Design and Achievement of User Interface Automation Testing of Linux Based on Element Tree of DogTail

Wen-Chao Yuan [1,a], Hong-Mei Zhu[2] and Zi-Qian Sun [3]

[1]College of Computer Science and Technology, Shan Dong Agricultural University, Tan'an 271000;
[2]Red Hat Software(Beijing)Co., R&D Branch, Beijing 100190
[a] Corresponding author:yuanwenchao1@163.com

Abstract. As Linux gets more popular around the world, the advantage of the open source on software makes people do automated UI test by unified testing framework. UI software testing can guarantee the rationality of User Interface of Linux and accuracy of the UI's widgets. In order to set free from fuzzy and repeated manual testing, and improve efficiency, this paper achieves automation testing of UI under Linux, and proposes a method to identify and test UI widgets under Linux, which is according to element tree of DogTail automaton testing framework. It achieves automation test of UI under Linux. According to this method，Aiming at the product of Red Hat Subscription Manager under Red Hat Enterprise Linux, it designs the automation test plan of this series of product's dialogs. After many tests, it is indicated that this plan can identify UI widgets accurately and rationally, describe the structure of software clearly, avoid software errors and improve efficiency of the software. Simultaneously, it also can be used in the internationalization testing for checking translation during software internationalization.

1 Introduction

With the development of the science technology and wide-spreading of the computer application software. Linux operating system gets more and more concerning and approving. It has the advantages of security, freedom and open source, which appeals more and more users. User Interface provides user information as carrier. And the user's experience relies on its design and development. Owing to the open source of software, Linux users can debug and upload the new version software at any time. During this process, it is necessary to implement UI testing.

UI testing is the test for user interface, which can ensure the qualification of the user interface. Recent years, many researchers are working on studying and improving the method of UI testing. He Hao [1] proposes a effective and common UI testing method that is based on Silk and XML, but this method has limit of operating system. Memon [2] proposes a method that it creates test cases by AI classification technology. However, the maneuverability of this method is not good during testing. Wu Hengshan [3] proposes a method that is based FSM testing model of Graphical User Interface state finiteness by analysing problems of GUI automation testing, which gives method of building model and testing. His method can produce GUI testing path and testing input data automatically. Wang Zhao [4] proposes that building model can utilize colored Petri net for user interface object and designs methods that are based on different testing coverage criteria. Lu Yongzhong [5] creates a GUI automation testing system which is based on event flow chart. The system implements reverse engineering for GUI of tested criteria to get event flow and expected results of it. Then the method produces test case by daily smoking testing based on Ant Colony Optimization and Depth Regression Testing based on width-first search algorithm. Finally, it tests the GUI by implementing these produced test cases. However, these testing methods are not specific for the testing framework. And it is restrained by testing bench. Zheng Yaoming [6] proposes a method of function testing system of Linux GUI which are based on LDTP testing framework. It produces test script by keyword driven technology. But this method does not involve multilingual and multi-version testing. It is difficult to implement.

Red Hat Subscription Manager is the product of Red Hat Enterprise Linux, which is the product about system subscriptions. International software requires multilingual and multi-version internalization testing and localization testing. Each kind of testing will test again and again.

This paper analyses several common testing tools firstly. Then it creates a widget identification method of element tree of DogTail based on applications of Linux [7]. It designs the automation testing plan for the product of Red Hat Subscription Manager under Red Hat Enterprise Linux. The method records UI widget information in the form of element tree and produces automation testing script [8].It is an efficient testing method in Linux operating system environment. Simultaneously, this plan can describe UI each widget's structure and connection. It shortens more time of identifying widget, reduces the redundancy of testing script and improves efficiency of testing.

2 Related Work

In more and more violent market competition, in order to improve software testing efficiency, automation testing plays an more and more important role [9]. It makes tester set free from repeated testing [10].Automation testing can finish fussy and repeated parts of the testing accurately and efficiently. It can shorten testing]time[11], improve testing coverage area and save labor source.[12] UI's automation testing relies on testing script, which can trigger UI widget automatically and check the validity of widget to ensure the qualification of the user interface. For any kind of testing such as function testing or internationalization testing, UI automation is very popular and important. There are some automation framework ideas like: testing script modularization [13], test libraries, data driven[13,14,15]and keyword driven [16]. These will be used repeatedly at many times, which is important for the repetition and function test of same system. Most important of all, it is more effective in multi-version test [17].

