RELEVANCE OF SOFTWARE METRICS FOR A SOFTWARE PROJECT

Sunil Dhawan*

Neenu Juneja**

Abstract: *The Central role that software development plays in today's world is beyond anyone's imagination. Software development is one of the major areas of focus for any organisations. The managers are increasingly focusing on process improvement in software development area. This demand has led to improved approaches to software development, with the most prominent being object-oriented software design. The focus on process improvement has increased the demand for software measures or metrics with which we can manage the things effectively and efficiently. This research addresses these needs through the development and relevance of metrics that can be used in a project. We have used Metrics developed in previous research, while contributing to the field's understanding of software design and finding out relevance to the real world tests that can provide us and idea about how relevant the measurement is for design of a software project.*

An automated data analysis tool was used to compare the relevance of these metrics in different projects. We have taken three projects of Web, Desktop and Mobile nature to ascertain the value proposition of these metrics.

Keywords: *Software Measurement, Metrics, Relevance, Relation, Performance Measurement, Complexity.*

*Department of Computer Science, NIMS University, Jaipur, India

**Department of Computer Applications, Chandigarh Group of Colleges, Landran, Mohali, India

I. INTRODUCTION

There has been a widely recognize fact that measuring a process is an important component in improvement of any design of a product. The focus on process improvement has increased the demand for software measures, or metrics with which to manage the process. Software Metrics are basically about measurement which involve numbers, the use of which aims at making things better in a way to improve the process of software development and all associated aspects of management of that process. Software Metrics can be defined as *"The continuous application of measurement based techniques to the software development process and its products to supply meaningful and timely management information, together with the use of the techniques to improve that process and its products."*[1] Software metrics are applicable to the complete life cycle from initiation to monitoring of reliability of end product and track the way the product changes over time with enhancement. [2]It covers the areas of controlling and monitoring the progress of the software development. In development of a project software metrics provide the basis of measurement using which we can predict its behaviour. Thus software metrics have high significance and relevance in making of a software project.

II. RESEARCH PROBLEM

The major criticisms that can be applied to current software designs are they apply old traditional, non-object oriented software design methodologies and measurements or metrics to develop a project which do not suffice enough theoretical base and lack appropriate properties. The traditional metrics and designs don't possess appropriate mathematical properties and consequently fail to display what might be termed as normal predictable behaviour. Thus the primary task is to find out software metrics that have relevance for the project. Relevance is the key that can be applied to a metric that helps in visualisation and prediction of a module. It can be employed to object oriented design and approach. This identification and selection of metrics would help in finding out the relevance of metrics in overall process and design improvement of different projects.

III. RELEVANCE OF METRICS

Relevance of metrics is a complete process that starts with identification and selection of metrics and then we use the relevant data for the metrics to analyse the impact of a metric on a software project. Higher the impact of a metric, higher would be its relevance.

Identification of metrics means to identify the metrics that we can associate with the relevant software application. It means what are the metrics that we are going to use for the assessment and measurement of software projects. These metrics must confirm to the Object Oriented methodologies having following components:

a. Identification of Potential classes and objects which can provide key abstractions to the key problem areas.
b. Finding out the meaning or semantics of the classes and objects identified in the first step, this includes definition of life cycles of each object.
c. Finding out the relationships between classes and objects interactions, such as patterns of inheritance among classes.
d. Construction of detailed internal views including definitions of methods and various behaviours under circumstances.

In order to check the efficiency and reliability of the software application we need to use certain metrics that can help us in estimation of different factors such as cost and estimation.

In this case we have chosen following metrics:

1. Token Count

Tokens are classified as either operators or operands. Any symbol used to represent data is considered an operand. Operators consist of arithmetic symbols and command names

n= n1+ n2

- n= Vocabulary of a program.
- n1= No. Of unique operators
- n2= no. Of unique operands.

2. *Program Length*

Program Length means the sum of occurrences of operators and operands. Any symbol used to represent data is considered an operand. Operators consist of arithmetic symbols and command names.

n=n1+n2

n= vocabulary of a program. n_1= no. of unique operators. n_2 = no. of unique operands.

N= Program Length.

$N = N_1 + N_2$

N_1= Total occurrences Of Operators.

N_2= Total occurrences of operands.

