A study on the Methods of Software Testing based on the Design Models

Saka Uma Maheswara Rao[1], Praveena Iduri[2]
[1]Associate Professor, [2]Assistant Professor
[12]Department of Computer Science & Engineering,
[12]Swarnandhra College of Engineering & Technology, Narspur, W. G. Dt., A.P., India

Abstract- With the increase in growth of computer applications, they are used drastically in fundamentalfields, so it requires higher quality and reliability for software. Though the most important techniques are software quality and reliability, software testing becomes more and more essential in software development. However, software engineering technology development regularlyestablish new requirements for software test technique, software test model and test case generation are the vital components for software testing, and it is important to prefer the best test models and test methods to progress the efficiency of test cases. Considering that most of software errors could be featured to the difference between design and implementation, this paper demonstrates a software testing method based on the design model by examining the design model and implement model obtained from design process and transform it to formal test models, and finally consider the design model and the implement model to find the difference between requirement and implement.

Index Terms- design model, test model, software testing, test case

I. INTRODUCTION

With the development and accessibility to computertechnology, the use of software is near to all levels of society, and becomes the core technology backingfor the enterprises, industrial control, and communicationsindustry, and also performing an important role in the Militarydepartments, which makes people increase demand for thehigher and higher dependability of computer systems. How the software quality should get improved is one of the main issues to discuss forsoftware engineering. Software testing is the most fundamentaland important tool to provide accuracy of software and improvereliability of software, and prevailing technology used in theindustrial.Software test techniques and methods are changed in an artificial way bydesign models, development process, programming languagesand other software development technologies andmethodologies; therefore, there are not regular test methodsthat are appropriate to all the software and test requirements. Forthe past 20 years, the software engineering researches hasobtained many important progresses that are many newapplications, development models and developmentmethodologies, so the research on software testing will also becontinuously changing [1].

II. THE SIGNIFICANCE OF THE STUDY

Most of the software errors may be associated to thecontrast between the design and the implementation in thedevelopment. Traditional test methods areimperfectespecially in the test efficiency, thus people thought thatfinding a way to use existing resources for automatic software testmethods. This paper proposes to identify the differencebetween the design and the implement by comparing the designmodels and the implementation models, and finally aims todevelop the software test tool for automatic testing.

III. THE IDEAS AND THE METHODS OF SOFTWARE TESTING BASED ON THE DESIGN MODEL

The design model in this paper is referred to the UMLdesign model that is extensively used in industry now. This paper works with the precise method to analyse UML design models to getthe regular test models, and transforms theimplement modelsat the same time from the code, then change the implementmodels to the formal test models, and finally compares thetwo formal models to recognize the difference to generate testcases. We can also design the automatic test software by this type of method to increase the test efficiency.

A. The formal Analysis of the design models

The software models abstract and describe the software inthe software development so that helping people to understand,clarify and develop the software. In the software developmentprocess, there are differentkinds of software models in the variousdevelopment phases. Along with the continuation of developmentphase, software models are becomingfurther detailed until the code isgenerated. Software testing can also use the design modelswhich are generated in the design process, but in reality,different designers can get different design models. So, thedesign models are to beproperlycharacterized in order tomake standardization and reasoning comparison, which is a basis forall consecutive work. The example for the formal description ofcollaboration diagrams is as follows:

The collaboration diagram gives priority to the object'sorganization structure between the sending messages of objectsand the receiving messages of objects. The collaborationdiagram can be used to model a definite scene of the use caseaccording to the role and their relationship of the interaction.

It describes the particularbehaviour of the objects in the passivestructure and the active interaction that is a set of messages.

Definition: The UML collaboration Diagram CD can be expressed as a pair: CD = (OD, M).

OS = (OS_1, OS_2, ……. OSm) is the nonempty finite of object set in CD.

M = {m_1, m_2, m_3………m_n} is the finite of messages set described in CD. Each message is defined as:

M=(guard,sequence_expression,m_name, parameter list,□□*)return_value,receive,send, type, m_name* is the message name,*guard* is the guard condition of the sending message,*sequence_expression* is the order of entry list, *parameter_list*and *return_value,* are respectively message of the parametersand return values of the message, *receive* is the sending objectof the message, *send* is the receiving object of the message.

