Volume 5, Issue 2, February 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Development of Software Metrics Framework for Reducing Cost of Project

Sunil Dhawan[1], Vikas Verma[2]
[1]Affiliation, Research Scholar, NIMS University, Jaipur, Rajasthan, India
[2]HOD, Department of Computer Applications, CBS, Landran, Punjab, India

Abstract: This paper provides an insight of how Software Metrics Framework can be used to build a platform that can help in improving the overall software design process. A framework provides a simple set of guidelines for approaching any software measurement tasks. Software framework brings together all the different components to enable development of a project or solution. This paper deals with various ambiguities that are normally responsible for a bad design. Thus using this framework we can create a better design through precision measurements. The paper also provides the fact that the cost of building and deploying a software metrics framework is much less compared to the cost of maintenance arising out of many issues relating to software design.

General Terms: Software Metrics for design, Software Metrics and their use. Design metrics, Software Metrics Framework, Metrics Design Framework.
Keywords: Software Design Improvement, Software Metrics Framework, Software Design.

I. INTRODUCTION

Any design in this world requires some sort of measurement. Without the use of measurement one cannot achieve the purpose of that particular design. And when contextual design or plan is to be formed the measurement is of utmost importance. Precision is the most important factor in engineering design for achievement of desired results. To attain desired results one requires certain target and a set of measurements and precise information of various factors and their attributes. A software metrics framework allows us to build a platform that can be used to distinguish the applicability of many metrics. It helps us determine what metrics can be used for a certain type of project. Once a software metrics framework is established the information provided by a set of metrics can be used to take important decisions with regards to software design and its allied factors. The critical factors of software design can be effectively measured and dealt with a scientific approach resulting in improvement in contextual plan and design of software.

II. DEFINITIONS

2.1 Software Metrics: A software metric is a measure of some property of a piece of software or its specifications. 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 those techniques to improve that process and its products.[1]

2.2 Metrics for Design / Design Metrics: These are themetrics which have direct relationship with contextual design of software. These metrics involve size, reuse, modularity, coupling, cohesiveness, functionality etc. as their primary measurement areas.

2.3 Software Metrics Framework: Software Metricsframework can be described as a universal, reusable software platform that helps us in development, deployment and analysis of various factors and issues relating to software design and development. It is a platform that provides us the ability to analyse and establish a way of distinguishing the applicability of many metrics. It provides a set of guidelines that can be used as a way of measuring various factors leading to software development.

III. NECESSITY OF A SOFTWARE METRICS FRAMEWORK:

Software Metrics are the core of software measurement process and one can achieve the required targets if software metrics are a part of each and every process of software design and development. The necessity for software metrics arise due to the very fact that software design or plan is a very complex process and one has to refine each and every process at many levels and multiple times till the customer and developer are not satisfied for various factors such as boot time, result time, size, lines of code, depth of inheritance, bandwidth used etc. Software Metrics are available in abundance. You can even create your own customised set of metrics for your own application. Software Metrics Framework provides some checks that make sure the applicability of a particular framework is there and in accordance with their use in an application. It is a platform for measurement and it only involves those metrics which have some relevance with the application or area we are dealing with. It has two primary functionalities:

a. Selection of Metrics: The first step involvesselection of metrics for a particular area of concern or metrics that can be used at product or process level. We can find out many metrics that can be used for measuring specifications, design, code, detailed design, usability and reliability etc. Thus selection of a set of metrics provides you the metrics that can be used in the next step to check their applicability in a given scenario. Using selection can be seen as a primary requirement for achievement of the goal.

b. Applicability of Metrics: In case wrong metrics areselected for analysis you won't be able to get the results. Wrong selection can be termed as bad selection or choice which can lead to bad design and other secondary issues. Thus we need to test the applicability of the selected set of metrics. We need to find out whether the chosen metrics or even a few of them can provide us relevant data or information providing a base for further decision making process.

IV. DEVELOPMENT OF SOFTWARE METRICS

Framework-WSMF-1

Any Measurement activity has to comply with the following five characteristics. These characteristics are the base of any measurement process and would be the part of WSMF-1 are discussed as follows:

a. Formulation: Formulation means to create oridentify some problem for which solution is to be thought about. Formulation of a problem should be quantifiable as using non-quantifiable data or problem can result in secondary issues and often lead to unfavourable conditions for decision making.[4] It becomes really very tough to take decisions based on non-statistical data.

b. Collection of Data: Collection of data is one of themost important aspects of any measurement process and these act as ingredients for the solution. Any part of ambiguity or carless attitude can lead to very serious issues in development. Collection of data must be in accordance with the format developed for the application and must not be in another form.

