IOP Conf. Series: Materials Science and Engineering 466 (2018) 012052 doi:10.1088/1757-899X/466/1/012052

Research on Non Intrusive Methods for Dynamic Monitoring of Software

Wei Zhang[1 a], Bowen Yuan[1], Zhenyu Ma[1 2] and Qingling Lu[2]

[1] Department of Information and Communication, Army Academy of Armored Forces, Beijing 100072, China
[2] Department of Equipment Support and Remanufacturing, Army Academy of Armored Forces, Beijing 100072, China
[a]zw2051@126.com

Abstract. After the software is released, its users are mainly operation staff. They can discover the problems of the software. But the software failure can not be identified and reproduced. In the process of software operation, Software data is of great significance for upgrading and maintaining software in the later stage. In order to record the running process of the software accurately, this paper proposes a method of software operation monitoring. It monitors user's operation and program status through monitoring program and records preservation. The method records information about user operation and state parameter changes. These data can achieve error recurrence for the corresponding errors, test cases can be generated. The method is of great significance. The experiment proved that the method is effective and reliable.

1. Introduction

Due to functional increase, performance tuning, error repair and other reasons, software often presents dynamic evolution. The existing testing technology is difficult to meet the changing test requirements in the process of software evolution. The analysis and research of software application data can make up for the defects of the existing technology. Under the precondition of obtaining the user's consent, The software monitoring tool is used to monitor the running state of the software and record the corresponding data. In the late stage of software maintenance and upgrading, the data can be used as an important data support for error reproduction and test case generation. Therefore, the monitoring and analysis of application data in the process of software operation is of great practical significance.

2. The Framework of Monitoring Software Operation

2.1. The Design of the Overall Framework

The system function block diagram designed in this article is shown in Figure 1. The ellipse at the bottom of the block diagram represents the software to be tested, and the rectangular boxes at the left and right sides represent the two major system functional modules, and the dotted lines at the top represent the test results. The function module of the system is divided into user operation monitoring module and software state monitoring module. Users operate on the monitored software mainly through mouse and keyboard commands. The user operation monitoring module can record user actions by intercepting these commands. The software state monitoring module monitors the state of software to be tested. The monitoring record is recorded together with the bug report of the software fault.

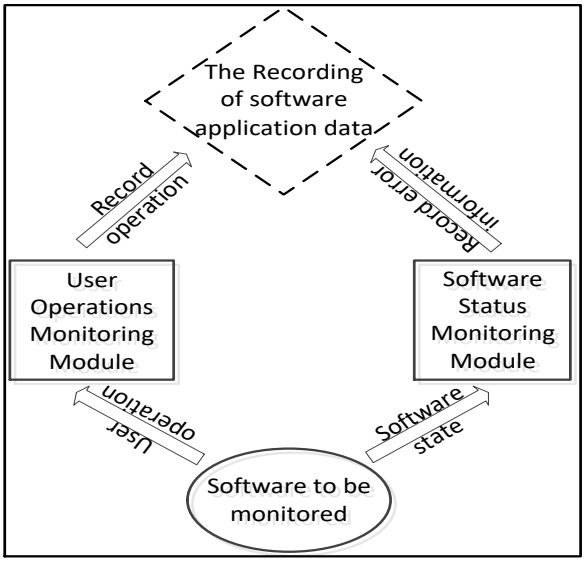

Figure 1. System function block diagram

2.2. User Operation Monitoring Module

The main function of this module is to record the user's operation on the monitored software. Software testers often use a large number of test cases to test the software in order to achieve higher test coverage, so the design of test cases is the key point to improve the efficiency of testing.The module uses the call interface left by the operating system to the application to monitor the user's action information on the system. The APIs that this module needs to call are KeyboardEvent and MouseEvent. The block diagram of this module is shown in Figure 2. When the monitoring module starts to work, the API interface is first monitored. If a API event occurs, check the keyboard or mouse event . When the API event is determined as a keyboard or mouse event, and the event object is the monitored software, the module records the event information.