There are many kinds of automation testing tools at home. For example, Rational Robot testing tool is created by IBM Rational company, WinRunner is supplied by Mercury, and QTP [18] is produced by Hewlett-Packard Development Company. These automation testing tools are quick to develop script, efficient to manage script and accurate to check script.

Whereas, these tools are limited by operating system. They are almost utilized at environment of Windows operating system instead of Linux. Linux operating system differs from Windows [19].It has universal and unified [20] desktop such as GNOME or KDE. DogTail is a kind of automation testing framework for open source. Relying on the advantage of open source, more and more testers are using and updating it. It is a framework for UI programmed by python. Although it does not have test case for the application, it provides a convenient and practical framework to produce their own testing scripts. Generally, most UI testing tools achieve automation by capturing the information of visual interface at a specific application. However, DogTail sets up a memory model of UI element by utilizing metadata connected with Accessibility.

3 Method of UI Testing Based on DogTail Element Tree

3.1 DogTail Automation Testing Introduction

DogTail framework provides two kinds of Application Programming Interface (API) for testing script. One is procedural API, the other is object-oriented API. Procedural API is used for function test of desktop application. As for simple user, it can help them to write script to finish some simple operations for UI object. And it is contained in the module named procedural of DogTail. If user needs to finish some meticulous operations, the object-oriented API can achieve it successfully. This kind of API can drive the application easily to reach a specific state to finish testing and revising errors. This API is contained in the module named tree of DogTail.

3.2 DogTail Automation Testing Workflow

In the specific automation testing of UI, it needs to select a specific application. The main work is identifying information and triggering action for the widget by searching and locating dialog and widgets of window. User Interface widget consist of element of dialog. Firstly, DogTail can capture information of the dialog, which is generated by function of it. Then it will locate widgets including button, label, sub-menu, sub-dialog one by one to identify information such as attribute, word, and formation and so on. These information is on basis of writing automation testing scripts for UI. Finally, DogTail will show the feedback after executing the testing scripts. In addition, testers will get the results after analysing the feedbacks. Generally, there are several kinds of errors like UI widgets truncation, shortcut key repetition and not availability, information unlocalization and UI widgets loss and not availability.

3.3 Element Tree of DogTail Theory

In Linux operating system, its desktop applications are in the form of tree. DogTail automation testing framework is the set of many modules of scripts written by python language. For example, Tree module and procedural module can finish operations of UI. Internationalization module can achieve internationalization and localization. Different modules will finish the automation testing by cooperating each other. In the tree module which can operate with GUI elements meticulously, it contains many classes. Each class defines implementations methods. The most important class of tree modules is node class. Here is the process of the design:

Firstly, it needs to design a element tree. The root of the tree is application named ROOT. ROOT application has the higher authority. It defines another desktop applications, high-level windows and dialogs generated by ROOT application as its offsprings. And it continues to define the widgets like button, labels and textboxes of the offsprings as the next generation of it Node is the element that includes ROOT application, applications, windows and widgets. It can be instantiated in the element tree. In a summary, node class is the mixed interface to the accessible elements and accessible user interface.

Then, in the element tree, next work is to search a node and locate it for automation testing. Module named Predicate of DogTail framework can finish searching. It can search nodes with the criteria. DogTail automation testing framework applies a kind of high-level searching mode. It will search one node or some nodes which are satisfied with criteria formulated by tester with the method of 'backoff and retry' algorithm.

Finally, after searching and locating a specific node successfully, it will define this node as a variable for programming. The variable will achieve operations by invoking functions in the class named action. In the action class, it defines and achieves functions for the node. These functions are exports made by the specific node on the accessible layer. For example, it has actions like click, write.

3.4 Element Tree of DogTail Development

Linux operating system's UI automation testing is on the basis of the searching node and matching it with rules. What is different from Windows operating system is that it does not have specific testing tools except testing framework. However, because of its unity, it is convenient for every tester to test UI, which makes UI's automation testing more efficient and universal. This paper proposes the method of element tree of DogTail for UI automation testing. Here are steps of developing:

3.4.1 Preparations:

It is necessary for tester to install an Unix-like operating system such as Red Hat Enterprise Linux. And it requires to install DogTail automation framework and GUI. It still needs to install Python language support.

