Research on Testing Method of Heterogeneous Software Based on CDD Model

Yanyuan Cao

College of Computer and Information Science, Southwest University, Chongqing, China

cookie5@swu.edu.cn

Keywords: heterogeneous software; test model; CCD model; test method.

Abstract: The complicated heterogeneous software that is featured by systems such as: E-commerce, E-government affairs, enterprise informatization, has developed at a rapid rate. In order to guarantee the quality of heterogeneous software, it is an urgent issue to research the testing method that is suited for its features. The software test based on model can generate test case according to software behavior model and structure model, effectively improving test efficiency. This paper, based on the analysis of the feature of heterogeneous software, proposes test model of heterogeneous software—CCD (Component Collaboration Diagram) model, by constructing Components Specification Framework, and CCD to model the interactive relations between heterogeneous components.

1. Introduction

With the development and dissemination of IT technology, the current software is no longer confined to the desktop system of one single machine, the complicated heterogeneous software that is featured by systems such as: E-commerce, E-government affairs, enterprise informatization have become a major trend of software development. The multi-pattern feature, embodied in the hardware and software platform of those software, leads to a large amount of speciality in developing and testing those software, and the traditional software design and testing methods have obviously been out of date for the demand of heterogeneous software. It is an important and urgent issue to research the testing technology and method that are fit for the features of large-size complicated heterogeneous software.

Software test based on model is a software testing technology that was attached great importance to in the academic and industrial circles, with this feature that it is based on the software model and derived model (generally it is called testing model)when producing testing case and evaluating the testing result. Based on the test task, it describes software function, behavior and structure in a understandable and abstract way, then an accurately described model for the tested software system is to be gained, the model can be used for software test [1]. This paper mainly discusses testing method according to heterogeneous software model, proposes a CCD model that is appropriate to heterogeneous software testing. Based on testing model, we propose the way to generate heterogeneous software test case and the testing strategy.

The structure of this paper is as follows: the second part introduces the feature of heterogeneous software and its testing problem; the third part proposes heterogeneous software testing model CCD against the testing problem and challenge of heterogeneous software, and lists a testing method of heterogeneous software; the fourth part lists the experimental data from applied testing model and method; finally we come to a conclusion and look forward to the future.

2. Feature of heterogeneous software and its testing problem

Heterogeneous software is a general term of large-size complicated software system that exists in heterogeneous software/hardware environment during the process of software system exploration or operation. Heterogeneous software is represented by complicated, distributed enterprise applied system and governmental information system, those systems are usually systematically collected by

  DOI: 10.25236/systca.18.001

multi-platform and multi developers. Some of the software has large scaled, adopt many heterogeneous sub-system and software to be a complete system. The software development in the heterogeneous environment basically shake off the pattern of "starting from scratch", instead, it heavily rely on components, it is a software exploration method that is based on software design and exploration, assembling and building large-size complicated software system.

Based on the above mentioned features, the difference between heterogeneous software testing and common software testing, apart from testing all components' accurate internal function by traditional testing method, it needs to spare more energy to check the mutual relations of heterogeneous components, and overall performance evaluation of heterogeneous software, etc. Currently the the testing technology was developed at a slow rate, more often than not, people adopted traditional software testing method, and no breakthrough was made. Thus, it is of vital importance to research the technology and method of integrated heterogeneous software system tests. The testing problems of heterogeneous software system include:

2.1 The exploration problem related to applied environment of software components.

TestS ∈ C-adequate-on-P1, But not necessarily TestS∈ C-adequate-on-P2. Among which, TestS stands for testing suites, C stands for testing standard of evaluate testing sufficiency, P1, the initial applied environment, P2, applied environment after component.

Heterogeneous software is assembled by different component, when developing components, component developers lack the applied environment information in which the component is to be used, they need to predict and imagine a applied environment to design and develop, the problem related to this applied environment is that, the imagined applied environment will not necessarily the real environment the components will be used in. The relative development of applied environment of components, rely on the imagined applied environment to test the component, once the applied environment changes, component needs to be retested.

2.2 Component users rely on the component developer.

To the users of component, invisibility of source code brought much trouble for test designers and test case generation. Component users find the problem arising from the flaws of components, then need to rely on the relative files, testing plan, source code to separate or get rid of its flaws, so components users usually rely on maintenance and support of the developers. It is advised that there is transferring contract and protection guarantee between developers and users, in order to prevent the developers refusing to supply maintenance and support.