KeyboardEvent triggers the recording of keyboard input events. The information that needs to be recorded is:

1)Event name(KeyboardEvent);
2)Event time;
3)Process id;
4)Window name ;
5)Input box name ;
6)The character of the input ;

MouseEvent triggers the recording of mouse events. The information that needs to be recorded is :

1)Event name(MouseEvent) ;
2)Event time ;
3)Process id ;
4)Window name ;
5)Clicked button name ;

Recording user actions in accordance with the above way, the implementation of user operation will become very simple.

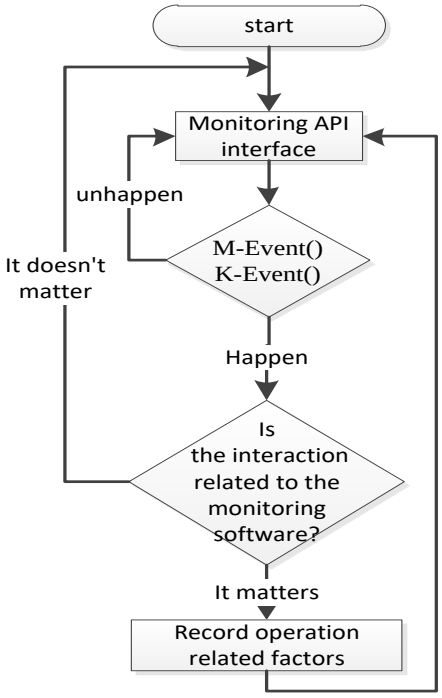

Figure 2. Block diagram of user operation monitoring module

2.3. Software Status Monitoring Module

The main function of this module is to record the status of the software being monitored. This module is based on the windows system function of the "system to check the problem report". When the software running process is abnormally interrupted or unresponsive, the windows system will automatically record a series of parameters in the software running error process and upload it to the background server for analysis. The module first monitors the parameters of the monitored software system. When the system parameters change, the Windows system error reporting function is used to verify and obtain the software error report.

2.4. Data Record Implementation

When the user runs the target program, the monitoring program starts to work and extract the software application data. The flow chart of the monitoring program is shown in Figure 3. The concrete steps are as follows:

Step 1: The monitoring software starts running. First create a log file

Step 2: Create two threads using the CreateThread() function, where the parent thread monitors the status of the monitored software (including CPU, memory, disk, network, etc.) through the interface provided by Windows. The child thread calls the mouse provided by Windows, and the API application interface such as keyboard acquires the user's operation on the system.

Step 3: The parent thread monitors the software parameters. When the parameter jumps, the module accesses the Windows error report check again. When the verification software fails, the program error report is recorded. (The wrong information can be obtained from the ExceptionInformation member of the EXCEPTION_RECORD structure.) The child thread intercepts all of the user's external operation information.

Step 4: Repeat step 3 until the monitoring program is turned off.

