GUI自动化测试框架的研究与实现

江 鲸[1], 罗惠琼[1], 吴凯华[2]
(1.电子科技大学计算机科学与工程学院 四川 成都 610054; 2. UT斯达康深圳研发中心 广东 深圳 518057)

【摘 要】 WinRunner是当今图形化测试领域里功能最强大的工具之一,如何灵活高效的利用此工具实现自动化测试是提高GUI产品测试质量和效率很重要的一个途径。本文讲述了一个基于WinRunner测试工具本身开发的一个自动化测试用例开发框架,详细描述了它的设计原因,实现思想和原理,以及如何与自动化测试系统ATS进行集成。

【关键词】 GUI测试; WinRunner; GUI Map; TSL; 自动化测试系统

0 前言

WinRunner是MI(Mercury Interactive)公司的一个图形界面自动化测试工具,从上世纪90年代中期WinRunner便开始在欧美广泛被应用在测试上,至今已有成千上万公司的产品测试通过此工具自动完成,自几年前进入中国后,WinRunner以它本身强大的功能和灵活的可编程性正在吸引越来越多的中国工程师加快对它的学习和应用。

1 WinRunner简介

WinRunner需要在Windows操作系统上运行,用户可以简单的通过'录制-回放'的特点来完成一个测试用例,在WinRunner录制的过程中,它可以自动捕获检测当前的界面,把用户在界面上的操作自动转换成可描述性的语言和事件,之后可以对相应的参数进行必要的更改/提取,以及加入相应的测试点检查,然后回放此次录制过程完成某个测试任务。

GUI Map在WinRunner里是一个非常重要的知识点。所有界面上的显示内容,在WinRunner里都被标识为对象,如文本编辑框、按钮、图片、超级链接,这些对象被存在一个称为'GUI MAP'的属性文件里。一个GUI MAP文件实际就相当于一组GUI对象的集合体,窗口、按钮、菜单、列表、链接、图片、文本等所有对象被视为此GUI MAP的一个元素,这些对象属性会在学习一个界面的对象时自动添加到GUI MAP里。所有的对象以树状结构组织在GUI MAP编辑器里,点中某个对象后,在"Physical Description"会显示此对象的详细属性,用户可以点击"Modify"按钮来对此对象进行编辑更改。GUI MAP文件可以通过调用GUI_load(gui_map_file)来把此GUI MAP文件里的所有对象加载到某测试脚本里去,可以调用GUI_close(gui_map_file)把刚加载的对象卸载掉。

TSL是WinRunner工具的开发使用语言,它的语法结构类似C语言。作为WinRunner自己独特的编程开发语言,TSL具有一定的使用局限性,离开WinRunner,没有任何一个地方可以使用此语言,然而TSL本身还是提供了功能强大灵活的编程机制,有利保障了对WinRunner最大限度的使用,只有精通TSL的编程之后,才能对WinRunner的使用做到游刃有余。

2 WinRunner自动化架构介绍

作为一个自动化测试工具,WinRunner可以非常灵活地实现录制-->回放的使用机制,然而在大多数情况下,用户期望所录制的某段脚本可以在不同的被测对象上运行,同时因为在不同环境下所使用的资源(例如电话号码,IP地址等)会不同,如果单纯的录制回放,当有大量的脚本存在并且需要调整脚本里的资源参数值时,对所有脚本一行行的检查改动将会是一个巨大的工作量,必将大大增加以后的维护工作。另外,当有大批量的测试脚本运行时,用户期望测试脚本能够把大量的日志和运行情况统一记录在某个指定的地方,等所有脚本运行完后,可以一次性查看所有的信息,完成后续的用例测试结果分析和调试,而不是依赖WinRunner本身所带的执行结果来分析测试情况(因为这样需要花费大量时间在WinRunner图形化结果分析工具里来查找某个错误原因,而往往在此情况下,大多数的测试结果数据不是用户所关心的)。另外还有一个非常重要的一点是如果单纯的录制-->回放,而没有一个测试脚本的规范,每个用户的脚本格式都可能不一致,相互之间的维护支持将会非常困难。

