Volume 3, Issue 12, December 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

An Analysis of Software Quality under the Object Oriented Metrics

Suparna Ahuja
Setup Configuration Specialist
AON Hewitt, Gurgaon, Haryana, India

Abstract— ***Today most of the available systems are designed under some language framework. Object Orientation is one of the most adaptive frameworks used by most of software applications. The transparency provided by this language framework is the true evaluation of the software system by using different metrics provided by the framework itself. When an object oriented system is designed it is designed under same quality vectors defined under the same framework. Some of the design quality parameters of such system are the cohesion, couple, module interactivity etc. In this paper, an exploration to such metrics based quality vectors is defined.***

Keywords— Quality Vectors, Metric Based, Object Orientation

I. Introduction

Software Measurement is the key aspect of software development that is integrated with each stage of software development to estimate the software quality, to build software plan and to estimate the software cost. In most of the organization, software measurement is used at the planning phase or at the post development phase to estimate cost and quality of the software. Software measurement is actually a statistical approach that uses the structural information of software development. Based on the kind of information used for the software analysis, the measurement matrices are divided in two categories called direct measure and indirect measure. Direct measure includes the process and product based matrices to estimate the cost, efforts, execution speed etc. The indirect metrics are the measures to estimate the efficiency, reliability complexity of the software product [1][2]. The classification of software measurement process is shown in Fig 1.

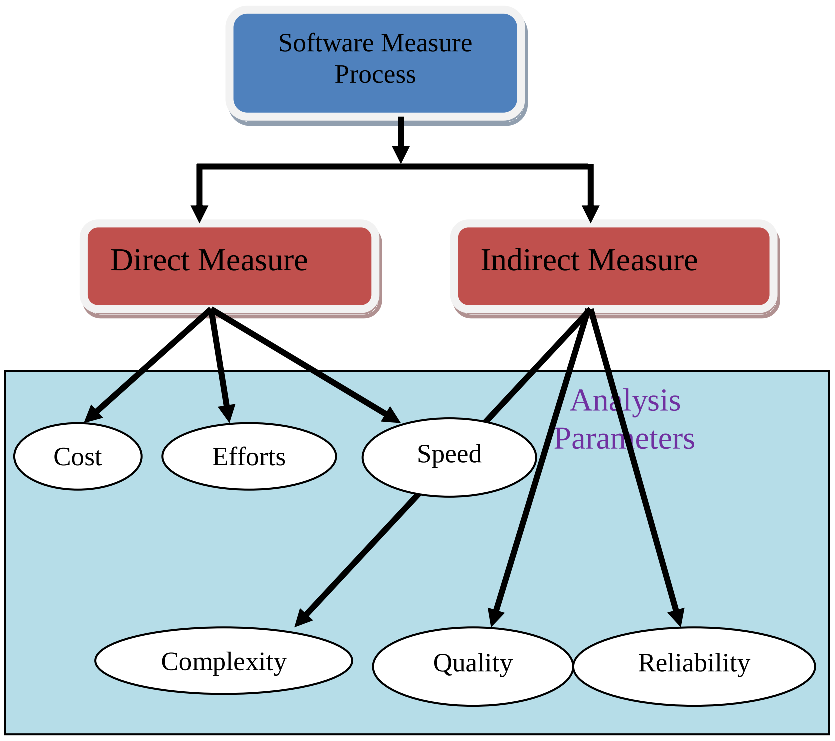

Fig 1: Software Measurement Process

Software measurement process must be accurate and quantitative so that effective results will be drawn. The effectiveness of the process is based on the assessment, adjustment, refinement etc. The most considerable aspect of accurate assessment is the data collection. Accuracy of data collection includes the concerns, issues and quality of the measurement process. There are different approaches adopted by measurement process to collect the data and to represent the results [3][4]. The key components of effective measurement process are shown in Fig 2.

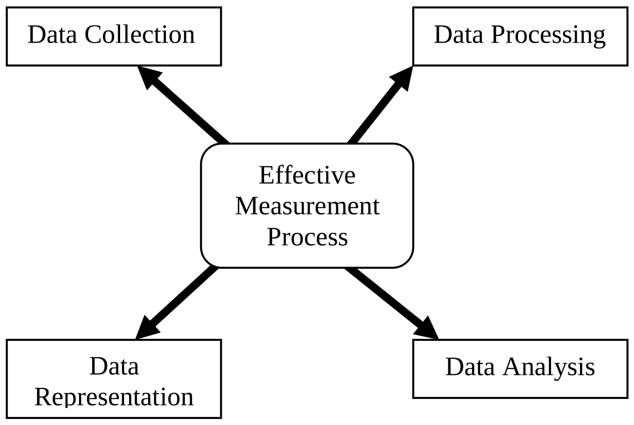

Fig 2: Components of Measurement Process

A. Software Metrics

Software metrics are defined as a set of formulas or the techniques to measure the software system or the product. Software metrics are capable to measure the software product, processes and the quality of the software system. It can also identify the success or the failure of a software product or the process. It is defined as a process or the measure to estimate the software reliability. The block diagram of software metric analysis is shown in Fig 3.

