Design and implementation of an automated testing tool for GUI based on SWTBot

Chun Shan[1], Zequn Yu[1], Yifei Jin[1], Rui Ma[1] & Jingfeng Xue[1], Yifei Jin[2]

[1]School of Software, Beijing Institute of Technology,Beijing, China

[2]Development center,Agricultural bank of China,Beijing, China

sherryshan@bit.edu.cn,1096452051@qq.com, yifeifox@outlook.com, mary@bit.edu.cn, xuejf@bit.edu.cn, yifeifox@outlook.com

Keywords: automated testing; Testing tool; SWTBot; GUI software

Abstract. The born of GUI is a milestone in the development of software. It is very popular and welcomed by the consumers because of its friendly user interface and easy straightforward operation. Meanwhile it brings new challenges for the automation software testing technology. This paper begin to discuss with a company's GUI software testing project, finally, designing and implementing a set of GUI software test automation tool which based on understanding of testing strategy and requirement as well as characteristics of the software which to be tested. This tool based on SWTBot，an open-source testing framework, conducting a customized development of under test software, which has the features of stable, effectiveness, easy to maintenance, at the same time, has a strong portability. This tool has been successfully applied to the testing project, operating over 1000 test cases. It is nearly replaced all the manual testing in the regression testing. The efficiency and maintainability have high reputation from the QA engineers who are participated.

Introduction

With the the software applied to all walks of life, the quality of software has become more and more important. As an important means to ensure software quality, software testing technology has been paid more and more attention. In many kinds of software, graphical user interface software is the most popular, referred to as GUI software. Before the extensive application of automatic test technology, GUI software is tested by manual testing. With the continuous expansion of the scale of software, the continuous integration of model, the increase of function, there are some drawbacks in the traditional manual GUI test:

First of all, for the typical GUI software, the testing is mostly accomplished by a single keystroke and data entry. The repetitive and mechanical work can easily make tester tired and affect the accuracy and efficiency of software testing.

Secondly, some software defects can only be found through rapid and repetitive pressure test, it's mentioned in brackets [1].

In addition, now software updates frequently, software development and testing are often carried out at the same time. Testers must find the hidden defects as early as possible, then correct them timely in the new version. This requires the guarantee of test efficiency, as mentioned in brackets [2].

Miu Liu [3] found that GUI software products often have complex function, numbers of modules, and the production cycle is often very short, the traditional manual testing is not competent to this heavy work, so more testers tend to use automated testing. Therefore, the development of automated test tool for GUI has been put on the agenda.

Correlational studies

Some enterprises, scientific research institutions and software technology communities have invested into the research of GUI automated test, and have developed different tools to adapt the different characteristics. At present, GUI automated testing tools can be divided into three types:

(1) recording and playback type

This type generally uses the method of direct recording to realize. It is convenient but poor in stability and compatibility. It's found that it is difficult to find hidden defects because of the lack of verification of test results in brackets [4]. Chengchuan Yang and Li Yao [5] said that such testing tools are generally used for the regression testing of known defects.

(2) automated test cases type

This type refers to change test cases into automated code. The test tool runs them and collects test results. It's mentioned in brackets [6] that this kind of automated testing tools is the most flexible and stable, not only can find more software defects, but also performs better in coordination with test plan. Automated testing tools of this type are used widely which was said in brackets [7].

(3) automatic test type

This type refers to create test cases automatically and execute test cases automatically. Intelligence is the biggest advantage of this type. Edward Kit [8] said that this type has high investment in the early and the technology is difficult, its effect is restricted by its intelligent degree. Sun Y and Jones E L [9] said that the research of this type is frontier domains of test technology, the available technology can't be used in automated testing products.

An automated testing tool based on SWTBot

A company has developed a comprehensive data management tool based on Eclipse. This new product updates frequently, poses a big pressure on test. This tool is designed to solve this problem.

