Exploring the Efficiency of the Program using OOAD Metrics

Mr. S. Pasupathy [1] and Dr. R. Bhavani [2]

[1] *Associate Professor, Dept. of CSE, FEAT, Annamalai University, Tamil Nadu, India.*

[2] *Professor, Dept. of CSE, FEAT, Annamalai University, Tamil Nadu, India.*

Abstract

In this today world, everything has being computerized to make and to carry out the task easily and quickly. In order to computerize the task, everything has being programmed. The program consists of a set of instructions to complete the task. These programs can be developed by using various languages. First, using the language, we have to develop software and with the help of the developed software, we have to develop a program to implement the task.

Of these, now-a-days, the object-oriented programming software is commonly used in order to develop a program with less effort. This object-oriented program can do the task in much simpler way. In this paper, we have to analysis the developed object-oriented software with proper measures and to develop a report based on the quality of the software. Also, we have to propose a methodology to verify whether the software has been error-free or not through compilation process. Based on the analysis report, the user has been reported with the efficiency of the program and they take proper steps to improve the quality of the software.

Keywords: Computerized, Efficiency, Error-free, Instruction, Object-Oriented, Program, Quality, Software, Task.

1. Introduction

In this computerized world, as our day-to-day life depends upon computer, all the things have been programmed. To program the tasks, we have to utilize the software. Software may be of different categories depending upon the functionality. Upon those categories, the most commonly used one is termed as **Object Oriented Software.** The main feature involved in this software is to split the program into number of sub-programs based on the functionality and finally, the programmer has to build these sub-programs into a program using some built-in functionality.

In spite of the category of software being used to develop the program, the main task of the programmer is to verify whether the developed program performs its task correctly. To do this, each software consists of some compilers and interpreters. With the help of these compilers and interpreters, the program has to verify line by line and can able to identify the error(s) occurred in the program. The error may be of various kinds such as syntax error, type mismatch error and so on. Only when the program becomes error-free, the program is identified as a successful program and any user can use the program efficiently. But, if the program is not error-free, then the program has been completed partially and the program has been used less.

To do this process, we have to use the new emerging technique called OOAD (Object Oriented Analysis and Design). OOAD is a software engineering practice that manages large projects professionally. It creates models by defining objects and their rules of interaction. With the help of this technique, we have to analysis the software by means of the rules and to identify the quality of the software. This Object-Oriented technique is a method to design and build large programs with a long lifetime.

Today, when it comes to analysis and design of software systems, the first thing that comes to mind is OOAD.

1. Why is OOAD so popular?

There were many Systems Analysis and Design methodologies that came in. We had Structured Systems Analysis and Design (SSAD) wherein DFDs and ER diagrams were basically a part of it. After that OOAD came into picture. And since then whenever we think about Analysis and Design of systems, OOAD is the first thing that comes to mind. There are many frameworks, patterns and languages that are following the OOAD concepts. One of the reasons why OOAD as a methodology has succeeded is because OOAD allows us to model the real world in its most natural form.

Upon developing the program based on Object-Oriented language, the next step is to process the program using the OOAD technique, to verify for the program efficiency. Using the rules defined in the OOAD technique, the program is verified line by line and the error is identified. Based on the error identified, the error rate has been calculated as defined in previous research. In that paper, the quality of the software can be measured using the identified error rate. In this paper, based on the compilation error, the program efficiency can be determined and using this determination result, the program is analyzed for its utilization. To achieve this, the proposed methodology of this paper provides suitable algorithm and solution.

2. Previous Research

In paper [1], Sujata et al investigated several object oriented metrics proposed by various researchers. These object oriented metrics are than applied to several C sharp programs. A critical analysis of the results was presented listing the crucial points about the language so that software managers and developers can use these results for building object oriented system in C#.

The design and development of software using object oriented paradigm is gaining popularity day by day. Object Oriented Analysis and Design of software provide many benefits to both the program designer and the user. Object Orientation contributes to the solution of many problems associated with the development and quality of software product. This technology promises greater programmer productivity, better quality of software and lesser maintenance cost [2].