3. Estimated Program Length

Based on program Length, estimated program length allows us early insight on the program size. So that we can have an approx. figures with us using which we can calculate different factors which are critical to a project's success or failure.

$N\hat{} = n_1 \log n_1 + n_2 \log n_2$

4. Cyclomatic Complexity

Cyclomatic complexity is a software metric that provides a quantitative measure of logical complexity of a program. It aims at providing numerical data while evaluating complexity of software project.[10]

$V(G) = e-n+2$

$V(G)$ = Cyclomatic complexity

e = no. of flow graph edges

n = no. of flow graph nodes

5. Potential Volume

It is the actual size of a program if a uniform binary encoding for the vocabulary is used. It provides us size of the program. It is normally seen that different program use similar algorithm to implement or provide desired result.[9] The program that has minimal size or footprint is considered to have the potential volume. It is expressed as

$V^* = (2 + n_2^*) \log_2 (2 + n_2^*)$

V^* represents Potential Volume, 2 in the first parentheses represents the two unique operators for the procedure call.$n2^*$ represents the no. of conceptually unique input and output parameters.

6. Program Weakness

It is average of weakness of different modules as we know that a program is normally a combination of various modules. The weakness of a module is used to estimate the testability and maintainability. If weakness of a module is more, it is easy to test and if it is less it is easy to maintain.[8]

γ = Weakness of ith module. WP = Weakness of the program m= no. of modules in the program WM= ¯ * Avg. Life of Variables

7. Information Flow

Information Flow metrics model the degree of cohesion and coupling for a particular system component. IF metrics aims to derive information using metrics and how to use that Information.

IF (A) = [( ) × ( )]

***A** could be any component*

( ) = + + + a = no. of components that Call A.

b= no. of parameters passed to A from components higher in the hierarchy.

c = no. of parameters passed to A from components lower in the hierarchy.

d = no. of data elements read by component A.

( ) = + + + h e = the no. of components called by A

f = the no. of parameters passed from A to components in the higher hierarchy.

g = the no. of parameters passed from A to components in the lower hierarchy.

h = the no. of data elements written to by A.

8. System Complexity

It is the overall complexity of the system. We can compute system complexity as follows

C(i) = S(i) + D(i)

C(i) = System Complexity S(i) = Structural Complexity S (i) = S(i) = $f_{out}^2(i)$

Fan out is the number of modules directly invoked.

D (i) = Data Complexity D (i) = $v(i)/[f_{out}(i)+1]$

v(i) is the number of inputs and outputs passed to and from i.

System Complexity depends on Structural as well data complexity.

IV. ASSESSMENT OF SOFTWARE METRICS

We can use different metrics for the evaluation and improvement of software projects/ products. All the metrics that we use produce data. This data is expressed in terms of numerical values. [3] But it comprises of first stage of assessment only or we call it the starting point. The data acts as a base for assessment to make deductions about the quality of the software product and determine if it is good to be released to customers.

But this data is normally vary large and one cannot just assess the whole thing by just looking at the pages of data supplied by different metrics. We need to use statistics to understand the nos. to make deductions and produce evidence to support these deductions.[7] We need to assess data from different metrics in different manner e.g. analysing failure data needs very particular type of statistics and there is a range of models which specifically used to predict future reliability.

Statistics is a large body of knowledge which provides us various methods to be used with different metrics. The main purpose and primary functionality is:

A. Reduction of no. of variables Analysed

Statistics provides us various techniques such as Mean, Median, Mode, Linear Programming, Pie Charts, Box Plots etc. that help us by eliminating those variables which are not important to the context of metric/ data.[4] It reduces the no. of variables that are available to us in form of data so that we can make a decision as it is easy to take decision on smaller set of data compared to larger set of data.

B. Finding out Relationships

Statistics also provides programming techniques that can be used to set up relationships with data sets. It provides us with numbers that can be used to evaluate the bonding between data sets. Regression and Correlation techniques are used to perform relationship analysis.

C. Reliability Models for predicting future reliability

Statistics provide different tools to predict the future reliability of a project. As reliability is one of the most important aspects we need to correctly predict the future using probabilistic models.