Test tool based on SWTBot framework combines the inherent advantages of SWTBot. It's layered by structure, simple to maintain, easy to expand. The encapsulation and extension of its function makes it easy to understand and use. Simple maintenance will be able to adapt to the change of product.

A. Overall design

In the three layer of GUI automated testing tool, some of the basic operations are placed in the bottom. For example, control atomic operation, general test tools and some common controls and so on.

The second layer packages all the components of the software under testing. These components include all the window objects in software. The window objects are generally the combination of basic control such as menu, dialog box, list and tree structure. In addition, component layer also contains classes and methods for obtaining configuration file.

The top layer of the architecture is the user case layer, it is the container of test case scripts. Developers of test cases achieve the simulation of manual operation through calling the method and interface first and second layer provides. To sum up, the system structure of the GUI automation testing tool is shown in Figure 1.

B. Design of instrumentation layer

1) Function of instrumentation layer

Instrumentation layer based on SWTBot atomic components provide simulation operation support to Eclipse products. This layer not only adapts to a particular software, but also adapts to all Eclipse products. It's a general layer. It should be good in portability of the code to reduce associated workload when change software. This layer is mainly composed of these function in Figure 2.

Eclipse general module: encapsulates generic functions provided by Eclipse, such as dialog box confirmation and cancel, project import and so on.

Controls atomic operation module: encapsulates the atomic operation of every control, such as editing text box, getting table contents.

General tools module: encapsulates methods which are often used in testing process, such as changing character format and screenshot function etc..

Log module: output and storage log.

In addition, the SWTBot framework is based on the JUnit, so the instrumentation layer should communicate with JUnit, such as driving test cases, relevant information's input and output.

2) Design of instrumentation layer

The structure of instrumentation layer is shown in Figure 3.

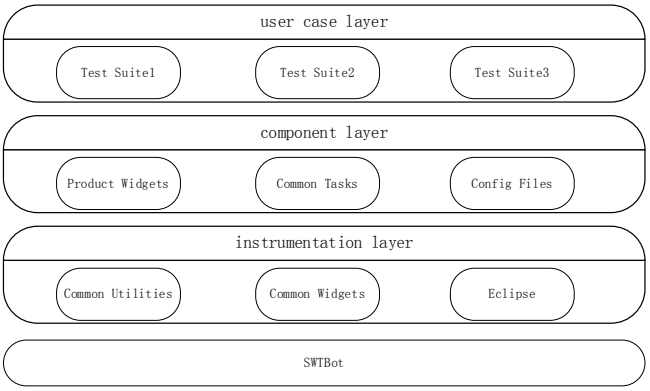

Figure 1. System structure of the GUI automation testing tool.

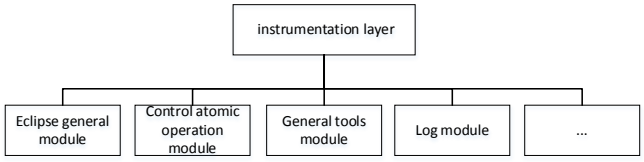

Figure 2. Instrumentation layer function model.

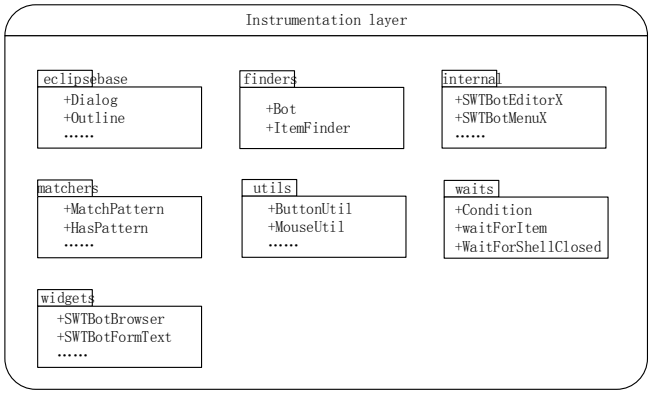

Figure 3. The structure of Instrumentation layer.

