ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 2, No 5, May 2013

Cost Estimation Modal to find faulty Objects in Software Reusable Components

Ramadeep Kaur, Sami Anand
[1] *M.Tech Student, Department of Computer Science & Engineering,*
Lovely Professional University, Jalandhar (Punjab), India,

[2]*Assistant Professor, Department of Computer Science & Engineering,*
Lovely Professional University, Jalandhar (Punjab), India,

Abstract

The software development cost can be reduced by reusing the existing components. These exciting components can be the object oriented software components .The object oriented components can be easily reused. Reusing the exciting components will save time and cost .When new software is developed before starting development the cost estimation is done, if the overall cost of the new software is reduced by reusing the components then only excited components are reused otherwise not. In this paper, we will review the cost modal for reengineering object oriented software. Some times in the excited components the faulty objects exists. The faulty objects mean the objects which are not compatible with the new software and cause errors. In our work we focus on to identify these faulty objects in the exciting object oriented software components.

Keywords:

Reusing, existing components, object oriented, fault objects

1 Introduction

In previous times, we believe in original development. But with the advancement in the software engineering we believe that overall cost of the software will be reduced by reusing the exciting software components. The object oriented software components are easy to reuse. Object oriented software components are the independent components that can be plugged in the plugged out in the software system. The plugged in and plugged out of the software component will not effect the exciting software system. The object oriented software components contains the object methods and variable these methods and variables are globally defined to achieve the compatibility factor. The developers need to put extra efforts on the developing the software components which will be reused in future. Object oriented software components have higher level of abstraction. These higher level abstractions object oriented components can be easily plugged in and plugged out without effect the exciting software system. But sometimes, these higher level abstractions object oriented components contains the faulty objects. Faulty objects means the objects which are not compatible with software system and gave incorrect responses. To identify these faulty objects we need to perform reengineering, reengineering process is to time consuming and lots of efforts need to identify and modify these faulty objects. When the faulty objects are correctly identified we need to redesign the exciting objects oriented component to remove the faulty object. The large software systems are more complex to reduce the complexity of the large software systems we need to use the object oriented components when the faulty objects are there in the object oriented components, which are used in the large software systems it is very difficult to identify these faulty objects .So, we need such a approach which identify these faulty objects and our not much efforts are wasted to identify these faulty objects in the objects oriented software components.

Related Work is presented in section 2. We present need of reengineering and object oriented software in section 3.In section 4 reengineering cost modal of software System is presented. In section 5 we present the conclusion and Future work. In section 6 references were written.

2 Related Works

2.1 Dr.Varun Kumar, Mohit Kumar

They proposed a new approach which consider the severity of fault based on requirement prioritization .Main aim is to find the severe fault early in the testing process and hence to improve the quality of the software according to the customer point of view.

2.2 Dr.Arvinder kaur, Shivani Goyal

They had presented BCO algorithm for maximum fault coverage using two examples whose results are comparable to optimal solution .In this they using Average Percentage fault detection metrics and charts have been used to show the effectiveness of the proposed algorithms.

2.3 Phuc V.Nguyen, Ph.D candidate, David Eichmann

They had discussed about the direct relationship between the attributes and the quality factors. They classified attributes as external product attributes and internal product attributes .The external product attributes are those which influence the external behavior of the system had how system behaviors in the external environment. The internal product attributes are those which influence the size, cost and performance of the system.

2.4 Bakhshsish Singh Gill, Dr. Ashok Kumar

They had discussed the object oriented technology of software re-engineering and define objects, attributes, values.They had discussed that object oriented components are independent components which can be pulled in and pulled out according to the need in the software system.

2.5 Dr.Linda H.Rosenbar

In this paper they had discussed about the need of re-engineering and with the process we can reduce the software cost and time. To effectively implement the re-engineering we need certain set to parameter metrics .In the cost of the software decreases by using the re-engineering matrices then only the reengineering can by done otherwise we adopt the tradition mechanism of software development that is fresh development.

2.6 Searnendu Biswas and Rajib Mall

In this paper they had described the number of regression test selection techniques for different programming paradigms such as procedural, object-oriented, component-based, database, aspect and web applications. In this paper they review the important regression test selection techniques proposed for various categories of programs and identify the emerging trends.

2.7 Bhati Suri and Isha Mangal

They had proposed Hybrid technique based on BCO for analyzing text case selection. Their result shows that a huge amount of reduction in test suit takes place. Reduction in test suit reduces time as well as cost. They have proposed hybrid approach which proves much faster than ACO technique. The tool which they developed runs much faster to provide the minimum subset of test cases .The tool can provide different result in each group. This approach will reduce the time for testing the software and we can easily identify the faulty objects in the object oriented software system.

