doi:10.3969/j.issn.1672 - 4348.2015.03.008

浏览器兼容性自动化测试方法研究

高培[1], 何栋炜[2]

(1. 福建对外经济贸易职业技术学院 信息技术系, 福建 福州 350016;
2. 福建工程学院 信息科学与工程学院, 福建 福州 350118)

摘要: 浏览器是 Web 应用程序的客户端软件,针对浏览器兼容性测试的手工效率低,工作量大的问题, 提出了一种并行的自动化测试方法,并将图像匹配技术结合到该自动化测试系统中,解决了脚本编写过程中界面检查点无法设置的问题。对该方法中的自动化测试框架组成结构和测试脚本编写进行了研究,并对如何生成测试结果及输出的测试报告内容进行了说明。为验证方法的有效性针对三款不同的浏览器在 Web 程序上进行了自动和手动测试时间对比,实验结果表明使用所提出的方法可以使测试速度提升大约 3 倍。

关键词: 浏览器; 兼容性; 自动测试
中图分类号: TP311.52; TP311.56 **文献标志码**: A **文章编号**: 1672 - 4348(2015)03 - 0244 - 06

Study on automatic browser compatibility testing method

Gao Pei[1], He Dongwei[2]

(1. Information Technology Department, Fujian International Business and Economic College, Fuzhou 350016, China;
2. College of Information Science and Engineering, Fujian University of Technology, Fuzhou 350118, China)

Abstract: Browser is a client software of Web application. To tackle the low efficiency and huge workload in manual cross-browser testing, a parallel automatic test method was proposed in which image matching technology was incorporated to solve the checkpoint setting problem within scripting. The components of automatic testing framework and the writing of test scripts were studied. In addition, the generation of the test results and the content of the test report were described. Finally a validation experiment of three different browsers was conducted to compare the time durations of automatic test and manual test. The results demonstrate that the proposed method can improve the testing speed by about three times.

Keywords: browser; compatibility; automatic test

当前的基于 Web 开发的系统,大多数都是基于 B/S 架构的。浏览器是 Web 服务的客户端软件,即使用者访问的服务器的内容在客户端是通过浏览器显示出来的。网页浏览器对于单独的一台电脑而言,也相当于操作系统之上的一个应用软件,它的作用就是对于网页中的文件进行显示,这些文件的内容一般是由 HTML 代码来组成的。在个人电脑上常用的网页浏览器,现在是越来越多样了,常用的浏览器有 IE、Firefox、Google Chrome、360 安全浏览器、搜狗浏览器等[1]。但是人们在使用一些浏览器访问 Web 服务器时,发现一些浏览器对 Web 页面并不能很好地兼容,比如可能出现布局不合理、内容不完全显示、一些功能无法使用等[2]。要解决这些问题,首先需要对 Web 应用程序的浏览器兼容性进行测试,对这些浏览器不能正常使用的功能进行分析,以便有针对性

收稿日期: 2015 - 04 - 11
第一作者简介: 高培(1984 -),女,湖北黄冈人,助教,硕士,研究方向:应用软件测试技术。

地改进[3]。使用自动化测试工具进行浏览器兼容性测试能够减少测试过程中的重复劳动,实现测试自动化,提高测试质量。本文在对浏览器兼容性测试技术分析的基础上,提出了一种新的浏览器兼容性自动化测试方法,并具体说明了该自动化测试方法的功能架构及工作流程。