基于以上情况,在WinRunner里提供一个通用的平台,让所有用户的测试用例具有统一的风格和执行模式,可以灵活的配置测试参数和数据便成为一个势在必行的工作。因此,有必要对WinRunner进行封装和二次开发。基于此,我们开发出了一个内嵌于WinRunner工具内的一个框架,同时还期望只要用户知道对应的自动化测试用例的编写规则,可以完全忽略此框架的存在,不必了解此框架的工作原理,便能轻松的编写自动化测试用例。

2.1 功能概括

此框架现在支持以下所有功能:

2.1.1 参数可配置化

这个基于WinRunner的平台支持参数配置文件化,可以读取分析配置文件,确定具体要执行的测试用例,以及测试用例里参数的设置值。

2.1.2 测试结果输出到固定文本文件里

框架提供了一个结果保存机制,所有的测试用例的Log信息可以被完整的保存下来,这样就方便了后续的分析调试工作,并且测试结果的Log信息格式固定,还可以提供一个结果分析工具来实现对此测试结果的自动分析和报告。

2.1.3 统一的测试用例编写规则

统一的Case编写规则让其他的开发人员非常容易读懂他人的自动化用例,大大有利于以后的自动化用例的维护和扩展。

2.1.4 自动地运行所有在配置文件里指定的所有用例

有了统一的参数配置方法,同时运行很多测试用例便非常容易,此框架可以支持对在配置里的所有用例进行分析,然后去调用相应的自动化用例执行它们。

2.1.5 与ATS有机的集成

提供了与ATS有机集成的方案,用户可以利用ATS在某一指定时间运行测试的功能运用在Web自动化测试上,可以实现晚上无人监控下自动测试。

2.2 框架设计思想/原理

框架设计如图1所示。它使用TSL开发完成,它其实已经完全自动融合到WinRunner里,只要实现完成一个固定的配置分析和Case运行调度即可;用户不需关心此框架的工作原理,只要按照固定的格式来编写配置文件和自动化测试用例,之后向框架发起运行的请求,等测试完成之后,分析测试结果即可完成相应的任务。

当它收到用户的运行请求后,分析用户提供的配置文件,然后调用对应的测试用例,向WinRunner发起运行命令,在用例运行过程中,不断产生测试信息和结果。

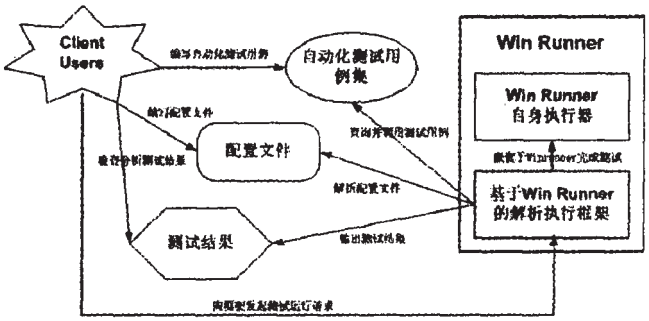

图 1 WinRunner 自动化框架思想

配置文件必须是按照规定的格式编写，当检测到无法识别的配置信息时，此框架会产成一个错误日志，然后退出执行。所有的配置会被分析然后存放在一个数据结构里，框架开始分析所指定运行的 Case，从 Case 数据集里把对应的 Case 加载到当前运行环境里，顺序运行所有的 Case。如果监测到某一 Case 运行失败，框架仍然会继续往下运行，也就是说，某一 Case 的运行情况不会对其他 Case 的运行带来影响，框架本身已经作了必要的出错控制。在运行过程中，关于 Case 的所有参数数据和用户在 Case 里输出的信息会自动带着运行时间存放到测试结果日志里，供以后用户浏览分析。

下面是此框架的实现详细流程图 (全部是基于 TSL 开发完成)