3 Need of reengineering and object oriented software

The object oriented software's in which the modules can work independently and each module can be pulled in and pulled out from the software system when required. Designing an object oriented software system is very hard but designing a reusable object oriented software system is even harder. When designing a reusable object oriented software components the design should be able to solve the future problems and must compatible with the most software systems. While designing a system software we follow the tradition approach of fresh development and if we want to reduce the cost and time of development, we use the object oriented reusable components. To estimate the cost of system software, certain set of metrics are used. While, devolping system software our main motive is to reduce the cost and time of devolvement, if overall cost of the system software is reduced by using object oriented software components then only reusable components are used otherwise not.

The object oriented components when pulled in the system software there will be a chance that they will gave incorrect responses to certain set of inputs. The incorrect responses can be due to the faulty objects that excites in the reusable components. To identify these faulty objects in the object oriented software components, we need reengineering approach. The reengineering approach is very time consuming and inefficient, it is very hard to identify these faulty in the reusable software components, in the complex software systems it even harder to identify these faults objects.

4 Reengineering Cost Modal of Software System

The object oriented software components are more reusable and reusability can reduce the cost and time of the software development. At the time of reusing the software components we need to perform reengineering to estimate the overall cost of the software system if it is less than the fresh development then only reusable components are used otherwise not .For the cost estimation we use the reengineering cost modal .The cost estimation can be done in the reengineering cost modal by considering two factors:

- Number of objects to be reused
- Number of attributes in the reusable objects (size of objects)

The object oriented components have the attributes according the requirement as specified in the previous software system .when these object are reused with the other system software there will be chance that these objects miss behaves with the new system software .The objects which miss behaves with the new system software are called faulty objects .To identify these faulty objects we need reengineering approach. The certain efforts are required for reengineering.Identifing the faulty objects we need to review the code of the object. The efforts required for reengineering are estimated of the bases of line of code that need to review, to identify the faulty objects. The common method for the effort estimation is:

Efforts=a*(Size)*b

Here a, b are the constants that are determined at the time of regression analyzing applied to historical data .The size is the number lines in the code.

When the code review is done the revere can identify the faulty objects from the set of objects .Among all the objects some objects works fine and some of them will not work fine and gave incorrect responses to the certain set of inputs .

The set of fault objects which are identified are:

O=O1+O2+O3………………..On. To identify these set of faulty objects certain efforts are required.

The set of objects which are not faulty are described as:

E=E1+E2+E3………………….En

Identifying these non-faulty objects will also require some efforts. So total efforts which are required to identify faulty and non-faulty objects are:

Total estimated efforts=E + O

The total reengineering cost required to identify faulty objects in terms of money is given as:

Estimated reengineering cost = Total Estimated Effort * current Rate of Person-Month

5 Conclusions and Future Work

In this paper, we review the reengineering cost modal and present the efforts required for reengineering to identify the faulty objects .In our works we also discuss about the reusable object oriented software components .In our future work, we will develop a software which will identify the faulty objects in the reusable object oriented software components and we try to minimum the efforts required for reengineering for object oriented software components.

5 References

Dr. Varun Kumar, Sujata, Mohit Kumar "Test Case prioritization using faulty Severity", IJCST International Journal of computer Science and Technology, Vol. 1, Issue 1, September 2010

Dr. Arvind Kaur, Shiavngi Goyal "A Bee Colony Algorithm for Fault Coverage based Regression Test Suit Prioritization" IJAST, Vol.29, April 2011

Phuc V, Nguyen, Ph.D candidate "The Study and approach of software reengineering." IJAST, Vol.29, 2010

Dr.Ashok Kumar, Bakhshsish Singh Gill "Cost Model for Reengineering an Object Oriented Software System" Global Journal of Computer Science and Technology Volume 11 Issue 20 Version 1.0 December 2011

DR.LINDA, H.Rosenbergy "Software Re-engineering"

Swarnendu Biswas and Rajib Mall,"Regression Test Selection Technique: A Survey, Informatics 35 (2011)

Bharti Suri and Isha Mangal," Analyzing Test Case Selection using Proposed Hybrid Technique based on BCO and Genetic Algorithm and a Comparison with ACO,"Volume 2,Issue,April 2012

Robert S. Arnold, "Software Reengineering", IEEE Computer Society Press Los Alamitos, California

Bernd Bruegge, Dutoit Allen H., "Object-Oriented Software Engineering Using UML, Patterns, and Java", Pearson Education (Singapore), p.724.

Brock R.W., Wilkerson B., Wiener L. (2007) "Designing Object-Oriented Software", Prentice-Hall of India, New Delhi p. 5

http://softwaredesign.com/objects.html