1　浏览器兼容性测试技术分析

浏览器是指可以显示网页服务器或者文件系统的 HTML 文件内容,并让用户与这些文件交互的一种软件。浏览器的核心部分是"rendering engine",可大概译为"渲染引擎",一般习惯称为"浏览器内核",负责对网页语法的解释(如 HTML、CSS、JavaScript 等)并渲染(显示)网页。不同的浏览器内核对网页编写语法的解释不同,因此同一网页在不同内核的浏览器里的渲染(显示)效果也可能不同。简单来说,一些浏览器对网页产生不兼容的主要原因包括下列几项:1)对网页设计语言解释不一致。不同浏览器对网页语言(如 HTML、CSS、JavaScript 等)解释不一致,导致呈现出来的界面显示不一致,比如有些样式在 IE 下都能显示正常,但在其他的浏览器可能无法显示或显示不全。2)开发人员没有考虑到浏览器之间的差异性。尽管浏览器的客户端技术不断发展,但每个浏览器都有自己特殊的属性,因此开发人员要写出支持不同浏览器的代码,这需要开发人员有足够的经验能够判断哪些地方会出现问题,并采取针对性的措施去解决。3)页面语法不正确。大部分浏览器对 HTML 标签丢失的处理比较温和,但有些浏览器对标签丢失的处理方式会有所不同,毕竟业界也没有一定的标准要求如何处理。比如一个 table 标签的丢失在旧的 Netscape Navigator 上显示为空白而在 IE 浏览器上显示却是正确的。

对于浏览器兼容性问题及产生的原因有许多研究[4],并且网络上也存在一些浏览器兼容性测试工具,比如 IETester、SuperPreview 等,但这些工具只是针对静态网页或电子商务网站之类的,对于需要交互的动态的 Web 网页,则是束手无策。对于使用不同浏览器的 Web 应用程序的兼容性测试当前主要还是以手动测试为主[4-5],手动测试通过测试人员执行测试用例,再将测试结果和预期结果进行对比并且记录。随着需要支持的浏览器产品类型越来越多,测试部门的工作量越来越大,且人工测试的效率比较低,很多时候都无法满足测试的需求,但自动化测试能够有效地解决这些问题。

自动化测试是借助于测试工具、测试规范,模拟人工操作,从而局部或全部代替人工进行测试及提高测试效率的过程。自动化测试有很强的优势,它借助计算机的计算能力,可以重复地、不知疲倦地运行,对于数据,能进行精确的、大批量的比较,而且不会出错[6]。以下对比传统的手工测试和自动化测试在进行浏览器兼容性测试时的测试工作量。

1.1　传统的手工测试

测试内容:测试一款软件在 IE 6.0、IE 7.0、IE 8.0 上的兼容性,需要执行用例 N 条。

测试步骤:

(1)测试执行人员在 IE 6.0 上手工执行该软件需要执行的所有 N 条用例;

(2)根据 N 条用例对应的预期结果检查软件功能是否正确,界面布局、控件大小、背景颜色是否显示正常;

(3)在 IE 7.0 和 IE 8.0 上分别重复上述步骤 1、2。

1.2　自动化测试

测试内容:测试一款软件在 IE 6.0、IE 7.0、IE 8.0 上的兼容性,需要执行用例 N 条。

测试步骤:

(1)测试人员在 IE 6.0 上对每条用例录制执行的步骤,创建测试脚本,如点击菜单、输入数据等;

(2)根据每条用例对应的预期结果设置检查点,如判断某个功能是否生效,提示信息是否正确;

(3)将编写好的脚本在 IE 7.0 和 IE 8.0 上分别进行回放,并对比实际结果,如果和实际结果一样,则通过,不一样,则失败。

通过比较可以看出,手工测试的工作量是巨大的,同一条用例需要在支持的所有浏览器上手工执行一遍,即 $3 \times N$。而自动化测试只需要一次编写好测试脚本,即可在多款浏览器上进行回放操作,特别适用于需要测试的浏览器种类很多的软件。但它也有一定的局限性,主要问题在于:(1)录制脚本时需要根据预期结果设置检查点,

存在额外的工作量。(2)对于浏览器经常出现的界面问题,如界面布局、样式、背景颜色等显示问题,则无法通过自动化进行测试,还是需要人眼去查看。

对于设置检查点和检查界面问题,本文采用的是图像匹配的方法,图像匹配是通过对影像内容、特征、结构、关系、纹理及灰度等的对应关系,相似性和一致性的分析,寻求相似影像目标的方法[7]。通过图像匹配可以省去很多需要设置的测试检查点,并且只需要将匹配图和参考图进行对比,便能很快地找出页面中存在的问题。