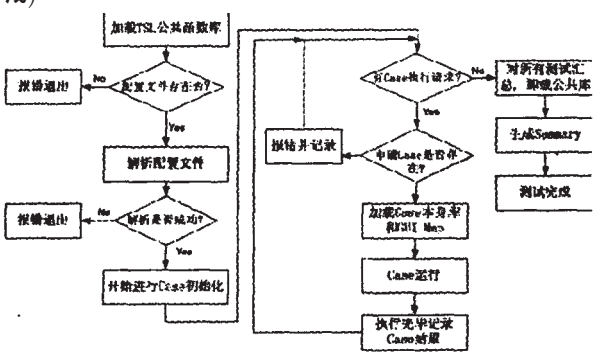

图 2 WinRunner 框架设计实现流程

2.3. 框架与 ATS 的集成

如图 3 所示，WinRunner 自动化框架与 WinRunner 放在一起，共同安装在一台 PC 机上，在此 PC 机上，有一个基于 TCP/IP 的 socket 守护进程一直处于运行监听状态，ATS Server 端作为此 TCP/IP 的客户端，当有测试请求时，向 WinRunner 所在的 PC 机上的守护进程发出处理请求，当得到肯定回复后，把测试的详细数据通过 TCP/IP 发送过去，在 PC 机端，守护进程会调用另外的命令完成对配置文件的封装，然后调用 WinRunner 自动化框架进行真正的测试。测试完成后，仍然通过 TCP/IP 把测试结果和详细的日志信息发送到 ATS Server 上，整个运行告一段落。

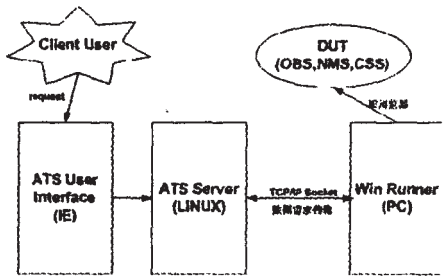

图 3 与 ATS 的集成

3 WinRunner 自动化框架的应用

对于 GUI 的自动化测试，ATS 现在支持两种方式的自动化 Case 的开发：

1) 一种是基于 WinRunner 本身编辑器，按照框架规定的开发规则来开发。这种方法必须生成一些实实在在存在的 WinRunner 脚本，放在 WinRunner 对应的 PC 机上，在 ATS Server 这边仅仅要求根据对应 Case 的名字去调用执行。

这种方法的好处是比较简单易懂，只要掌握了框架的编程规则，自己可以任意编写自己需要的脚本。同时用此方法编写的 Case 以后完全可以脱离开 ATS 来独立运行。缺点是要有大批的 WinRunner 脚本被开发出来，以后维护的代价会比较高。

2) 另一种是完全基于 ATS 本身自己的 Case 开发编辑界面，让 GUI 的每一个测试步骤调用众多的已经集成于 ATS 的 WinRunner API 来完成，之后 ATS 会自动封装所有这些步骤，转化成按照 WinRunner 框架要求的自动化 Case，然后通过 TCP/IP 送到 WinRunner 所在的 PC 上完成测试。

这种方法的好处是所有 Case 的步骤都存在 ATS 数据库里，脚本只有在运行时实时产生，当有 Case 的变更时，只要修改其中某一很小的一步或几步便可，缺点是对于编程的开发能力要求较高，同时还要不停开发新的 ATS WinRunner API 来支持相关的新的测试用例，新的业务。

4 结论

自 2003 年底公司自动化组开始接触学习 WinRunner 之后，它已经被应用在我们多个产品线的测试部门。自动化组综合了 WinRunner 自身的特点和功能，进行了必要的内嵌开发，推出了一个灵活跨平台的自动化框架，现已经被无缝集成到了自动化测试平台(ATS 系统)中，强有力的支撑起了 Web 的自动化功能测试。

参考文献

[1] 贺平著 《软件测试技术》机械工业出版社 2004 年
[2] [美]Brent B.Welch 著 崔凯译 《Tcl/Tk 编程权威指南》中国电力出版社 2002 年
[3] [美]Kanglin Li, Mengqi Wu 著 王铁昆译 《图形用户界面测试自动化》电子工业出版社 2005 年