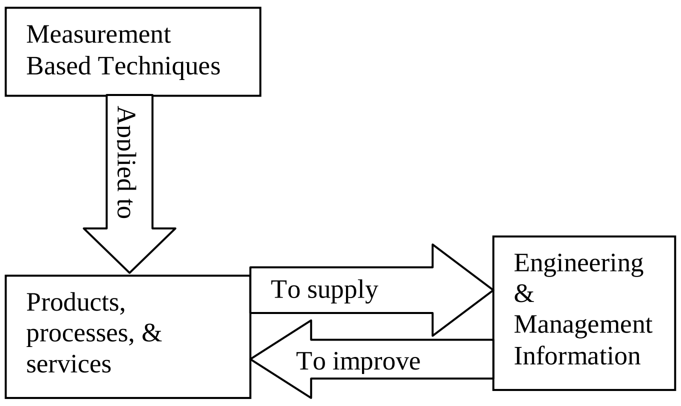
Measurement Based Techniques
Applied to
Products, processes, & services
To supply
To improve
Engineering & Management Information

Fig 3: Software Metrics

Software metrics are applied to determine the software cost, analyse the software requirement, estimate the software reliability and identify the software fault. These metrics are code based metrics to analyse the software code and the product information [5][6].

In this paper, the analysis of the software system is performed to estimate the software reliability using software metrics. In this section, basic review of software measurement process and the software metrics is defined. In section II, different kind of software metrics and its classification is discussed. In section III, object oriented metrics are discussed as the metric suit to estimate the software system. In section IV, conclusion of this research work is defined.

II. CLASSIFICATION OF SOFTWARE METRICS

There are number of standards to estimate the software products under different frameworks and paradigms to perform the software evaluation.

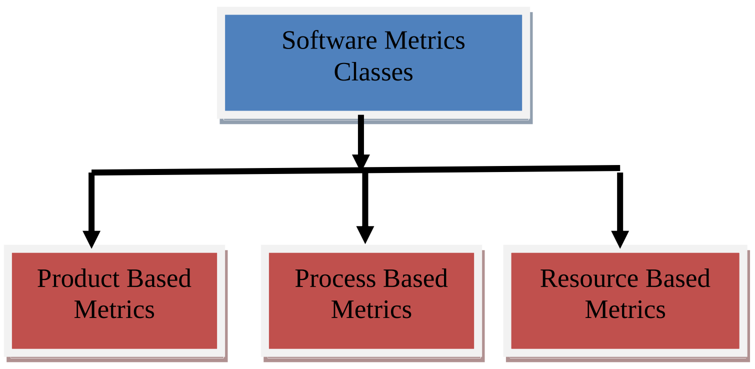

Fig 4 : Classes of Software Metrics

To perform each kind of software evaluation, there are different associated attributes [7][8]. Based on the evaluation type and the associated attributes, software metrics are divided in three main categories shown in Fig 4.

A. Product Based Metrics

In this kind of software measure, software code and design analysis is performed. This kind of software metrics class includes a vast range of metrics to measure the software product. It basically uses the external attributes of the software system such as code size, number of modules, number of classes etc. It is able to perform the measurement under different aspects such as usability, efficiency, portability. It performs the estimation based on the environmental vectors as well as the features of software product.

B. Process Based Metrics

These kinds of metrics plays important role in software system under the process modeling. These metrics are based on different processing terminology such as type of methodology used in software development, development time, change estimation in software modules and the documents, bug identification and analysis. These kinds of metrics also include the estimation in the quantitative forms such as SLOC (Source Line of Code) metric. These kinds of metrics are estimated based on the line count. In same way, module estimation and the interconnectivity analysis is also performed.

C. Resource Based Metrics

These kinds of metrics are concerned with the resource estimation in terms of availability and the requirement. These resources include the human and non human resources. Non human resources are the physical resources such as the material, computer, location etc.