2.3 Insufficient files.

For component users, they lack of detailed files when developing heterogeneous system. The files information offered by component may not conform to the expected grammar and semantics, or inefficient. The obvious problem brought about by inefficiency files is that the users must retest the component on a unit scale.

In addition, heterogeneity (mixture) in the components system is not conductive for the realization of unity and automation of test standard. The fast upgrading and uncertainty of component edition require frequent regression test for components system [2]. Lack of necessary information exchange between developers and users are the main reason that a series of problem arise.

3. Heterogeneous software testing model and method

The testing problem existing in heterogeneous software brings challenge for large-size complicated heterogeneous software system assembled by components. This paper, guided by model testing method, put forward the testing model that is appropriate to the feature of heterogeneous software, then propose a testing method of heterogeneous software, guide the whole testing circle of heterogeneous software.

3.1 Test model.

Heterogeneous software is formed by assembling all types of components, the test mainly focus on the check of the mutual relations between all heterogeneous components. Thus the test mode of heterogeneous software mainly model based on the mutual relations of heterogeneous components. The main elements that related to interactive performance include:

Interface: encapsulate one or more methods of function, describe behavior and responsibility under particular environment, disguise the implementation details, use "I" to stand for it.

Event: calling for one interface will generate a interface event, additional events include abnormally dealt event and user behavior event, use "E" to stand for it.

Based on this, we give the definition of heterogeneous software testing model.

Definition1: Component Cooperation Diagram(CCD) , component cooperation diagram G=（V, E）,V stands for vertex, E stands for edge of components interface and events. V=VI∪VE,VI stands for component interface, VE stands for component event. A route or sub-route in CCD is a exact limited series $(v_0,v_1,\ldots,v_k)$, among which $e_{i,\,i+1}=(v_i,v_{i+1})\in E$， i=0,1,…,k-1. A simple route is route $(v_0,v_1,\ldots,v_k)$,for any i and j, when $i\neq j$, $v_i\neq v_j$.

In CCD, the interactive relations between interface and event can be directly generated, the relations of controlling and data dependence among components can be defined in CCD:

Definition 2: Controlling Dependence, if there exist a simple route $v_0,v_1,\ldots,v_k$, among which $v_i \in V$, i=1,2,…,k-1. Edge=$(v_i, v_{i+1})\in E$, then we can define there is controlling dependence between two interfaces or between interface and event.

Definition 3: Data Dependence, ①If I_1 call I_1', I_2 call I_2', and I_1' rely on I_2', then there is data dependence relations between I_1 and I_2. ② If E is controllingly relied on I_1, and I rely on I_1, or I_1 rely on I, then there is data dependence relations between I and E. ③IF E_1 controllingly rely I_1, E_2 controllingly rely I_2 , and I_1 was relied on I_2 , then there is direct data dependence relations between E_1 and E_2.

3.2 Testing method.

Base on the testing model, we provide a way of testing heterogeneous software.

Phase 1: preparation before test

Preparation includes planning for the target and foundation of the test, analysis of test demand, define the object and range of the test, make a plan for the test, evaluate the work and resource, schedule,risk and test strategy.

Phase 2: execute the test

Executing the test include adopting appropriate method and technology to design test case, develop test tool or script, execute test case, etc. Hereby we mainly research the process of generating testing case through testing method.

Due to the feature of heterogeneous software, the test mainly includes unit test of component itself and the collective test of interactive relations of heterogeneous components. Component producer test all the function of components in a independent method, this is unit test of heterogeneous system. Component users should consider components function and interactive relations related with applied environment, this is a collective test of heterogeneous system.

As for the unit test of component, article [3] provided several testing methods in which component producer test component and provide the summary information for component user, include procedure partitioned method, control dependence analysis and data flow test method, etc. In addition, as for the component users usually can't get detailed and correct information related to test from the producer, (such as source code or other technological information), self-testing method can be adopted, component itself increase analysis and test functional specification, by executing some or all the activities in the process of testing by components users, this enhanced components can test the internal methods and behaviors. Automatically generate technical information needed by test when operating components, and be used as encapsulating internally. The self-testing method can be found in article [1].