The Object Oriented technology forced the growth of object oriented metrics. Although many metrics have been proposed, few have been based on the sound measurement theory or, further have been empirically validated. One of the first attempts to do this was by Chidamber and Kemerer (C&K). They have proposed six new OO metrics based on theoretical concepts [3].

In paper [4], Bandar Alshammari et al presented a hierarchical model for assessing an object-oriented program's security. Security was quantified using structural properties of the program code to identify the ways in which 'classified' data values may be transferred between objects. The model was validated via an experiment involving five open source Java programs; using a static analysis tool they had developed to automatically extract the security metrics from compiled Java byte code.

In paper[5], Edgar Gabriel et al pointed that a large number of MPI (Multiple Programming Interface-like Multitasking) implementations are currently available, each of which emphasize different aspects of high-performance computing or are intended to solve a specific research problem. It also presented a high-level overview the goals, design, and implementation of Open MPI.

In paper [6], Robyn stated that Open source software systems were becoming increasingly important these days. Many companies are investing in open source projects and lots of them were also using such software in their own work. This was also introduced the fact extraction process to show what logic drives the various tools of the Columbus framework and what steps need to be taken to obtain the desired facts.

In paper [7], Deepak et al described that Software metrics are required to measure quality in terms of software performance and reliability related characteristics like dependencies, coupling and cohesion etc. It provides a way to measure the progress of code during development and having direct relationship with cost and time incurred in the software design and development at their later stages. These major issues must be checked and informed early in the development stage, so that reliability of any software product could be ensured for any large and complex software project. Object oriented software metrics directly focuses on the issues like complexity, reliability and robustness of the software developed using object oriented design methodologies. It reflects the time, cost and effort that would be incurred in development at later stage. While the software in its development stage, it was desirable that the complexity levels at every stage should be minimized to make the end product more reliable and manageable. Object oriented metrics provides all parameters through which one can estimate the complexities and quality related issues of any software at their early stages of development. In the paper, authors have studied three object oriented metrics namely MOOD Metrics, CK Metrics, and QMOOD Metrics and given a case study to show, how these metrics are useful in determining the quality of any software designed by using object oriented paradigm.

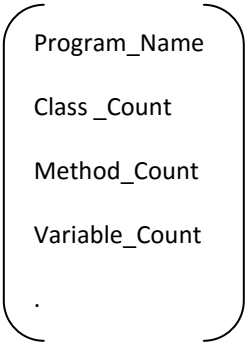
In paper [8], Henderson described that Object oriented approach was capable of classifying the problem in terms of objects and provide many paybacks like reliability, reusability, decomposition of problem into easily understood object and aiding of future modifications.

In paper [9][10][11], Briand et al stated that Object-Oriented Metrics are useless if they are not mapped to software quality parameters. Many number of quality models are proposed to map parameters of the Object Oriented software like Extensibility, Reusability, efforts, manageability and cost. To know more about the internal structure of the product one should know more about the interdependencies of parameters of metrics and Software quality parameters.

3. Proposed Work

The aim of the research paper is to analysis the software to determine the program efficiency. Based on the analysis report, the quality of the software can be measured and can be improved as per user wish.

The summary of the proposed methodology is described in this section. From the previous research work, we have to identify certain techniques to measure the quality of the software. Based on that, we had to define a methodology in order to split the program into number of classes and then from the sub-classes, the functions can be identified. Thus, we have to get the resultant of number of classes and functions involved in the given program. From the program, we can also get the quality of the software through compilation. While compiling the program, the programmer has to identify the number of errors occurred in the program. From the error identified, the error rate can be determined and from this resultant rate, the quality of the software can be easily measured. Thus, from the given inputted program, the details such as number of classes, functions, identifiers, errors can be determined. From these details, the error rate can be calculated.