2 浏览器兼容性自动化测试方法

针对现有的 Web 应用程序的浏览器兼容性手动测试效率低及自动化测试中无法设置检查点等问题,提出了一种新的测试方法,并针对该方法设计了一种针对 Web 应用程序的浏览器兼容性测试的自动化测试工具[8]。为满足用户对浏览器兼容性测试的自动化工具的测试需求,要求该功能能够满足以下几个特点[9]:(1)支持输入测试脚本命令,编写和执行测试脚本,在脚本执行过程中可以调用方法库中的函数进行操作;(2)能够模拟人工对浏览器的操作,根据设置的检查点比对操作结果是否正确,同时对于不易设置检查点的界面可以采取截图的方式进行检查;(3)自动生成测试报告;(4)从软件使用的效果来看,在很大程度上代替了浏览器兼容性测试中大量的重复人工劳动,节省了大量的测试时间和人员;(5)从测试结果来看,能够精确直观地向测试人员展示测试结果,从而能够避免不必要的人工疏忽。

2.1 整体功能架构

本方法通过各系统组件及相互协作,来达到自动化进行浏览器兼容性的测试目的,自动化测试框架如图 1 所示,主要包含以下几种组件[10]:

控制中心:整个系统的核心部分,控制整个自动化的运行过程;

脚本运行器:测试 PC,安装一款需要测试的浏览器,通过接收控制中心的命令启动浏览器并执行脚本;

图像比较器:用来比较参考图(脚本运行器 1 运行脚本过程中的截图)和匹配图(脚本运行器 2 运行脚本过程中的截图)是否一致,由程序实现;

方法库:存放控制中心运行过程中需要调用的函数和方法,由程序实现;

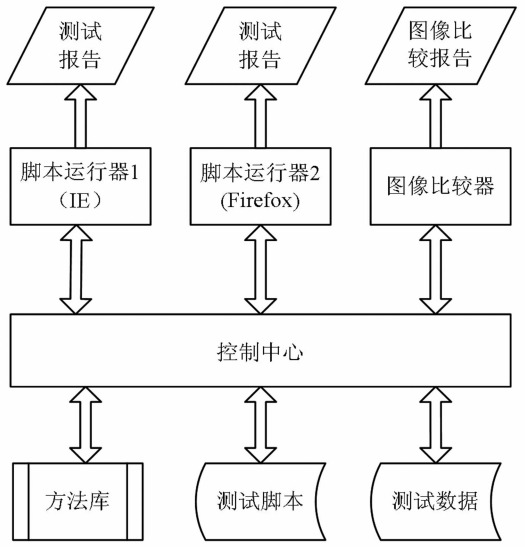

图 1 浏览器兼容性自动化测试框架

Fig. 1 Automatic testing framework of browser compatability

测试脚本:存放需要运行的测试脚本;

测试数据:存放脚本执行过程中需要测试的数据;

测试报告:输出脚本执行的每个步骤是否成功的报告;

图像比较报告:输出图像是否匹配的报告。

该自动化测试框架的工作流程如下:

第一步:编写测试脚本。根据之前写好的测试用例编写测试脚本,每条测试用例对应一个测试脚本。

第二步:准备测试环境。准备好浏览器兼容性自动化测试的软件和硬件。硬件包括 PC 设备、软件则包括控制中心和方法库的程序,测试脚本及测试数据。

第三步:执行测试脚本。由控制中心逐个脚本开始执行,控制中心对两台装有 IE 和 Firefox 的浏览器的 PC 发出脚本执行的命令,每执行完一条脚本后两台 PC 都会分别输出一个测试结果成功或者失败。如果脚本中的检查点需要查看界面的,则需要截图,将两台浏览器分别截取的界面进行比较,一般 IE 浏览器截取的界面为参考图,而另外的一款浏览器截取的界面为参考图,图片比较完成会生成图像比较报告。