III. **OBJECT ORIENTED METRICS**

Object Oriented Metrics itself defines a framework for the development for bug free software product of the system so that effective software development will be obtained. These metrics are responsible to develop a reliable software system. Object orientation is the basic framework adapted by most of the languages. The design and development process adapted by object oriented metrics is focused on the object definition and its computation under different vectors. Object oriented design and development basically focuses the software system entity and represented in the form of classes and the operations. The main motive of such software system is to deliver the quality product. To estimate the software quality we have defined a metric suit to quality all the object oriented metrics [9][10]. In this present work, the software quality estimation is defined under the estimation of software module, its components, its interfacing and the integration. The presented suit covers the concepts of software interfacing under different parameters and the values. It also defines the software system as the integration of these metrics to analyse the complexity of complete system. The analysis of the presented system is been defined at 4 levels shown in Fig 5.

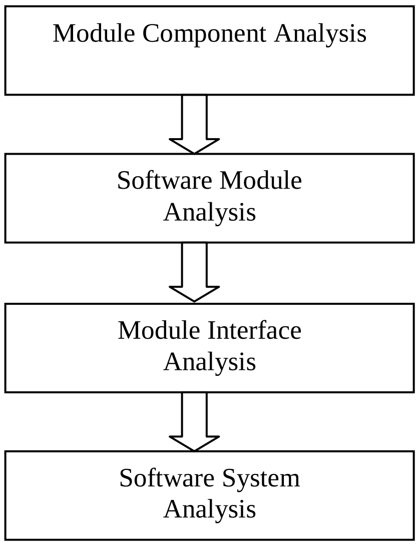

Fig 5: Stages of Object Oriented Metrics Analysis

As shown in Fig 5, the estimation of an object oriented software system is here defined under 4 stages. In first stage, individual software module is analysed under the attribute and method complexity analysis. In this work, the integration of the module attributes and the methods is analysed. Based on this analysis, quality of each individual module will be analysed. Once the module analysis will be done, in next stage, module analysis will be done as the whole. This kind of analysis estimates the direct integration of software system. Based on this integration, the direct calling of the software module with other modules is defined. In third phase of software system, the indirect integration between the software modules is estimated. This kind of integration is performed using the object based calling between different independent modules. At the final stage, the complete software system is analysed [11].

The presented study is about to estimate the software quality based on the software components in an object oriented software system. The presented system defines and validates the component and the interface metrics to analyse the software complexities. There are different methods and the properties defined with software system to identify the quality of software system.

Here the software component can be a method, attribute, class or the package. These software components and its integration with other software systems is the main aspects to estimate the software quality. According to the language specification, the definition of the software component can be changed. Such as in java beans, software component can be described as a bean component or the class where as in .net it can be defined as the activeX control, COM component, namespace of the class. To perform the component based estimation, some factors that are required to analyse are listed as under.

1. Identify different software components in the software system.
2. Identify the interrelation between different the modules and the internal components.
3. Identify the direct relationship between modules
4. Identify the Indirect relationship between modules.
5. Perform the metric based analysis on each module.
6. Perform the collective analysis on overall system.
7. Identify the software system complexity to represent the software quality

In this presented work, whole analysis is defined in terms of 4 metrics. These metrics are shown in Fig 6.

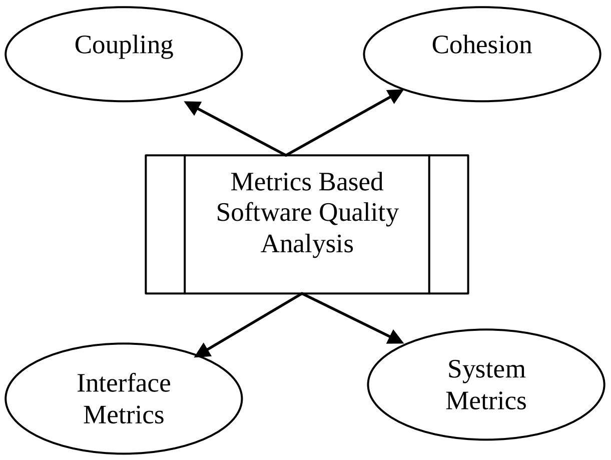

Fig 6 : Object Oriented Metrics

A. Cohesion

Cohesion is the relationship between the software module and its inter-defined components. This measure defines the attributes defined in a software module and its availability to the software module. The member interrelation is defined as the cohesion metrics. Let we have defined a Module with some attributes (A1,A2…Am) for a specific module and the instance variables are represented by V. In such case, cohesion metrics is given by the equation (1).

$$\text{Cohesion}(C) = \frac{|E(C)|}{|V(C)| * |M(C)|} \quad (1)$$

Here E(C) defines the pair set to represent the instance and the related member to the instance. V represent the variable, modified in that method M(C).