This type of M is known as:

type €{syncall *asyncall send return},syncall* represents thesynchronous call of the method, *asyncall represents* theasynchronous call of the method, *send* represents the sendingof signal*return* represents the return of synchronous call.

B. The Extraction and the analysis of the implementation models

The implement model cannot directly come into action, this will require the reversalof the code to generate. Even thoughthere are manyexplorations on the generation of implementation model fromreverse engineering that exploited the industrialproduction requirements [2-4]. The implementation modelsconverted from the code should be properly examined in orderto standardize. This formal description methods of theimplement models refer to formal description methodsof the design models.

C. The choice of the test models

The test model is a very important to the testing methodbased on design models. Design models and implementationmodels are compared by test models. When the different designmodels are used as the test models, the construction method ofdesign models, the extraction method of implementationmodels, the comparison method of test models, and the testcases generation method will be different. And the appropriatetest model language will benefit the entire test process toimprove the efficiency and reliability. The test model describedby the natural language cannot be reasoned and compared, therefore the description language of the test model must be theformal language.

The choice of the abstraction level of the test model isvital. If the abstraction level of the test model is upscale,then the information source of code will be reduced a lot more in theelicitation process of the implementation model so that it islikely to suffer the difference of the implementation model andthe design model. Otherwise, if the level abstraction ofthe test model is close to the ground, the elicitation process of theimplementation model is comparatively simple to achieve and will hold on more information. Nonetheless, testers have the only way to construct thetest model with the formal language by themselves from theexisting information. Even if testers had to go through morecomplicate and larger amount of work to complete theconstruction process, they should announce the error which isused for them to understand and nothing to do with the code inthe construction process. The implementation model that ischanged in the construction process does not actuallydemonstrate thecode, so the comparison result of the design model and theimplementation model is not significant. Therefore, the choiceof the test model should be based on detailed analysis of thetesters in the actual situation.

D. Generating Test cases by Comparing the Models

The model comparison must be depicted and figured out based on explicit language, so it must firstly definethe design models and the implementation models as reported bythe test models, and then will compare the design models andthe implementation models to get the difference between two models.

IV. SUMMARY

This paper presents the indication of future course of action and the unique implementation method on the software testing based on thedesign model. There are still some important complications for furtherresearch, such as the specialized selection rules of the test model,the transformation of formal models and so on. The followingpicture shows that the test cases are automatically extractedfrom the activity diagram of beverage vending machines by theModeltest tools that designed by the author.

V. REFERENCES

[1]. Lawrence P, Sergey B, Rajeev M, et al. The PageRank Citation Ranking: Bringing Order to the Web[Z]. Stanford, CA, USA:Stanford Digital Libraries, 1998.

[2]. Liao Husheng. Data Specialization and Program Specialization Based on Control Flow Graph. Chinese Journal of Computers.2001(9).

[3]. P.Runeson.A Survey of Unit Testing Practices.IEEE Software,2006,23(4):22-29.

[4]. Xie Xiaodong; Lu Yansheng; Mao Chengyin. Software Testing Method Based on Model Comparison. Journal of Southwest Jiaotong University.2008(2)

[5]. Meudec C.Automatic Generation of Software TestCases From Formal Specifications [D]. Belfast,UK:The Queen's University of Belfast,1998.

[6]. A.Polini A.Bertolino.A Framework for Component Deployment Testing.In:A.J.a.F.Titsworth Ed.Proc.of ICSE'03.Portland,Oregon USA.Los Alamitos,CA:IEEE Computer Society Press,Press,2003,221~231.

[7]. Fu Jianjing. Code Protection Based on Feature of JAVAC and JVM. Computer Engineering.2010(1).

Author Profile:

Saka Uma maheswara Rao
Associate Professor
Department of CSE, SWARNANDHRA College of Engineering & Technology,
NARSPUR, W. G. Dt., A.P., India

Praveena Iduri
Assistant Professor
Department of CSE, SWARNANDHRA College of Engineering & Technology,
NARSPUR, W. G. Dt., A.P., India