2. Testing Script Programming

This paper proposes establishing the model of element tree of DogTail, and coding for node on searching, locating, identifying and operating at the testing script. There are three-level identification for nodes. The first-level identification is identifying ROOT application. The second-level identification is identifying desktop applications and high-level windows and dialogs. The third-level identification is identifying widgets of windows. DogTail automation testing framework can identify node information by information of string name. After identification, it will set the node as the variable. By invoking functions and methods, the variable can make operations and language testing.
Here are the specific steps:

(1)Configure environment and import modules. DogTail need import functional modules to the testing script so that it can finish related operations. In fact, It is made of many python language scripts that can achieve sub-module functions. The definition is below:

```
#! /usr/bin/python
from dogtail import tree
from dogtail.utils import tree
from time import run
from os import environ, path, remove
```

(2)Invoke functions to start or shutdown the dialog. It will invoke the 'run' function from the module named Util. The function can start the desktop applications and windows to be tested.

(3)If it is successful to search the node, the node will be set the value to get the authority. By coding, it can finish dominating the widgets of the main dialog. If the main dialog can generate the sub-dialog, it need set value for the node of sub-dialog after matching the string name of sub-dialog. Then the sub-dialog node will continue to execute its widgets for testing.

(4)Execute methods of Action class for operation. String name is the key to be identified. DogTail can obtains the string name of dialogs and widgets of UI automatically, then matches it with them in testing script. If successful, the node can succeed to achieve operations like Click (),and Close().The definition is below:

```
#Get a handle to subscription-manager-gui's menu
Systemmenu=subscription_manager_gui.menu('Sys
tem')
Systemmenu.click()
Systemmenu.grabFocus()
Systemmenu.menuItem('Register').click()
#Get a handle to subscription_manager_gui System
Registration Dialog
Register_Dia =
subscription_manager_gui.dialog('System Registration')
Register_Dia.child(roleName = 'text').text = 'xxx
xxx xxx'
Register_Dia.button('Next').click()
```

(5)Import internationalization modules. This module can achieve testing of language translations. International software has the feature of the multilingual and multi-version. Thus as for translation testing, it needs to import related modules and language support to be tested to complete internationalization testing . It usually cooperate with gettext modules. The definition is below:

```
Def translateAllStrings(appName):
""
Test of the translation functions.
Take all user-visible strings in an app that's running
in the default localeand try translating them all into
locale that this script is running in.
"""for string in
root.application(appName).getUserVisibleStrings():
print(("User-visible string:%s" %string))
print(("Translation
is :%s"%dogtail.i18n.translate(string)))
print(("Packagedependencies:%sdogtail.distro.pack
ageDb.getDependencies('subscription-manager-gui')))
```

#print dogtail.i18n.getMoFilesForPackage('subscription-manager-gui',True)

print (("Translation domains:%s"%dogtail.i18n.getTranslationDomainsForPackage('subscription-manager-gui',True)))

(6)According to above method, it need to go through all the nodes of the element tree of the tested application to identify and check the information and function of the UI's widgets.

4 Analysis of the Testing Result

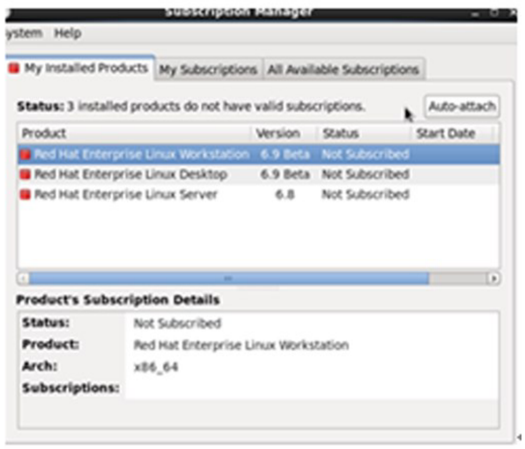
4.1Automation Testing Result Analysis

For the basic and weakly logical operation on the Linux user interface like function testing and stress testing, DogTail framework can achieve the goal clearly. By writing simple Python language testing script, it can finish numerous repeated operations. In addition, there are convenient feedbacks to be reference for testers to conclude testing results. It saves more manpower and resources. For the strongly logical operations, testing script will often need to import more modules to cooperate to complete testing.

As for the testing result, it is based of feedbacks after executing testing scripts. This paper will compare it from the aspect of logicality and time index with traditional manual testing, which can show the advantage of DogTail testing framework. Here is the specific analysis:

4.2 Feedbacks Analysis of Testing Script

After locating a specific node, DogTail can invoke the node's main dialog and test it by the method of this paper. The testing script should defines and quote information from the main dialog and widgets of the dialog including its string name, location and invalid area. Besides information, another definition in the scripts is actions to be executed of the widget. Figure1 shows the tested dialog.Figure2 shows the feedbacks of the testing.