第四步:输出测试报告。所有测试脚本执行

完成后,会根据脚本执行情况自动输出测试报告。

2.2　编写测试脚本

每种自动化测试工具都有自己的测试脚本,通过编写测试脚本,可以减少测试人员的工作量,提高测试效率。该 Web 应用程序的浏览器兼容性自动化测试工具的测试脚本可以看作一系列命令或者是执行步骤的集合,这些命令的集合可以组成一个测试用例执行的全过程,并且它可以解析的脚本是 Excel 文件编写的,便于初学者尽快掌握测试脚本的编写方法。在编写测试脚本之前,需要测试人员根据应用软件需求规格说明书,编写测试用例,然后根据测试用例编写测试脚本和设计测试数据。除了需要考虑到测试用例在执行前的测试环境以及一些必需的操作外,还需要为每条测试用例设置一个测试脚本的名称,并将每个测试脚本分为若干个操作步骤,每个操作步骤又包含以下几个组成部分:

1)界面及操作说明:是 Web 页面中对 Web 元素的操作说明。

2) GUI:指对 Web 元素操作的控件类型,常见的控件类型如:button、checkbox、text_field 等。

3)属性名:通过 IE Developer Toobar 工具查找所要操作的控件的属性名,可以用 ID、name、text 等表示,一般用 ID 表示,因为 ID 都是唯一的。

4)属性值:通过 IE Developer Toobar 工具查找所要操作的控件的属性名对应的属性值。

5)操作:对该控件进行怎样的操作,如单击、选择、输入等,也可以自定义操作类型,自定义的操作需要加入到方法库中。

6)值:输入的测试数据,一般为要输入到软件中的数据或者需要验证包含的数据,测试数据可以为空,可以有一个或者多个。

2.3　测试环境准备

在测试脚本执行之前,需要做一些准备工作,首先需要搭建测试环境,可以根据需要测试的浏览器数来决定 PC 的使用数。图 1 中使用的 PC 有 2 台,脚本运行器 1 上安装的浏览器为测试人员编写脚步时参考的浏览器,这样可以保证该浏览器对应的界面都显示为正确;脚本运行器 2 上安装一款需要测试的 Firefox 浏览器,也可以是其他的浏览器;其中控制中心的程序以及使用到的方法库、测试脚本和测试数据可以放在 IE 浏览器所在的 PC 上,也可以单独用一台性能较好的服务器运行。将所有硬件设备及网络环境搭建好后,需要修改控制中心程序运行的参数,设置控制中心的公共参数如表 1 所示:

表 1　控制中心公共参数设置

Tab.1　Public parameters setting of control center

参数说明	参数名称	参数值
需要测试的软件访问地址	SoftwareAccess	http://192. 168. 1. 2:8088/software/
脚本运行器 1 对应 PC 的 IP	PC1_IP	192. 168. 1. 3
脚本运行器 2 对应 PC 的 IP	PC2_IP	192. 168. 1. 4
脚本运行器 1 对应 PC 的浏览器	PC1_ Browser	IE 6. 0
脚本运行器 2 对应 PC 的浏览器	PC2_ Browser	Firefox

2.4　执行测试脚本

脚本编写和环境准备好后,就可以启动控制中心程序,开始执行脚本。具体控制中心软件工作的流程如图 2 所示。

每执行完一个脚本后,会输出一个测试结果,同时会判断是否有截图输出,如果有截图输出则将其他浏览器下的界面截图和 IE 浏览器下的界面截图输出到图像比较器进行对比,当然我们需要保证截取的图片大小一致,对比后输出图像比较结果。当两幅图片不匹配时则由人工确认是否是 Web 页面布局不合理、控件缺失、白页等界面问题引起的,还是自动化程序或者脚本本身的问题。一个脚本执行完并且图片比较完成后,控制中心会自动运行下一个脚本,保证所有需要执行的脚本完成执行操作并输出测试报告。