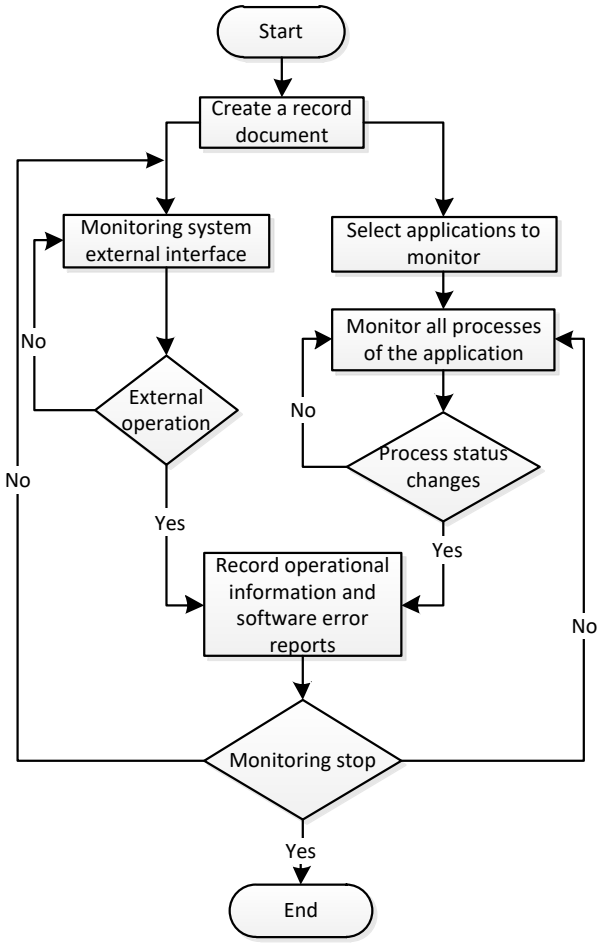

Figure 3. Flow chart of monitoring program

3. Case Analysis

In this paper, a method of non intrusive software operation monitoring is proposed. It mainly uses monitoring tools to monitor user operation and software status. It is an important support for later software upgrading and maintenance. In this paper, we design a test software for calculating the quadratic equation with one unknown. We enter the quadratic function parameters in the software window: a,b,c; Click the "calculation" button, and the software automatically calculates the two roots of the quadratic equation. An error is added to the original correct program code. When the software runs, it goes into the dead cycle. The results are shown in Figure 4, Figure 5.

As shown in Figure 5, the user enters the box "a", "b", "c", "5", "3", "5" in the window "my window", and then click the "calculation" button. According to the quadratic equation with one unknown is the root judgment formula:

$$\Delta = b^2 - 4ac = -91 < 0$$

At this point, the software should be prompted: the equation has no solution. The software is mistaken because of the error in the software. The software error information is shown in Figure 6. According to the recording information, when the user calculates the solution of the equation $5x^2 + 3x + 5 = 0$, the program stops responding, and is closed by the user, and gives a brief error description of the error: there is a problem that causes the program to stop interacting with the Windows.