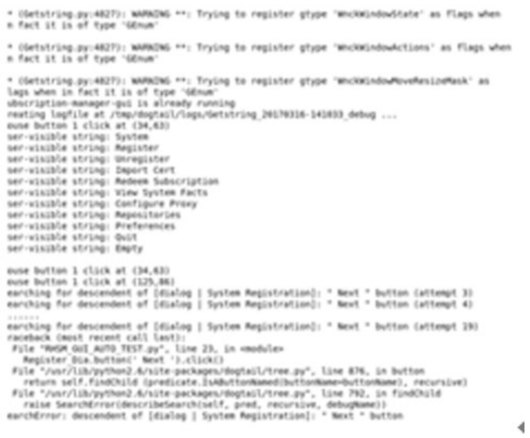


Status: 3 installed products do not have valid subscriptions.

Product	Version	Status	Start Date
Red Hat Enterprise Linux Workstation	6.9 Beta	Not Subscribed	
Red Hat Enterprise Linux Desktop	6.9 Beta	Not Subscribed	
Red Hat Enterprise Linux Server	6.8	Not Subscribed	

Status:	Not Subscribed
Product:	Red Hat Enterprise Linux Workstation
Arch:	x86_64
Subscriptions:	

Figure1. The dialog of tested software



```
* (Getstring.py:4827): WARNING **: Trying to register gtype 'WnckWindowState' as flags when
n fact it is of type 'GEnum'

* (Getstring.py:4827): WARNING **: Trying to register gtype 'WnckWindowActions' as flags when
n fact it is of type 'GEnum'

* (Getstring.py:4827): WARNING **: Trying to register gtype 'WnckWindowMoveResizeMask' as
lags when in fact it is of type 'GEnum'
ubscription-manager-gui is already running
reating logfile at /tmp/dogtail/logs/Getstring_20170316-141033_debug ...
ouse button 1 click at (34,63)
ser-visible string: System
ser-visible string: Register
ser-visible string: Unregister
ser-visible string: Import Cert
ser-visible string: Redeem Subscription
ser-visible string: View System Facts
ser-visible string: Configure Proxy
ser-visible string: Repositories
ser-visible string: Preferences
ser-visible string: Quit
ser-visible string: Empty

ouse button 1 click at (34,63)
ouse button 1 click at (125,86)
earching for descendent of [dialog | System Registration]: " Next " button (attempt 3)
earching for descendent of [dialog | System Registration]: " Next " button (attempt 4)
......
earching for descendent of [dialog | System Registration]: " Next " button (attempt 19)
raceback (most recent call last):
 File "RHSM_GUI_AUTO_TEST.py", line 23, in <module>
   Register_Dia.button(' Next ').click()
 File "/usr/lib/python2.6/site-packages/dogtail/tree.py", line 876, in button
   return self.findChild (predicate.IsAButtonNamed(buttonName=buttonName), recursive)
 File "/usr/lib/python2.6/site-packages/dogtail/tree.py", line 792, in findChild
   raise SearchError(describeSearch(self, pred, recursive, debugName))
earchError: descendent of [dialog | System Registration]: " Next " button
```

Figure2. the dialog of tested software

In the tested main dialog of Subscription Manager, the menu of 'System' is a widget. Its string name is 'System' and its attribute is menu. It also can generate other submenu. After defining the actions, such as click(),it will return coordinate. For example. (34,63),(125,86). If the string name of the UI' widget is not correct, after 20-time searching, it will report specific error of no matched widget. The feedback can help us to revise and check the information of widget.

4.3Testing Performance Analysis

By contrast with traditional manual UI testing, there is the advantages of the element tree of DogTail of UI automation testing:

(1)The method element tree of DogTail of UI automation testing shortens the time of identifying UI's widgets and reduce repetition errors comparing with traditional methods. The method also is suitable for another UI testing of Linux, which is more universal and efficient. As for fussy and repeated test case, it can make better use of resource and finish the testing accurately.

(2)Information of the node is different in international product such as string name of the dialog and UI information. Because they are in different language. For manual testing, tester need master many kinds of language. However, obviously, it is impossible. Therefore, it usually requires tester from different country in testing language to cooperate to finish testing. The method that is proposed by this paper will invoke target language vocabulary list to solve the problem. Target language vocabulary list is translation list from the source language to target language to identify UI elements. It can help drive tested software, which can solve the problem of cross language.

(3)The other feature of the method is clearly describing software structure. The elements of the tested software are distributing in the element tree in the form of node. The producing connection of each generation and attribute is obvious. It avoids the problems that complex connection of each widget leads to a testing mess. However, manual testing only tries to click each widget again and again. It is easy to test the same widget many times or miss the widget.