Now, we can improve this, by successively compiling the program. The first step of the proposed work is to analysis the error rate. Upon determining the error rate, the quality of the program can be determined. The next process is to compiling the program successively and then determining the error rate. The compilation result is stored sequentially with appropriate versions such as 1.1 for the first compilation of the first program, and 1.2 for the second compilation of the first program and so on. This compilation process is iterated until, the program gets error free. When the program gets error-free, based on the compilation versions stored, we have to make the comparison to determine the program efficiency. The compilation result is saved as follows:

This result can be saved with the <Version> name as Program_Name.Version_Number. Based on the number of versions saved for the same program, the quality of the program can be determined. When the error rate becomes null, then the program is said to Error_Free program and then a comparison is made based on the stored resultant. This comparison is made to determine how much time and effort can be utilized by the program to make the program error free. From the comparison chart, the program efficiency can be determined.

<Compilation Result>

<Version>Program_Name.Version </Version>

<Program_Analyzation>

<Program_Type>….</Program_Type>

<Class_Count>….</Class_Count>

<Method_Count>…</Method_Count>

</Program_Analyzation>

<Error_Report>

<Error_Count>….</Error_Count>

<Error_Type>…</Error_Type>

<Error_Rate>…</Error_Rate>

This proposed methodology also consists of an algorithm to determine the quality of the program. The algorithm is given below with proper explanation.

3.1 Algorithm

Start

Create the Configuration file with OOAD Metrics

Accept the Inputted Program

Analysis the Program using the Configuration file

Get the results of the program

Identify the error rate

Count=0

Repeat

Compile the Program using Compilation Procedure defined in the Configuration file

Identify the error

Save the compilation result with appropriate version

Increment the count

Until error="NULL"

For i=0 to count-1

Compare the error(i) with error(i+1)

Result=error(i+1) – error(i)

Next

Analysis the program efficiency based on the value on 'Result'

End

3.1 Algorithm Explanation

The algorithm defined above will process as follows: the initial step is to develop a configuration file with OOAD metrics. Based on the OOAD metrics defined on the Configuration file, the program undergoes for analysation. The efficiency of the program can be determined by compiling the program successively. The compilation process is iterated until there is no error detected while compiling. All the compilation results are saved with appropriate program version. After the program becomes error-free, the result stored can be compared to determine the program efficiency. Thus the efficiency of the program can be determined using the proposed algorithm.

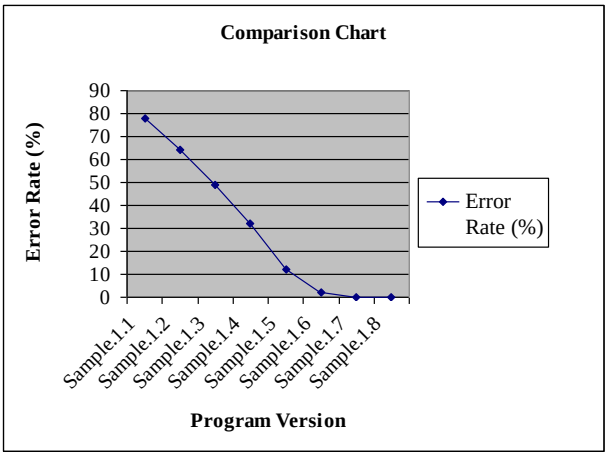
4. Experimental Results

The proposed methodology can be experimentally verified to determine the program efficiency. To experimentally verify the proposed algorithm, we have to undertake the program developed in the IT Company, with the domain Java named as 'Sample'. The program has been examined in two different ways. First, the program is verified manually and the efficiency is to be determined. Second, the program is verified through OOAD metrics defined in the configuration file. The program has been compiled successively to get the error-free program. The resultant is tabulated as follows:

Table-1: Compilation Result

Program Version	Error Rate (%)
Sample.1.1	78
Sample.1.2	64
Sample.1.3	49
Sample.1.4	32
Sample.1.5	12
Sample.1.6	2
Sample.1.7	0.01
Sample.1.8	0

From the resultant, while compiling the program for 8th time, the program will become error-free. Then the result has been compared to determine the program efficiency.



From this comparison chart, the efficiency of the program is determined. The curve in this chart decreases continuously and thus the efficiency of the program is "Good".

