基于Android的自动化测试的设计与实现

谢红霞，吴红梅

(浙江大学城市学院，浙江 杭州 310015)

摘 要： 以Android客户端的测试为研究内容，分析了Android自动化测试框架及其层次关系，尝试在现有测试方法的基础上进行测试手段的优化和创新。研究了基于JUnit和instrumentaion的Android自动化框架的实现技术，并利用Hudson进行集成，实现Android的自动化测试。产品的开发实践表明，这种自动化测试方法效率高、用户体验好，对于Android的自动化测试研究具有一定参考价值。

关键词： 测试自动化；Android；手机；设计

中图分类号： TP306+.2 **文献标识码:** A **文章编号:** 1006-8228(2012)02-20-03

Design and implementation of automated testing based on Android platform

Xie Hongxia, Wu Hongmei

(*City College of Zhejiang University, Hangzhou, Zhejiang 310015, China*)

Abstract: This paper focuses on testing of Android clients. The authors analyze the Android automated testing framework and its hierarchy. The implementation is based on two automated testing framework of Android platform: JUnit and instrumentation. The purpose of this article is to propose an optimized test method. Making use of the continuous integration features of Hudson, the real sense of automated testing can be achieved. It shows that this automated testing method improves efficiency and enhances user experience in some extent. This research will have values for automated testing of Android.

Key words: automated testing; Android; mobile phone; design

0 引言

随着新一代无线网络技术的发展，未来互联网的重心会从传统的计算机转移到新一代移动设备上。而随着Android开源手机系统的逐步普及，Android已受到手机生产厂商、移动运营商、手机应用开发商的广泛关注可以预见将会有更多基于Android的手机以及平板电脑出现。这些产品出现以后，就会有很多相应的应用，对测试人员来说，这些应用的测试工作是一个重大的课题，特别是基于Android的自动化测试。本文研究了基于Android的自动化测试的设计与实现技术。

1 Android自动化框架

Android是Google与开放手机联盟合作开发的全球首个完全开放的手机平台，其最大的特点就是开源、免费、智能。Android是基于Linux内核的操作系统，采用软件堆层(software stack)的架构[1,2]。

1.1 Android系统结构

Android系统主要分为四层，分别是应用程序层、应用程序框架层、系统运行库层和Linux核心层，如图1所示。上面两层为Java程序，第三层是为Java运行的虚拟机及C/C++编写的程序库，第四层则是Linux核心代码和驱动层[3,4]。

开发中我们接触最多的是应用程序框架层，当一些现有类方法无法满足需求时就会对这一层进行扩展。从测试的角度讲，必须先了解开发的方法和系统内部的结构，才能使测试更为高效，定位错误的效率更为快速。

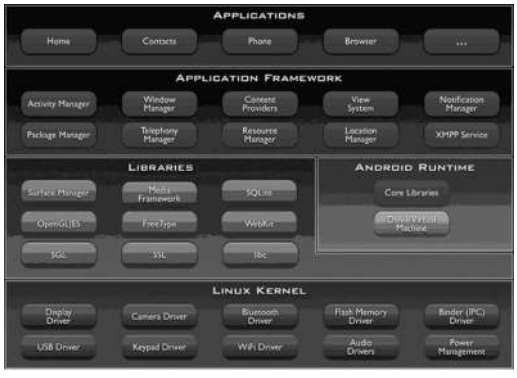

图1 Android系统框架图

1.2 常用Android测试手段

Android常用的测试手段有[5]：

(1) CTS用来确保设备符合Android兼容性规范。

(2) Monkey用来进行压力测试。无论是应用程序还是系统都可以使用它来测试其负载能力。

(3) ASE为Android 带入脚本语言，通过脚本(如 Python)

收稿日期: 2011-11-14

作者简介: 谢红霞(1971-)，女，浙江余姚人，主要研究方向：计算机软件测试，网络存储。

调用 Android 的功能，从而定制一些测试。

(4) 用Robotium工具实现黑盒的自动化测试，可以在有源码的情况下或仅有APK文件的情况下对目标应用进行测试。Robotimu 提供了模仿用户操作行为的API，例如在某个控件上单击，在Text 控件中输入信息等等。

(5) 单元测试。Android 整合了JUnit测试框架和Instrumentation机制，可以针对某个应用进行单元测试。因其功能强大，本文将在后面作重点讨论。

上述测试手段各有特色。对于 CTS/Monkey，不需要开发，直接执行测试就可以了。对于ASE，可以扩充它的现有API (Java)，用 Python 调用 API 实现丰富的测试功能。而用Robotium可以模仿普通用户行为，把一些原来由测试工程师作的测试变成Robotium自动化实现[6]。