V. USE OF SOFTWARE METRICS IN SOFTWARE QUALITY DESIGN

Software Quality means conforming to the set standards / requirements. Software quality is generally expressed in a defect rate and reliability. Quality must be defined and measured if improvement is to be achieved. Software quality is often referred to as 'Conformance to requirements'.

Defect Rate = No. of defects per Million Lines of Code / FP Reliability = No. of failures per n hours of operation.

To achieve the standards set for software quality we need to measure it for which we have metrics.[5] Software quality metrics are those software metrics that focus on quality aspects of product, process and project. The essence of software quality metrics is to investigate the relationship among project, process and end product quality.

There are many software metrics available for both product as well as processes. Software quality metrics attempts to quantify various quality oriented factors such as reliability and maintainability. Following are some of the facts:

1. Software quality factors need to be determined which are important to the application.[6]

2. Software metrics that correlate to these factors are used on the code to determine to what extent these factors have been reached.

3. Based on these results the developer determines whether the software meets the requirements set for it and how well the system will perform.[7]

VI. RELEVANCE OF SOFTWARE METRICS

We use the assessment techniques mentioned above to
find out its relevance using relevant data from a project consisting of various modules that allow us to validate the claims of a metric that whether it is relevant for our project or not. We have used a case study based on a banking solution that aims at providing enterprise and user level access via desktop, web and mobile solutions. If the metrics provide us the information that can help us in determining various factors such as cost, reliability and efficiency etc. then we can conclude that we have a set of metrics or tools that can provide us information that is not just relevant to the project but useful in planning the future course of development as well.

VII. CASE STUDY

A comprehensive study conducted on three projects Web Application, Desktop Application, Mobile Application. These three projects are based on a common problem and provide solution to that in form of a Banking Software solution which can provide the bank the necessary computational power needed for its day to day operations. All the three projects perform the functions desired by the bank management. But these projects implement these functions in a different manner. We have used 7 identified metrics to analyse and compare the performance and reliability factors of the projects.

Test Conditions:

1. Web and Desktop applications are developed and optimised for same platform. While Mobile application is optimized for android os.

Platform used:

Software: Microsoft Windows 7 Operating System, MS-ASP.net as development language, MS-Sql 2008 as database and MS- IIS 7.0 for web deployment purpose (local server)

2. We have used same platform for testing of all three projects. Hardware of Test System: Intel Core 2 duo 1.6 GHz CPU, 4 GB Ram, 320 Gb Hdd, 256 Mb discrete graphics card. Using Windows 7 professional as operating system.

3. For Testing of Mobile Application we have used a Samsung Galaxy S3 with Android Jelly Bean 4.1.1 OS.

4. We have used 10 test cases for each software metric and taken their average as the final result for comparison.

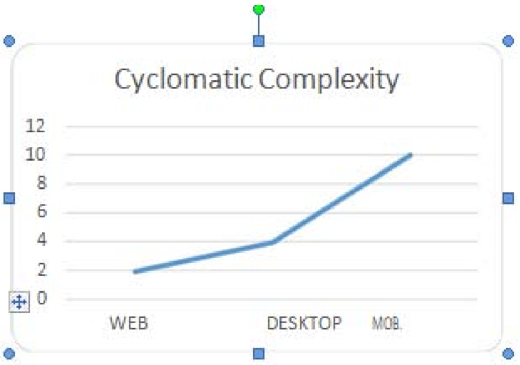
The results are computed over a period of 30 days and listed and compared in the below comparison table.

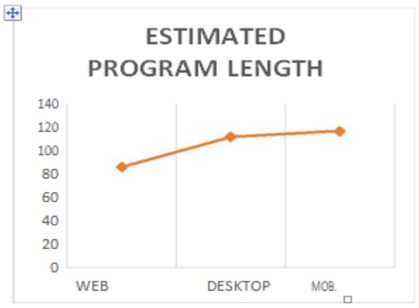
Sr. No.	**Name of Metric**	**Projects**		
		Web	**Desktop**	**Mobile**
1	Cyclomatic Complexity	2	4	10
2	Token Count	460	950	1100
3	Estimated Program Length	86.56	112.11	117.34
4	Volume	5092	7748	10242
5	Program Weakness	51.8	65.83	68.93
6	Information Flow	6631170624	7614656664	11045169220
7	System Complexity	310.81	412.92	531.13

