gauf B. Implementing automated software testing—How to save time and lower while raising quality [M]. Beijing: Publishing House of Electronic Industry, 2011: 98-100.

Watir自规范测试框架在一体化项目测试中的应用

作者： 徐嫣， 卢敬德， XU Yan， LU Jing-de
作者单位： 珠江水利科学研究院,广东 广州,510611
刊名： 计算机技术与发展
ISTIC
英文刊名： Computer Technology and Development
年，卷(期)： 2014(3)