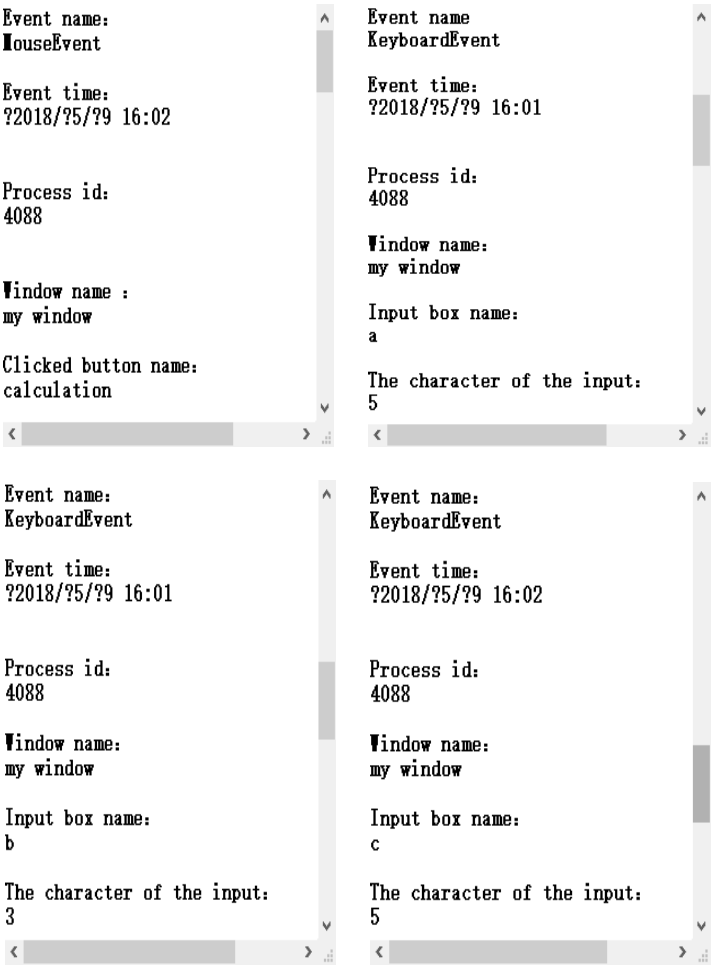

Figure 4. User operation record

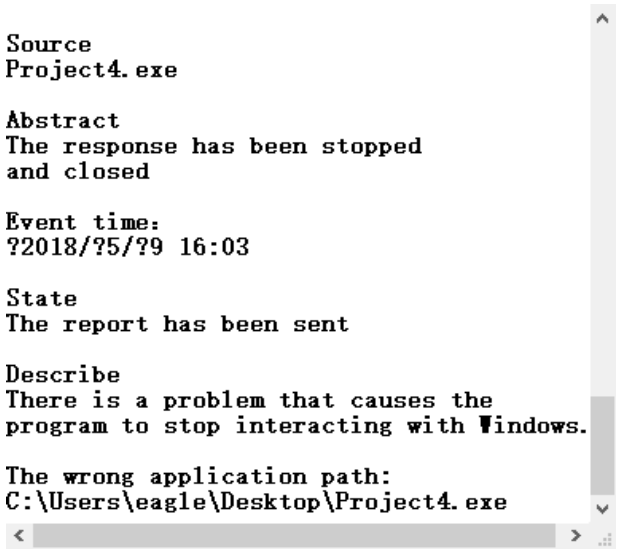

Figure 5. Software error report

When the software tester gets the record, the problem of software will soon be found in the experimental data combined with the software running.

4. Conclusion

In this paper, a method of non intrusive software operation monitoring is proposed. It monitors user operation and program status through monitoring programs, and generates software running status records combined with windows error reporting. It is proved by examples that when the program fails, it is reasonable to borrow the Windows system error reporting mechanism to reliably record the running state of the software. It has a great effect on software fault diagnosis.

At the same time, the data report in the process of software operation can be used for fault identification, and it can also be used for specific error analysis and pre location and software reliability analysis, and there are many places to be studied. The software monitoring will be further developed and more program running data can be obtained. Software testing will be carried out more deeply.

5. References

[1] Zhitie Zhang, Zhenyu Chen, Baowen Xu, Rui Yang. Research progress on test case evolution [J]. *Journal of Software*, 24(04): 663-674(2013).

[2] Jinhui Shan, Kejun Xu, Ji Wang. A kind of software faults diagnosing framework [J]. *Chinese Journal of Computers*, 34(02):371-382(2011).

[3] Lu Han. Research and Implementation of Test Case Generation Technology based on Web Users' Behavior [D]. *Zhengzhou University*, (2016).

[4] Yuan Liu, Yonghui Yang, Chunrui Zhang, Wei Wang. A Novel Method for Fuzzing Test Cases Generating Based on Genetic Algorithm [J]. *Acta Electronica Sinica*, 45(03):552-556, (2017).

[5] Juan Zhang.Research on test cast reuse in software testing [D].*Shanghai University*, (2012).

[6] Maha Kooli, Firas Kaddachi, Giorgio Di Natale, Alberto Bosio, Pascal Benoit, Lionel Torres. Computing Reliability: On the Differences between Software Testing and Software Fault Injection Techniques [J]. Microprocessors and Microsystems, (2017).

[7] Bo Yang, Ji Wu, Luo Xu, Kao Bi, Chao Liu. An approach of modeling software testing requirements and generating test case [J]. *Chinese Journal of Computers*, 37(03):522-538, (2014).

[8] Baowen Xu, Changhai Nie, Liang Shi, Huowang Chen. A Software Failure Debugging Method Based on Combinatorial Design Approach for Testing [J]. *Chinese Journal of Computers* (01):132-138, (2006).

[9] Takuya Naoe, Taketoshi Mizobe, Kohichi Yokoyama. Case studies of defect localization based on software-based fault diagnosis in comparison with PEMS/OBIRCH analysis [J]. *Microelectronics Reliability*, 54(6-7), (2014).