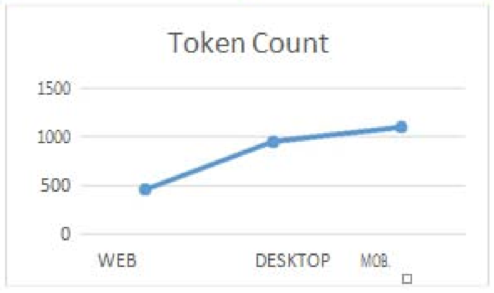
VII. EVALUATION OF CASE STUDY

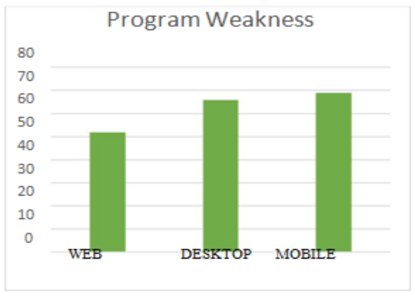
The evaluation of these three projects presents a very clear picture of the performance of three projects. It provides us reliable information that we can use to select the best possible solution for the given problem. We conducted this case study in order to see which solution is reliable and efficient. The evaluation provided us quantifiable scores using which we have one best performing(Web), one average performing (Desktop) and one worst performing (Mobile) option. The comparison reveals the truth about the claimed performance levels of projects and it allows us to select the best possible option.

Web application is most efficient and reliable in all the terms as it is highly optimised and would be easy to maintain compared to other two options available as they are slow in nature compared to Web Application.









In addition to the Web Application is optimised for 24 * 7 environment which is the basic requirement for expansion these days in case of banking industry. So it provides the scalability as well. We have also created some comparison charts to further elaborate the performance of three projects.

CONCLUSION

This research has identified and implemented a set of software metrics for Object oriented design that provide us data that depicts the relevance of these tools known as metrics. These metrics are based on measurement theory and also reflect the viewpoints of software developers which suggest some ways in which in OO approach may differ in terms of desirable or necessary features. In addition to the proposal and analytic test of theoretically grounded metrics, this paper has also presented empirical data on these metrics on different platforms. On the basis of the assessment we can conclude that the software metrics can play a very important role in selection and maintenance of software projects both at the development and end user level. The usual benefits obtained from valid measurements these metrics may be an especially critical one as organizations begin the process of migration from old software design methodologies to the new ones. It provides us right numerical data using which one can correctly differentiate between best and worst options. But one has to consider different environment scenarios as well. A caution of choosing the right metric is always there as there is no universally acceptable metric available that we can use to measure / assess all types of data.

REFERENCES

[1] Singh Yogesh & Pradeep Bhatia, " Module Weakness—A New Measure", ACM SIGSOFT Software Engineering Notes,81,July,1998

[2] Henry S. & Kafura D., "Software Structure Metrics Based on Information Flow", IEEE Trans. On Software Engineering SE-7, 5, 510-518, Sept. 1981

[3] Paul Goodman, " Practical Implementation of Software Metrics", McGraw Hill Book Co., UK, 1993

[4] Halstead M.H., "Elements of Software Science", New York, Elsevier North Holland, 1977

[5] Martin Neh, " Software Metrics for Product Assessment", McGraw Hill Book Co., UK, 1993

[6] Conte S.K., Dunsmore H.E., Shen V.Y., "Software Engineering Metrics and Models", The Benjamin / Cummings Pub. Co. Inc., California, USA, 1986.

[7] Dumsmore H.E. & Gannon J.D.," Analysis of The Effects of Programming Factors on Programming Effort", Journal of systems and software, 141-153, 1980

[8] Stephen H. Kan, " Metrics and Models in Software Quality Engineering, Second Edition", 2-82, 2002

[9] Ince D., " Software Metrics Measurement for Software Control and Assurance", New Yourk, Elsevier, 1989.

[10] Thomas J. McCabe, " A Complexity Measure", IEEE Transactions on Software

Engineering VOL. SE-2, No.4, December 1976

[11] Quality Software Project Management [Robert T. Futrell, Donald F. Shafer, Linda Isabell Shafer]

[12] Dhawan Sunil, Dr. Wadhwa Manoj, " Identification of Metrics for a Software Project." Sixth IRAJ International Conference, Oct., 6, 2013.