1.3 Android现有测试方法

首先搭建测试环境，安装AndroidSDK（开发包）、ADT/DDMS（开发插件。在AndroidSDK中已经给出了如何在系统上进行测试的方法，如图2所示。

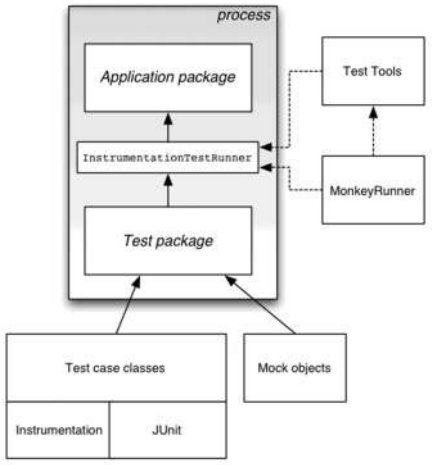

图2 测试框架图

(1) Android中的JUnit

JUnit采用测试驱动开发的方式，也就是说在开发前先写好测试代码，说明被测试的代码会被如何使用，错误处理等，然后开始写代码，并逐步测试这些代码，直到最后通过测试。AndroidSDK中所带的JUnit的功能如表2所示。

表2 AndroidSDK中的JUnit

类	功能说明
junit.framework	JUnit测试框架
junit.runner	实用工具类支持JUnit测试框架
android.test	Android 对JUnit测试框架的扩展包
android.test.mock	Android的一些辅助类
android.test.suitebuilder	实用工具类，支持类的测试运行

(2) Android中的Instrumentation类

在图2的测试包内除JUnit以外，还包含Instrumentation类。JUnit测试是自动进行的，而这个自动测试的过程就是由Instrumentation来完成的。Instrumentation类似于Windows里的“钩子（hook）函数”监听系统与应用间的各种行为。它在应用程序运行前初始化，与应用程序运行在同一个进程当中，监听与系统的交互过程，包括activity的开始、结束。还可以控制组件的一些交互事件，包括按键，拖拉等行为事件。

(3) Nstrumentation对JUnit的扩展

从SDK中我们还可以看到junit.framework中有以下树如图3所示：

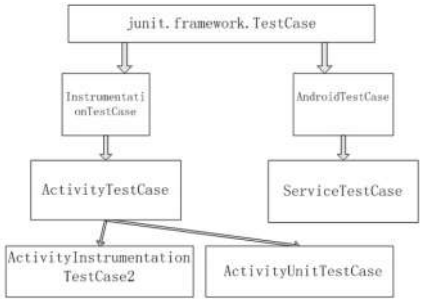

图3 junit.framework中的测试扩展

图3中所示的是比较重要的几个扩展。比如淘宝的客户端首页的Activity是TaoActivity，我们想测试它上面的一些功能，就可以实现一个叫作ActivityInstruemtationTestCase2<TaoActivity>的子类，而当这个子类运行的时候，TaoActivity就会自动启动，而不需要调用intent的方法。然后用Instrumentation来模拟用户的点击、拖拉等行为，这样就实现了自动化测试。从所建的工程来看，将setup()方法重写，然后在setup()方法里面调用自己封装的方法，就会使得整个测试更简洁。

```
public class TEST extends ActivityInstrumentationTestCase2{
    private Dox dox;
        public TEST() {
            super("com.taobao.taobao", MainActivity2.class);
        }
    @Override
    public void setUp() throws Exception {
        dox = new Dox(getInstrumentation(), getActivity());
    }
    public void test() throws Exception
    { }
}
```

可以通过直接调用封装的dox方法来调用一些用户基本行为，如点击、拖拉等。下面的代码就是调用sendKeyDown–UpSync()方法实现简单的返回。在常规测试中若要调用返回必须写相当量的代码，而封装以后就可以直接调用dox.goback()方法来实现）[7]。

```
public void goBack() {
    sleeper.sleep();
    inst.waitForIdleSync();
    try { inst.sendKeyDownUpSync(KeyEvent.KEYCODE_BACK);
        sleeper.sleep();
    } catch (Throwable e) { }
}
```