We mainly research collective testing method of heterogeneous software.

Generate Component Specification Framework. According to the testing specification introduction and file information gained from unit test of component, generate the information of component interface and interaction, then we can determine component assembling and its physical dependence relation, generate Component Specification Framework.

Build CCD model. After getting physical dependence from component specification and introduction, further make sure the interactive mutual relation between components. According to the analysis model and document come from the analysis and design phase, analyze mutual interactive dependence between components, including direct dependence and indirect dependence, direct dependence include the interaction of components interfaces and events, indirect dependence include controlling dependence and data dependence between them. Build CCD model for the interactive relations between components, supply model support for testing heterogeneous software.

Traverse test model, generate test case according to the following test coverage rules: Coverage rule 1: In CCD, each interface should be insured to be tested at least once, each interface calling event should also be tested at least once, other event which is not related with interface calling, but has effect on it ,and should also be tested.

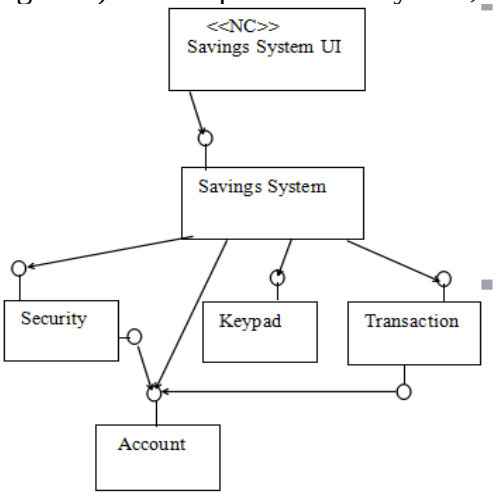
Coverage rule 2: For controlling dependence, when two interfaces or one interface and an event has controlling dependence, this controlling dependence may be provided by different scene, in CCD, there may be more than one simple route existing in interface and event, one way is to only require covering at least one simple route, another way, require covering all the simple routes. Theoretically, it requires covering all the simple routes, but actually, for the large amount of test case in test, generally require at least one simple route.

Coverage rule 3: For data dependence, due to the unknown source code, we can only use an approximately supposed method, a complete hypothesis presuming each interface pair has data dependence, in addition, simply suppose there is no data dependence between each interface. For the test of data dependence, we should combine Components Cooperation Diagram, other analysis and designed files, as well as other detailed functional description.

Phrase 3: evaluation of testing result

According to the end of the test standard given by test plan, compare the relation between testing result and expectation result in the process of test execution, make evaluation to the whole test, make a test conclusion and analysis report. Test evaluation sometimes include the evaluation to the effectiveness of test process, use the tool to check the coverage ratio of the test case in the execution process, to get effectiveness of test case in the whole test, to compare the relation between the number of mistakes injected in during the test process and the number of mistakes after the test, evaluate the effectiveness of different graded test.

We building a Component Specification Framework (figure 1) and Component Collaboration Diagram (Figure 2) of a simplified bank system, demonstrate the application of test model.



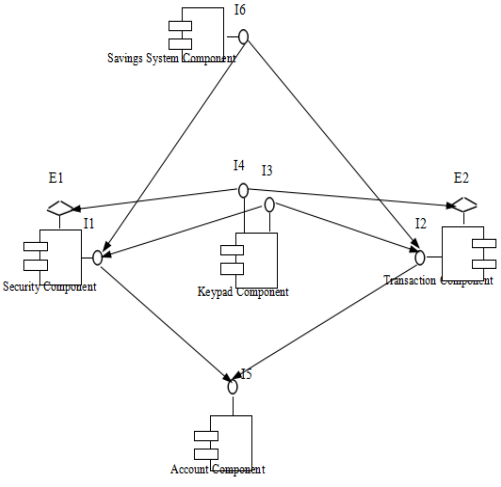


Figure 1 Component Specification framework Figure 2 Component Collaboration Diagram

In CCD, the following test case cover all the interfaces and events

Case1=<I_4, E_1>

Case2=<I_4, E_2>

Case3=<I_5, I_1, I_3>

Case4=<I_6, I_2 >

Case1, Case2, Case3, Case4 cover all the interfaces and events of the five components in Figure 2,but can not cover the controlling dependence relation between V_5 and V_6,we add two test case to covering the controlling dependence,

Case5=<I_6, I_1, I_5>

Case6=<I_6, I_2, I_5 >

4. Experimentation data

In order to verify the validity and availability of the testing method, we made tests practice for two heterogeneous systems, system I is a sub-system of large-scale E-commerce management software, it has 19 components, some parts of which are third-party components, 1085 interfaces, 2368 events; system II is a governmental information system, which has 16 components,913 interfaces, 1967 events.