c. Analysis: Correct Analysis leads to proper decisionmaking. It is often seen that if analysis is correctly done one can see this as half the job done. Analysis of data is normally a complex process involving
many statistical analysis and interpretations and must be strictly according to the application or framework you are using. Analysis can also be done in a generalised way if the results using the generic algorithm leads us to a satisfactory level.

d. Interpretation: Once the work on analysis iscompleted one has statistical information or data for further analysis or developing viewpoint out of it. Interpretation is one such task. It is like getting refined information out of the data collected and analysed. Interpretation is done using the requirements of the formulated problems. Interpretation is a process that formally puts everything into place and creates a base using which we can take some important decision that can collectively be recognised as solution for the formulated problem.

e. Feedback: Any solution found has to be tested in areal environment and then we can employ that solution. Feedback provides us the important data relating to solution we have designed. Normally while designing a particular solution various factors and results are kept in mind. We need to check with the feedback data if there are any derivations with the standards. Once derivations are found we any derivations found can be corrected in the solution made so that a good solution can be formulated.

4.1 Objectives for Software Metrics

Framework: The objectives are the goals that onemust achieve for success. Software Metrics Framework needs to solve some primary goals or objectives thus while creating a framework this is the first step towards the development. A preliminary study has to be conducted to reveal the centre goal for a system.

The primary goal of WSMF-1 is to lower down the cost by bringing down the wastage of resources at many points in a software design. Using this as a primary goal two types of attributes as goals can be formulated as internal and external which can be used for further elaboration.

4.2 Finding out areas of concern for Software Metrics Framework-WSMF-1:

Using the primary goal of Software Metrics Framework one can find out the relevant areas where the problem lies or which needs to be corrected. The areas of concern cannot be limited to the basic needs of the goals but to its affected areas as well.

Internal Factors: The resources are used at many pointwhile designing a software/ system. Following internal factors or attributes were found that is used in WSMF-1 as the Internal factors of concern for the problem.

4.2.1 Size: Size is the first internal factor. Normally it isseen that as the size of a project grows the wastage of resources also grow. Thus this is one of the critical factors that is to be dealt. Size involves use of many metrics which would provide data relating to size of modules, links, projects which is the part of WSMF-1 framework.

4.2.2 Reuse: Reuse is the second internal factor. It isseen that if a component is reused only once its cost is cut down to half. Thus reusing a component in software design can help lower down the costs as well as control the wastage of resources. Reuse has a set of metrics that evaluates how and where components can be reused and what can be its effect on the performance of the software. Metrics for reusability would be the part of WSMF-1 framework with certain limits to no. of times a component can be reused to address the resource availability factor.

4.2.3 Modularity: Modularity is the third internal factorthat must be taken care of if one needs each and every module of the system to work effectively and precisely. Modularity refers to the extent to which a software may be divided into smaller modules. Modularity indicates that the number of application modules are capable of serving a specified domain. Modularity offers greater software development manageability.[2] Metrics for modularity would be the part of WSMF-1

framework due to the critical importance to efficiency as modularity provides the effective structure for increasing efficiency.

4.2.4Coupling: Coupling is an indication of the strengthof interconnections between program units. Highly coupled have program units dependent on each other.

Loosely coupled are made up of units that are independent or almost independent. Modules are independent if they can function completely without the presence of the other. But it is not possible up to a certain extent modules must work together in order to produce the required output. Thus it is the relationship of more dependent or less dependent that decides if the coupling is high or loose. Metrics for coupling must be kept as a part of WSMF-1 framework to measure what level of coupling is there in a project.

4.2.5 Cohesiveness: Cohesiveness or Cohesion tells ushow well module fits together. A component should implement a single logical function or single logical entity. There are many levels of cohesion that must be taken into consideration for a framework and metrics for cohesion must be a part of the framework. Coincidental cohesion occurs when components are not related but simply bundled into a single component. Logical association makes code for more than one function may be intertwined, leading to severe maintenance problems. Temporal cohesion occurs when functions are weekly related to one another. These are some of the commonly used cohesion that makes analysing cohesion components even more critical to evaluate a design as wrong cohesion requirements could prove to be a high cost maintenance affair and cohesion metrics are part of WSMF-1 framework.

External Factors: In addition to the internal factorsthere are few external factors that directly impact the performance of the design.[3] The resources are used at many point while designing a software/ system. Following internal factors or attributes were found that is used in WSMF-1 as the External factors of concern for the problem.

4.2.6 Quality of Design: Quality of design is one of theexternal factors that is one of the prime concerns in development of Software Metrics Framework. Quality of design means that the contextual design has been put in after conforming it to a standardised process as only then we can expect quality or reliable outputs from the design.