Eclipsebase package: the package contains Dialog, View, Welcome, Workbench, Outline, Project classes, these classes encapsulate generic components of Eclipse and prepare corresponding methods in the light of the operation of these components.

Finders package: the package contains the Bot class and ItemFinder class, mainly realizes the extension of the SWTBot framework.

Internal package: the package contains SWTBotEditorX, SWTBotMenuX, SWTBotTextX, SWTBotEclipseEditorX, SWTbotLinkX, SWTBotTableX, ContextMenuFinderX classes. These classes encapsulate some atomic operations of SWTBot, including the text box editor, right-click menu etc..

Matchers package: the package contains MatchPattern, HasPattern, WithPatternedText, WithPatternedLabel, WithPatternedTooltip classes, these classes encapsulate matching method used in the preparation of test cases.

Utils package: the package contains ButtonUtil, MouseUtil, MenuUtil, EditorUtil, WaitUtil, TableUtil classes, provides a simulation method for atomic operation of various control objects.

Waits package: the package contains Condition, WaitForItem, WaitForShellClosed classes. It Inherits and extends SWTBot's Condition class. WaitForItem and WaitForShellClosed provide some waiting methods for software to be measured.

Widgets package: the package contains SWTBotBrowser, SWTBotFormText classes, mainly encapsulates some common control.

C. *Design of component layer*

1) *Function of component layer*

Component layer is located on the Instrumentation layer. It uses the automated test tools based on SWTBot to realize customized development according to product to be measured. So the component layer is mainly composed of test objects in the product to be measured and some common tasks. The classes in this layer are connected with the product component, package all the window objects in the product to be measured and methods of objects.

2) *Design of component layer*

In component layer, the same type window objects are placed in the same package. This layer encapsulates window objects and realizes the operation of window controls. The basic architecture of component layer is shown in figure 4.

Among them, the utils package contains some tools related with software to be measured. The software to be tested is used to manage database. Many operations are associated with the database. For example, after the software connecting to the database, it can perform database backup operation, database parameters configuration operation and so on, which generates BackupDatabase class and Configure class in utils package. At the same time, the window components of the software to be tested are also associated with database objects. For example, users can view all tables in the database, the table structure and other information in the window, which generates TableOLE class in objectList package and TableProperties classes in properties package. Other function structure is similar to the above-mentioned function.

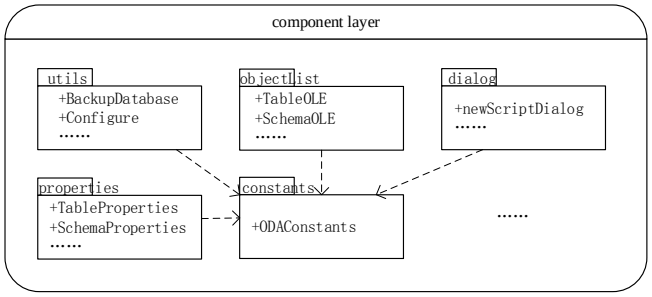

Figure 4. The basic architecture of component layer.

D. *Design of user case layer*

1) *Function of user case layer*

This layer except for hosting and executing the test scripts, but also initializes the test environment, including the start of software to be tested, configuration of parameters used by all test cases and the preset and cleaning of environment according to the different test cases. User case layer should also clean the environment after one test case has been executed, in order to execute other test cases.

2) *Design of user case layer*

User case layer is designed according to the function module of software to be tested. The test cases in the same module are placed in a package. The basic architecture of user case layer is shown in Figure 5.

We take test cases in Table as an example. The test suite is named TableTestSuite. Table I is a list of test case classes in TableTestSuite. All test cases associated with Table are placed in this test suite. The class which realizes test cases is named after test case name. So there are the implementation classes in the table such as CreateTable class and DropTable class.