When the system releases pre-QA test, 76 errors were found, during maintenance after the system was published, 27 errors were found and corrected (the circumstance of system II was given in Table 1). After analyzing the errors report and system code, 103 errors are categorized into three types, which is listed in Table 1.

TABLE 1

	Type A	Type B	Type C	Total
System I	36	41	26	103
System II	26	30	18	74

It can be noticed that 35% falls into type A, 40% fall into type B, 25% fall into C.

In addition we make comparison experiment, adopt traditional method and the method mentioned in this paper to test two systems. As for system I, adopt traditional method to execute 1465 test cases, and find 77 errors in the system; adopt the method mentioned in this paper to execute 923 test case, and find 87 errors. See the result in Table2 (the circumstance of system IIwas given in Table 2).

TABLE 2.

		Test case	Type A	Type B	Type C	Total
System I	Error report		36	41	26	103
	traditional test	1465	29	31	17	77
	CCD test	923	27	39	21	87
systemII	Error report		26	30	18	74
	traditional test	1226	20	23	13	56
	CCD test	764	21	27	16	64

Experiment result shows, to test heterogeneous software system, traditional test will find 76% of the error; The CCD testing method proposed in this paper will find 84% of the errors, and find some newly discovered error which has never been noticed before in the testing process.

This CCD testing method proposed in this paper take much less expense than traditional test, and find more systematic error than traditional test, it exert play a some important role in testing heterogeneous software system.

5. Conclusion

This paper put forward the test model of heterogeneous software according to the feature of heterogeneous software, in consideration of collective test of heterogeneous software system, using

the Component Specification Framework and the CCD modeling the interaction relations between the components, supply support for testing mutual operational error of heterogeneous software. According to the CCD, we analysis direct and indirect dependency of the components, then propose the test method of heterogeneous software. Testing technology and method of heterogeneous software, naturally speaking, all draw lessons from the traditional test technology, according to the feature of components to make some improvement. In the future work, we will start from method research, combine the empirically based research and the development of tools, develop test technology and method, improve the theoretical foundation of heterogeneous software test, research the sufficiency evidence of the test, solve cross-stage, cross language test problem.

Acknowledgment

The work described in this paper was supported by "the Fundamental Research Funds for the Central Universities", XDJK2011C077.

References

[1] YAN Jiong, WANG Ji, CHEN Huo-Wang, "Survey of Model-Based Software Testing," computer science,Vol.31,No.2,2004.

[2] S.Beydeda,"Self-Metamorphic-Testing Components," Proceedings of the 30th Annual International Computer Software and Applications Conference, 0-7695-2655-1/06, 2006.

[3] M. J. Harrold, D. Liang, S. Sinha, "An Approach to Analyzing and Testing Component-Based System[C]", In: Proc First International ICSE Workshop, PP134~140, 1999.

[4] W. Chan, T. Chen, H. Lu, T. Tse, and S. S. Yau, "A metamorphic approach to integration testing of context-sensitive middleware-based applications," In International Conference on Quality Software (QSIC), PP241–249, 2005.

[5] J Hornstein, H Edler, "Test reuse in CBSE using built-in tests[C]," In: Proc of the Workshop on Component-Based Software Engineering, Composing Systems from Components. Los Alamitos, CA: IEEE Computer Society Press, PP11-14, 2002.

[6] Naslavsky L., Ziv H., Richardson D.J., "A model-based regression test selection technique," Software Maintenance, 2009. ICSM2009. IEEE International Conference on 20-26 Sept. 2009 PP515 - 518.

[7] Yang Liu, Yafen Li, Pu Wang, "Design and Implementation of Automatic Generation of Test Cases Based on Model Driven Architecture," Information Technology and Computer Science (ITCS), 2010 Second International Conference on 10.1109/ITCS.2010.90 PP344 - 347.