2.5　输出测试报告

每个测试脚本执行完成后,会输出一个测试结果 pass 或 fail,等所有脚本执行完后,会输出一个测试执行报告,测试执行报告的内容如表 2 所示。如果脚本执行过程中有输出测试浏览器的界面截图,则会将输出的界面截图与参考浏览器的界面截图进行比较,每个界面截图比较后会输出一个测试结果 yes 或 no,所有图像比较完成后会输出图像比较报告,图像比较报告的内容如表 3 所示。

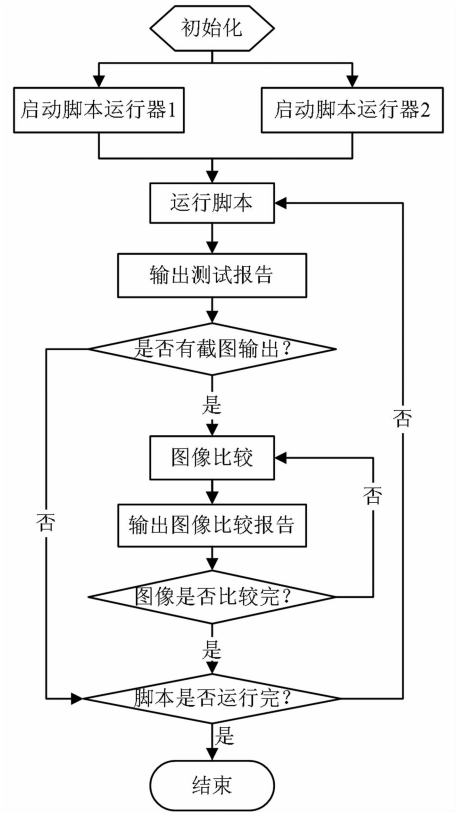

图 2 控制中心工作流程图

Fig. 2 Work flowchart of control center

表 2 测试执行报告

Tab. 2 Test execution report

测试浏览器	测试用例编号	测试结果
Firefox	TC1_01	pass
Firefox	TC1_02	fail

表 3 图像比较报告

Tab. 3 Image comparison report

参考图	匹配图	是否匹配
TC1_01 _IE6. 0_01	TC1_01_ Firefox_01	yes
TC1_02 _IE6. 0_01	TC1_02_ Firefox_01	no
TC1_02 _IE6. 0_02	TC1_02_ Firefox_02	yes

根据输出的测试报告,测试人员可以对执行失败的脚本进行分析,确认是否为 Web 程序的 bug,或者是脚本和自动化程序的错误。如果为 Web 程序的 Bug,则提交给开发人员解决,如果为脚本和自动化程序的错误,则由测试人员进行修改。同样根据图像比较报告,测试人员也需要对其进行分析,确认是否为 Web 程序的界面 Bug。

3 测试结果与分析

使用 Ruby 程序实现了所提出的测试框架。为验证所提出方法的有效性,进行了以下实验:针对某公司的一款 Web 网管软件,在相同测试环境下,对相同测试用例在三款不同浏览器上进行自动化测试,记录每个功能模块用例执行时间,该执行时间不包含自动化执行过程中必须的参数设置、执行结果确认时间。另外记录测试人员手动执行每个功能模块用例所需要的大致时间。图 3 为实验中设备管理模块的 10 条用例在 IE 6.0 浏览器上执行输出的测试报告,该测试报告包含浏览器型号、测试用例编号及对应的测试结果,并记录总共执行的测试用例数、测试失败执行用例数及测试总共消耗时间。通过该图可以看到用例 6020804 – TC1 和 60410 – TC11 执行结果失败。

图 3 设备管理模块测试报告

Fig. 3 Test report of equipment management module

图 4 为用例 6020804 – TC1 脚本在自动化测试执行过程中的详细记录,其中包含每个步骤的执行时间及执行结果。通过该记录可以查找到执行失败的命令,从图 4 中可以看到第 8 个步骤执行失败(该用例脚本总共包含 58 个步骤),进一步可以针对其分析失败的原因。测试脚本执行失败的原因主要包括脚本命令错误和 Web 程序缺陷。