B. Coupling

Coupling defines the direct relationship between different modules. When two software modules are defined called C1 and C2, then the coupling is about to analyse the direct access of the members of module C1 in module C2. Coupling is defined by the inheritance process. It is defined as the quantitative measure so that the inter-relation between the components will be identified. It actually identifies the dependency between the components. This interrelationship is defined in the form of a directed graph. Let the software system is defined with M methods and V variables respective to the component C. Then MV defines the set of methods and the related instance variables of particular module Ci that in been used by module Cj. MVj will represent the set of the variables and the methods that are been accessed in other module. The formula for the coupling metrics is given by equation (2)

$$MV_j = \frac{|\bigcup_{1 \le i \le mn i \ne j} MV_{j,i}|}{|M_j| + |V_j|} \quad (2)$$

The software system is also defined under different coupling aspects. Two of most known coupling types are afferent coupling and efferent coupling. The first kind of coupling defines the components count that depends on sub components within the component. It actually analyse the dependency between two components.Whereas efferent coupling defines the component count where the sub-component depends on the main module.

C. Interface Metrics

This kind of metrics defines the relationship of the component with the outer environment. Outer environment is defined in terms of API function used in the software project. The API can be defined in terms of library function. The integration process is shown by equation (3).

$$\text{API}(m) = \frac{1}{N * k} \qquad (3)$$

Here the metrics is analysed for the module m. ni is the number of API functions used by module i. N represents the number of API methods used in the module m. and k represents the number of modules m. On the basis of this individual metrics, the complete system is been analysed.

D. System Metrics

To obtain the system metric a weighted system can be applied to all these metrics. This weighted form is shown in equation (4)

$$\text{System Metric} = W1 * \text{Cohesion} + W2 *(1\text{-Coupling}) + W3 * \text{Interface Metric}$$

Based on this metric analysis, the estimation of the software quality can be performed. Higher the complexity of the system, lower the quality of the software system

IV. CONCLUSIONS

The presented work is about the analysis of software system under the software quality using the object oriented metrics. In this work, software components are analysed individually as well as respective to other software modules and the library. Based on which the complete software quality is estimated.

REFERENCES

[1] Michelle Cartwright and Martin Shepperd,(2000)"*An Empirical Investigation of an Object-Oriented Software System*" IEEE Transactions on Software Engineering.

[2] M.P. Thapaliyal Garima Verma,(2010)*"Software Defects and Object Oriented Metrics – An Empirical Analysis"*, International Journal of Computer Applications (0975-8887) Volume 9– No.5, November 2010.

[3] Rajendra K. Bandi, Vijay K. Vaishnavi, Fellow, IEEE, and Daniel E. Turk, Member, IEEE,(2003)"*Predicting Maintenance Performance Using Object-Oriented Design Complexity Metrics*", IEEE Transactions on Software Engineering, vol. 29, no. 1, January 2003.

[4] R. L. Om, L. M. V. W. Urtemberg, H. Holm and O. Luczak,(2010) *"Identifying Factors Affecting Software Development Cost"*, Industrial Information and Control Systems The Royal Institute of Technology Stockholm, Sweden, 2010.

[5] R. Selvarani, T.R.Gopalakrishnan Nair and V. Kamakshi Prasad,(2009)" *Estimation of Defect proneness Using Design complexity Measurements in Object-Oriented Software*". International Conference on Signal Processing Systems 978-0-7695-3654-5/09© 2009 IEEE.

[6] Santonu Sarkar, Girish Maskeri Rama, and Avinash C. Kak (2007)"*API-Based and Information-Theoretic Metrics for Measuring the Quality of Software Modularization*" IEEE Transactions On Software Engineering, Vol. 33, No. 1, January 2007

[7] Software Engineering, (2004) "Volume of the Computing Curricula Series", The Joint Task Force on Computing Curricula IEEE Computing Society and Association for Computing Machinery.

[8] T. J. McCabe, "*A Complexity Measure,"* ICSE '76: Proceedings of the 2nd international conference on Software engineering, 1976.

[9] I.Sommerville, "*Software Engineering*", 7th edition, pp 1-31, 2004.

[10] Tieng Wei Koh, Mohd Hasan Selamat, Abdul Azim Abdul Ghani, Rusli Abdullah,(2008)" *"Review of Complexity Metrics for Object Oriented Software Products*", IJCSNS International Journal of Computer Science and Network Security.

[11] U. L. Kulkarni, (2010) "*Validation of CK metrics for Object Oriented Design Measurement*", Third International Conference on Emerging Trends in Engineering and Technology 978-0-7695-4246-1/10© 2010 IEEE.