(4)The feedback of testing is timely and accurate. After executing the test scripts, it will show the correct or fault information of the testing automatically, which can be the reference to the testing results.

In summary, the method of element tree of DogTail automation testing framework tests the software accurately and efficiently. This plan is not only suitable for another application in Linux ,but also suitable for language translation testing. It can guarantee the accuracy and rationality of UI testing and improve the testing efficiency.

5 Conclusion

The advantage of the Linux operating system's UI testing is open source and uniformity. Linux in any version can implement automatic testing by using DogTail testing framework. When testing different desktop project, tester only installs different desktop support. International software is not only open source but also multilingual and multi-version. This paper proposes the method of element tree of DogTail testing framework to search, identify and trigger actions. It can not only describe the UI structure clearly and but also implement internationalization testing. The method guarantees the rationality of UI's widgets and avoids the lack testing and repetition testing according to the element tree, which makes the automation testing more logical. Simultaneously, the method also ensures the quality of the tested product and provides the reference for extra UI testing. How to handle the feedbacks of UI automation testing and automatically produce bugs is the next concern for the research.

Acknowledgements

The research work was supported by Shandong Internat ional Science and Technology Cooperation Project "Co llaborative research on methods and technologies of lar ge scale ontology modularization".

References

1. He Hao,Cheng Chunling,Zheng Zhengyu,Zheng Dengji. effective and common UI testing method based on Silk and XML [J].Journal of Computer Applications,2013,(01):258-261.
2. MEMON A M. Automatically repairing event sequence-based GUI test suites for regression testing [J]. ACM Transaction on Software Engineering and Methodology, 2001, 27(2): 144 - 155
3. Wu Hengshan,Wang Jinhong. Automation Testing Model of GUI Based on Effectiveness of Interface [J].Journal of HuazhongUniversity of Science and Technology(Natural Science version),2004,(12):34-36.
4. Wang Zhao , Bai Xiaoying, Dai Guilan.GUI Test Case automatically Producing Technology of Colored Petri Net Model [J].Journal of Tsinghua University (Science and Technology),2008,(04):600-603.
5. Lu Yongzhong,Yu Xinghua, Nie Songlin, Pei Xiaobing,Wang Chun. GUI Automation Testing System 's Development Based Event-flow Chart [J].Computer Engineering and Science,2008,(05):142-146.
6. Zheng Yaoming. Design and Development based on LDTP on function testing automatic script producing system of Linux GUI[D].Beijing Jiaotong University,2010.
7. Yuan Bing. Software Testing Methods and Technology 's Research on Linux Desktop System [D].Hunan University,2005.
8. Jiang Wei，Wang Houxiang, Automation Testing of GUI[J].Ship Electronic Engineering ,2004. 24(3):50-52
9. Ron Patton, Software Testing, Second Edition, Same Publishing, July, 2005.
10. Abt. Achieving the Full Potential of Software Test Automation. Logigear，2004(5), 26-29
11. Kanglin Li, Mengqi Wu. Effective GUI Test Automation Developing an Automated GUI Testing Tool[M].Sybex,2004
12. You Zeqing. Automation testing framework of GUI software's research and development[D].Southwest University,2012.
13. Sun Y, Jones E L. Specification-Driven Automated Testing. Proceeding of the 42nd Annual Southeast Regional Conference,2004
14. Li Feng, Sheng Zhuang.Action-Driven automation test framework for Graphical User face (GUI) software testing[C]. Autotestcon, 2007.IEEE 17-20, 2007:22-27
15. Jian Ping Zhao,Xiao Yang Liu,Hong Ming Xi,Li Ya Xu,Jian Hui Zhao,Huan Ming Liu. A Lightweight-Leveled Software Automated Test Framework[J]. Advanced Materials Research,2014,2817(834
16. Liu Sheng, Software Automation Testing Framework's Design and Practice Beijing: Posts & Telecom Press，2009
17. Bu Qianqian. Automation Testing Framework of GUI software 's research and application [D].University of Electronic Science and Technology of China,2010.
18. Zhang Qianqian. Automation Testing Tools based on Widget Identification of GUI 's Research and Development [D].Southeast University,2015.
19. Xi Haitao,Wang Fang. GUI and Related Technology of Linux Operating System's Discussion[J]. Heilongjiang Science and Technology Information,2013,(14):142.
20. Liu Shubin, He Yuzhu, Liu Jinkun. Universal Automation Testing Software Platform of Linux's design and development [J]. Electronic Measurement Technology,2009,(01):70-72+81.