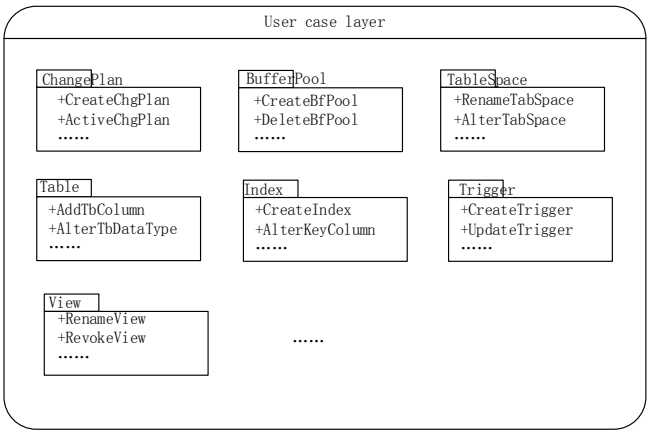

Figure 5. The basic architecture of user case layer.

TABLE I. THE TEST CASE CLASSES IN TABLETESTSUITE

Test case classes	Description
AlterPrimaryKeyTableColumn.class	change primary key
AlterTableColumnDataType.class	change data type of column
AlterTableCreateColumn.class	Add column
AlterTableDropColumn.class	Drop column
AlterTableGroupPrivileges.class	Change group privileges
AlterTableRenameTableWithTrigger.class	Rename table with trigger
AlterTableUserPrivileges.class	Change user privileges
CreateTable.class	Create table
DropTable.class	Drop table

Application and analysis of test tools

The test case number in test case library is up to 1325, 1026 of them are consistent with the demand. Regression testing result is shown in Table II. 958 test cases were executed successfully, 35 cases failed and 33 cases were wrong. The passing rate is as high as 93%. Software defects or environmental factors may cause failure. Interface changes when software update may cause error, if test tools are not adjusted. So these cases need manual test to verify.

TABLE II. STATISTICS OF TEST RESULTS

Test suite	Test cases	Pass	Fail	Error	Passing rate
TestSuite1	347	315	22	10	91%
TestSuite2	425	399	10	16	94%
TestSuite3	168	166	0	2	99%
TestSuite4	86	78	3	5	91%
sum	1026	958	35	33	93%

This automated testing tool is developed for the software to be tested, it is used to replace the traditional manual testing. It has properties like low development cost, low maintenance cost, high resource utilization rate and strong stability.

Conclusion

With the wide application of GUI software, GUI software testing becomes more and more important. The quality of test will directly affects the quality of the software. The introduction of GUI automatic testing tool makes automated testing apply to GUI level. At present, most of the open source GUI automated test frameworks have just started, there are many problems need to be solved. In addition they also need the inspection of practical software testing project.

References

[1] Z.Z. Gan, Implementation of a automatic testing management tool. Computer & Information Technology, (11), pp. 78-79, 2008.

[2] M.W. Xu and T. Wang, The best practice of DB2 database management, Publishing House of Electronics Industry: Beijing, pp. 167-169, 2011.

[3] M. Liu, K.J. Miao and Z. Liu, GUI automatic test framework based on distributed system. Computer Simulation, (2), pp. 34-46, 2007.

[4] A. Memon, M. Pollack and M. Soffa, Hierarchical GUI test case generation using automated planning. IEEE Transactions on Software Engineering , 27(2), pp. 144–155, 2001.

[5] C.C. Yang and L. Yao, Research and improvement of automation test framework based on GUI. Computer And Modernization, (11), pp. 38-41, 2009.

[6] Marathon Integrated Testing Environment, http://marathontesting.com/

[7] Abt, Achieving the Full Potential of Software Test Automation. Logigear, (5), pp. 26-29, 2010.

[8] K. Edward, Software Testing In The Real World: Improving the Process, China Machine Press: Beijing, pp. 5, 2006.

[9] Y. Sun and E.L. Jones, Specification-driven automated testing of GUI-based Java programs. Proc. of the 42nd annual Southeast regional conference, pp. 140-145, 2013.