5. Conclusion

The methodology proposed in this paper satisfies the aim of the research. That is, the proposed methodology performs well to determine the program efficiency. The determination can be carried out by developing a configuration file with necessary OOAD metrics which will be useful to measure the quality of the software. Thus the proposed methodology performs well to determine the efficiency of the program and based on the result, the quality of the software can be improved.

References

[1] Arti Chhikara, R.S. Chhillar, Sujatha Khatri, "Applying Object Oriented Metrics to C# (C Sharp) Programs", Int. J. Comp. Tech. Appl., Vol 2(3), 666-674.
[2] Patrick Naughton & Herbert Schildt."C# 3.0: The complete reference", McGraw-Hill Professional, UK, 2008.
[3] Chidamber, S., Darcy, D., Kemerer, C." Managerial use of Metrics for Object Oriented Software": an Exploratory Analysis, IEEE Transaction on Software Engineering, vol. 24, no. 8, pp. 629-639,1998.
[4] Bandar Alshammari, Colin Fidge and Diane Corney, "A Hierarchical Security Assessment Model for Object-Oriented Programs", Faculty of Science and Technology, Queensland University of Technology, Australia, 11th International Conference on Software Quality,pp. 218-227, May 2011.
[5] Edgar Gabriel, Graham E. Fagg, George Bosilca, Thara Angskun, Jack J. Dongarra, Jeffrey M. Squyres, Vishal Sahay, Prabhanjan Kambadur, Brian Barrett, Andrew Lumsdaine, Ralph H. Castain, David J. Daniel, Richard L. Graham, Timothy S. Woodall Innovative Computing Laboratory, University of Tennessee, Open System Laboratory, Indiana University, "Open MPI: Goals, Concept, and Design of a Next Generation MPI Implementation". Sep 2004.
[6] Robyn R. Lutz, Jet Propulsion Laboratory, "Software Engineering for Safety: A Roadmap",pp.213-226 June 2000.
[7] Deepak Arora, Pooja Khanna and Alpika Tripathi, Shipra Sharma and Sanchika Shukla, Faculty of Engineering,Department of Computer Science, Amity University, "Software Quality Estimation through Object Oriented Design Metrics".
[8] B.Henderson-sellers, "Object-Oriented Metrics: Measures of Complexity" Prentice Hall, 1996.
[9] L.C.Briand, J.Wuest, J.Daly and Porter V., "Exploring the Relationships Between Design Measures and Software Quality In Object Oriented Systems", Journal of Systems and Software, 51, 2000.
[10] L.C. Briand, W.L. Melo and J.Wust, " Assessing the Applicability of Fault Proneness Models Across Object Oriented Software Projects", IEEE transactions on Software Engineering. Vol. 28, No. 7, 2002.
[11] P.Coad and E.Yourdon, "Object Oriented Analysis", Yourdon Press, 1990.

BIOGRAPHY

Dr. R. Bhavani received her B.E., and M.E., degrees in Computer Science and Engineering from the Regional Engineering College, Trichy in the year 1989 and 1992, respectively. She received her Ph.D., degree in Computer Science and Engineering from Annamalai University, Tamil Nadu, in 2007. From 1990 to 1994 she worked as a Lecturer in the Department of Computer Science and Engineering, at Mookambigai college of Engineering, Keeranur. Currently she is the Professor of the Department of Computer Science and Engineering at Annamalai University, Tamil Nadu, India. Her current research interests include image processing, image segmentation, image compression, steganography, pattern recognition, medical imaging, content based image retrieval and software metrics. She serves as a reviewer for scientific articles in many international journals and scientific conferences. She has published more than 50 scientific articles in many international journals and scientific conferences.

Mr. S.Pasupathy received the B. E degree in Computer Science and Engineering from Government College Of Technology,Coimbatore in April -1990. He received the M.E degree in Computer Science and Engineering from Kongu Engineering College in the year Dec-2000. He worked at Kongu Engineering College,Perundhurai,Erode from 1990 to 2001. He has been with Annamalai University, since 2001. He is pursuing his Ph.D in Computer Science and Engineering at Annamalai University. He published 4 papers in international conferences and journals. His research interest includes Object Oriented Analysis and Design and Software Engineering.