A Method to Analyze & Evaluate the Reusability of Object Oriented Code

Mrs. Mala Dutta¹ Mr. Keshav Purswani²
¹,²Department of Computer Science & Engineering
¹,²IET, DAVV Indore

Abstract—Today everyone wants to increase productivity and maintainability of the application. Reusability is best way to increase the productivity and maintainability of the application. Software components made for one software application can be used for other software application having similar requirements. By reusing these components in other software one can reduce the cost of Today everyone wants to increase productivity and maintainability of the application. Reusability is best way to increase the productivity and maintainability of the application. Software components made for one software application can be used for other software application having similar requirements. By reusing these components in other software one can reduce the cost of developing and also reduce the time of developing an application. But the question is how to decide which software component is reusable and which one is not? The aim of this paper is to propose a way from which one can identify reusable software components.

Key words: Reusability, CK Metrics, Complexity, Maintainability, Understandability, Adaptability

I. INTRODUCTION

Reusability is the concept came in early days of the software developments. Software developers always use coding, functions and procedures. Reusability not only saves cost but it also saves lots of time of developing an application. It also saves the efforts that have already been made to a component. Many examples has been seen in which the organization saves lots of money by increasing the level of reuse by as little as 1%. The software developers not only emphasizing on reusing the existing modules but they are developing the new modules in a way that they can be reused easily because if they have spend lots of time and efforts on the development of a module then it should be reusable in similar application. That will save time as well as the effort and saves lots of money. Software performance also increased. So software can be of good quality and low cost. This is why reusability is very important concept and measurement makes it more valuable.

Some researchers have been working from early days of programming to make reuse speedy, easier and more systematic. This leads to the development of the object oriented programming language. Concept of inheritance is introduced in object oriented programming language which is purely based on reusability concept. In inheritance we use the properties of a class into another class. For example if you want to want to make a class for checking your balance in bank then you can inherit those characteristics from saving account class. One should discover system which existing component or module is further appropriate for use again, and try to reuse.

II. REUSABILITY

With the help of the reusability we can easily identify the reusable modules and encourages making modules that can be reused easily. Reusability attracts many software professionals today because it provides the benefit of low cost and quick development of module. Many organizations now using the concept of reusability and offering the low cost softwares to their customers plus quick deployment of the softwares. Reusability helps a lot to the organizations to increase the productivity and reduce the development time. Department of US has saved about 300 $ million by increasing the reusability by 1%. This example shows importance of reusability. This is the reason why this concept is used worldwide today but this is not the only example there are many more examples where reusability helps a lot:

1) Software reuse concept is used in Missile Systems Division which increased the 50% productivity.
2) American Navy decides to use reusable modules which reduces around 30% men power to manufacture and maintenance Restructured Naval Tactical Data Systems (RNTDS).
3) Magnavox tried to use the reusable modules to develop the Force Fusion System Prototype (FFSP); and as a result 20% development time for developing new system is reduced. In SDLC this concept is not only restricted to coding stage but there are some phases where this concept is used:
   - Code
   - Requirement
   - Architecture/design documentation
   - Test plans
   - Specifications
   - Design
   - Manuals
   - Templates
   - Design decisions

III. PREVIOUS WORK

1) In 1976 McCabe proposed a theory based on cyclomatic complexity. According to this theory the complexity increases if logical branches increases(decisions). McCabe presents a graph theory and says if you represent a module using a flow chart and count the branch area(do-while, if-then-else), more count means more complexity of the program.
2) In 1977 Halstead proposed a theory regarding the complexity of the program. According to his theory devide the program or problem into tokens and tokens further devided into operators(verbs,
functions etc) and operands (nouns, variables and files). Based on the equations these tokens give the complexity of the program in terms of estimated effort, program volume and size.

3) In 1987 Prieto Diaz and Freeman proposed a theory in which they categorize five attributes of the program and related metrics for calculating reusability. It helps to decide which module is reusable and which one is not. This theory helps programmers a lot to modify the module with ease. These five attributes are:
   - Size of the program.
   - Structure of the program.
   - Program documentation.
   - Programming language chosen for coding.
   - Reuse experience from previous modules.

4) In 1989 Selby gives a theory on Quantitative Studies of Software Reuse. Selby tried to find the instances in which reuse is successful and also tried to find the reason of the success. In his study he gives a statical study with the help of Database of NASA software environment.

5) In 1991 Caldiera and Basili gives a theory on “Identifying and Qualifying reusable software components” which states that reusability heavily depends on some basic properties like correctness, believability, testability, ease of modification and performance. But it does not give any way to measure these attributes.

6) In 1993 Chen and Lee has built around 130 reusable components in C++ and used these components in an experiment and shows how these affects software productivity and quality.

7) In 1991 Prieto Diaz gives a theory on “Understanding and Qualifying reusable software components” which states that reusability has substantial effect on software quality. According to the study software quality and reuse of the software complement each other. Software quality increases with the software reuse. But software quality will increase if there is a way to measure it. In this paper, they have given a good method to calculate the software reusability of OO class diagram.

8) In 2008, Pradeep Kumar Bhatia and Rajbeer Mann proposed an approach for measurement of reusability of Object Oriented Design. Software reusability has substantial effect on software quality. According to the study software quality and reuse of the software complement each other. Software quality increases with the software reuse. But software quality will increase if there is a way to measure it. In this phase we measure the reusability of the source code. Finally reusability of software code is calculated in numbers. And also the final result of weather the source code is reusable or not.

IV. APPROACH FOR IDENTIFICATION OF REUSABLE MODULE

A. Extract The Source Code:
In this stage we analyzed the source code and take out valuable information and save it in memory, which is essential for measuring the all metrics, these metrics are essential for evaluate features on which reusability depend.

B. Calculating the Metrics:
In this stage we measure four metrics:
   - DIT (for Inheritance):
   - The maximum length from the node to the root of the tree.
   - CBO (for Coupling): Number of immediate subclasses subordinated to a class in the class hierarchy.
   - NOC (for number of Children): CBO is defined as the count of the number of other classes to which it is coupled

Information gathers from extract phase will help us for implementing these metrics. And result, based on formulae, of the all metrics is saved in memory. All metrics are concern with object oriented system.

C. Display:
In this phase we measure the reusability of the source code. Finally reusability of software code is calculated in numbers. And also the final result of weather the source code is reusable or not.

Fig. 1: Steps to Calculate Reusability of Object Oriented Code.

V. CONCLUSION
The paper aims to provide a way to measure reusability of object oriented programs. Reusability is most important quality attribute and it is of the required attribute in object oriented software development as it increases the developer productivity, minimized development cost as well as reduces time to deployment. The work presented in this paper can be effectively used to calculate the reusability of any object oriented software module.

REFERENCES


[13] Richard W. Selby. “Quantitative studies of software reuse”. In Ted J. Biggersta and Alan J. Perlis, editors,