最后对该网管软件其他四个模块分别进行自动和手动测试,对测试执行时间进行统计,统计结果如表 4 所示。

从表 4 中可以看出自动化测试相比人工测试在测试精度上没有太大区别,但在执行时间上却大大缩短。使用所设计的自动化测试工具在三款不同浏览器上执行总计 35 条用例平均花费时间

图 4　用例 6020804 – TC1 执行步骤记录

Fig. 4　Implimentation step (record) of example 6020804 – TC1

为 2 h 左右。而使用手工测试同样的 35 条用例,则需要大约 6 h。可见相比于传统人工测试,使用该工具可以使测试效率得到明显的提升,大约为人工测试的 3 倍。

4　结 论

目前浏览器兼容性测试以手工测试为主,通过引入自动化测试框架及重新设计测试环境,实现多浏览器兼容性的并行自动化测试,并结合图像匹配技术,解决了测试脚本无法设置界面检查点的问题。实验结果表明,相比手工测试,该方法能有效地提高测试效率,缩短测试时间,使测试人员从繁重的手工测试执行工作中解放出来。

对于该方法还有探索和研究的空间,如可以增加脚本运行器的数量,进一步减少浏览器兼容性测试时间;通过改进现有的图像匹配技术,如引入文字识别的技术,可以对 Web 界面上的文字进行校验,进一步提高自动化程度。

表 4　测试结果记录

Tab. 4　The record of test results

功能模块	测试内容	自动化测试时间/min			手工测试时间/min
		IE 6.0	IE 8.0	Firefox	
设备管理	网络设备信息管理设备操作 设备接口管理(执行 10 条用例)	41	38	35	127
配置管理	设备配置管理 设备软件管理(执行 7 条用例)	26	24	21	58
性能管理	查看性能实时曲线 关键设备实时监视(执行 4 条用例)	15	13	13	34
告警管理	实时告警监视告警通知 告警规则设置(执行 8 条用例)	33	32	28	114
系统管理	系统参数设置管理员管理 日志记录(执行 6 条用例)	22	21	20	49
总计	35 条用例	136	129	117	382

参考文献:

[1] 董启雄,唐清安,陈广旭. 对几款浏览器兼容性的测试分析[J]. 计算机光盘软件与应用,2012(18):66 – 67.

[2] 陈广旭,董启雄,栗勇兵. 不同操作系统下浏览器兼容性测试的研究[J]. 计算机光盘软件与应用,2012(18):64 – 65.

[3] 岑柏滋,刘丽琳. 浅谈 Web 应用系统的测试[J]. 电脑与电信,2008(2):29 – 31.

[4] 苏海明. 图书馆学常用数据库的浏览器兼容性测试[J]. 图书与情报,2008(3):61 – 67.

[5] 游强华,王萍,黄伦东,等. 高校图书馆网站对浏览器兼容性测试[J]. 图书馆杂志,2011,30(10):66 – 71.

[6] 黄侨,葛世伦. 开源 Web 自动化测试框架的改进研究[J]. 科学技术与工程,2012,20(15):3630 – 3635.

[7] 杨晓敏,吴炜,卿粼波,等. 图像特征点提取及匹配技术[J]. 光学精密工程,2009,17(9):2276.

[8] 冯振华,高菊,曾红卫. Web 应用自动化测试的研究[J]. 计算机工程与设计,2010(1):175 – 178.

[9] 杨怡君,黄大庆. Android 手机自动化性能测试工具的研究与开发[J]. 计算机应用, 2012,32(2):554 – 556.

[10] 邓正宏,高逦,郑玉山. 面向对象自动化测试框架的研究与设计[J]. 微电子学与计算机,2005,22(2):168 – 171.

(责任编辑: 肖锡湘)