2 Android自动化测试的设计及其实现

流程控制以及对时间优化分析，自动化测试流程为：

⑴ 项目启动阶段。创建项目空间,同时参与产品需求评审并给出测试方面的建议。

⑵ 项目策划、需求分析阶段。参与UC评审,对开发实现方式进行沟通,编写测试计划。

⑶ 系统设计阶段。在系统设计的同时进行测试设计。

⑷ 编码阶段。测试部门编写测试用例并通知项目内成员进行评审。

⑸ 测试阶段。测试部门执行三轮测试,冒烟测试、集成测试、回归测试手段。

此外还需要对测试手段进行优化,除了传统的功能测试以外,引入接口测试等各种辅助测试以从整体上优化测试,使测试贯穿整个产品开发过程。

2.1 系统架构介绍

Android自动化测试架构有3个部分组成,包括测试管理服务器、测试设计执行客户端、移动设备脚本。

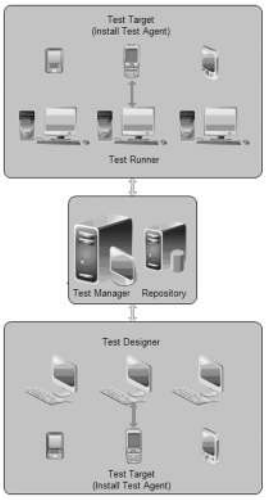

图4 系统架构

⑴ 测试管理服务器

该部分主要负责管理所有测试资源。包括测试设计执行客户端、测试设备、脚本及日志的管理;管理测试任务,监控测试状态,发送测试报告,保存测试信息等。

主要的执行流程包含以下几个部分:

①用户通过测试设计客户端远程登录到测试管理服务器。

②配置测试管理服务器中的测试任务。

③执行测试任务。测试管理服务器远程打开测试执行客户端,将其在模拟器或者真机上执行测试任务。

④完成测试任务后将报告收集到服务器保存。

⑤发送测试报告到指定用户。

⑥服务器主要是用hudson来实现以上任务,通过hudson来控制整个资源。

⑵ 测试设计执行客户端

客户端就是搭建了Android开发环境的电脑。设计客户端就是编写测试用例的机器。而执行客户端是绑定了真机或者模拟器的机器(通过有线或无线网络)。它们的核心是使用Android自动化框架完成测试编写工作。

⑶ 移动设备脚本

移动设备是指安装了Android系统的目标机器。将开发完成的脚本放到该目标机器内,通过远程命令启动该脚本,脚本被执行,统计结果被返回。

2.2 Android工程自动化实践

在一个团队中,如果要发布产品,就会按照发布流程规范进行。主要步骤有:

①运营部门将渠道号提交给开发,要求开发部门将最新的客户端进行打包。

②开发部门将开发完成的包提交给测试进行发布测试。

③测试部门通过后通知开发。

④开发部门使用渠道号进行打包。

⑤将打包完成的包提交给运营。

⑥运营部门分发给各个渠道。

运营、开发、测试的流程如图5所示。

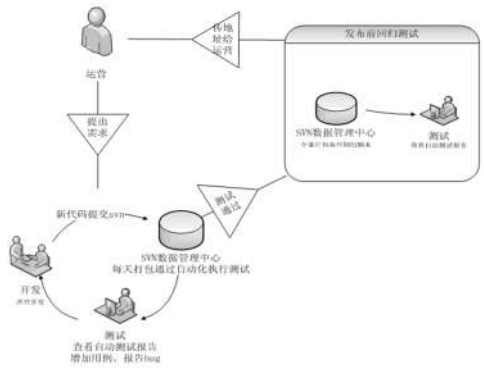

图5 软件发布流程

3 结束语

本文针对Android新一代移动设备的测试方法进行了优化和创新,设计开发了基于Android的自动化测试架构,并对目标客户端进行自动化模拟及用户行为的回归。系统的用户体验非常好,提高了开发团队的工作效率。后续将对框架进行二次开发,并进一步增加对系统性能部分的评估功能,提高对错误定位的准确性。

参考文献:

[1] 戴建国,郭理,曹传东. JUnit框架剖析[J]. 计算机与数字工程,2008.8:43~45,135

[2] 白凯,崔冬华.基于JUnit自动化单元测试的研究[J]. 计算机与数字工程,2010.2:52~54,103

[3] 刘巧玲,范冰冰,黄兴平. 基于Hudson的持续集成研究和应用[J].计算机系统应用,2010.12:151~154

[4] 戴建国等.持续集成在项目开发中的应用研究[J].计算机工程与设计,2009.10:2573~2576

[5] 陈哲,张国平,丁玉斌. 软件自动化测试方案研究[J].软件导刊,2007.13:35~36

[6] Rogers, R.. Android application development. 1st ed. ed. 2009, Sebastopol, Calif.: O'Reilly. 318.

[7] 林毅文. Android Visual User Interface测试技术[J].电子与电脑,2011.4:68~71