4.2.7 Complexity of Design: Complexity of design isanother aspect that is crucial factor to the involved entities. It is often said that no two modules of a project. Thus the complexities involved in creation of a module is different from the other one. Thus providing same level of complexity would not be prevalent in this case. e.g. size metrics normally ignore the complexity involved in creation of functions of a module. A module might have 10 bloc but might have simple functions compared to a module where the code is just 1 kloc but the functionality involves many complex and critical functions. Thus both modules can't be put on a similar scale. This makes counting complexity as a factor. For counting complexity one needs complexity based metrics to be part of the process. Complexity metrics such as Cyclomatic complexity/ MCCABE's cyclomatic complexity is to be added to the set of metrics to measure complexity of design.

4.2.8Maintainability of Design: Maintainability meansthe process or the product drums in perfect or near perfect level. We need to take care of many emergency or unwarranted situations that may arise. Because we can only control those situations only for which a contingency plan is already in place. So the maximum focus must be on low level of maintenance because zero level of maintenance can only be dealt in case of theories only. But one should focus on low maintenance as higher the maintenance higher would be the cost. The more you require it the more it would add towards the total cost of the project. Metrics relating to maintainability are kept at various levels in the WSMF-1 framework. The set of metrics must act as a deterrent to the increasing cost of maintenance.

4.3 Evaluation of Available Metrics for

areas of concern: There are numerous metricsthat are available for the areas of concern for development of framework. But it is not possible to allow each and every metric to be the part of the framework as this would increase the cost and time frame for evaluation. Following is the list of few of the available metrics for the purpose:

Table 1. Table of Metrics for Different areas of Concern

Sr No.	**Name of Metric**	**Area of Metric**
1	LOC/KLOC	SIZE
2	SLOC	SIZE
3	BUGS PER CODE SEGMENT	MAINTAINABILITY
4	CODE COVERAGE	QUALITY OF DESIGN
5	CLASS/ METHOD LENGTH	DESIGN/ DEVELOPMENT

6	NO. OF ATTRIBUTES IN A CLASS	DESIGN/ DEVELOPMENT
7	CONSTRUCTOR PARAMETER COUNTS	DESIGN/ DEVELOPMENT
8	STRING USAGE IN CODE FILES	DESIGN/ DEVELOPMENT
9	COMMENT LINE %AGE	SIZE/ DESIGN
10	CYCLOMATIC COMPLEXITY	DESIGN COMPLEXITY
11	TOKEN COUNT	SIZE / PROJECT
12	ESTIMATED PROGRAM LENGTH	SIZE / PROJECT
13	POTENTIAL VOLUME	SIZE/ DESIGN
14	LANGUAGE LEVEL	PERFORMANCE

The listed metrics are evaluated according to the areas of the concern using which we can decide on what metrics should be the part of our framework.

Criteria for Evaluation: Only those metrics which canbe helpful in reducing cost and have some impact on the design should be considered for framework. The metrics that have direct impact factor must be given priority as these metrics measure the core data for the application or the software.

4.4 Selection of Metrics based on Evaluation

It should be considered as one of the final stages for creation of this framework. Based on the evaluation conducted on the metrics available one should choose the metrics. The main aim was to reduce cost and improvement in contextual design of the software. Following are the metrics that can be the part of WSMF-1 software metrics framework. Out of the available metrics 10 metrics are selected based on their area of concern. These metrics take care of the issues relating to the design and common cost reduction using which it becomes possible for us to indicate the problem scenarios in a scientific way.Following is the table of selected metrics:

Table 2

Sr No.	**Name of Metric**	**Area of Metric**
1	SLOC	SIZE
2	BUGS PER CODE SEGMENT	MAINTAINABILITY
3	CODE COVERAGE	QUALITY OF

		DESIGN
4	CLASS/ METHOD LENGTH	DESIGN/ DEVELOPMENT
5	CONSTRUCTOR PARAMETER COUNTS	DESIGN/ DEVELOPMENT
6	COMMENT LINE %AGE	SIZE/ DESIGN
7	CYCLOMATIC COMPLEXITY	DESIGN COMPLEXITY
8	TOKEN COUNT	SIZE / PROJECT
9	ESTIMATED PROGRAM LENGTH	SIZE / PROJECT
10	POTENTIAL VOLUME	SIZE/ DESIGN

V. CONCLUSION AND FUTUREWORK

The WSMF-1 framework is a platform using which one gains access to a set of metrics that can be used to compute and analyse various properties of a given module or a project as a whole. One can have access to the information relating to relationship of modules. The overall impact of various internal and external factors can be measured in terms of data provided by the set of metrics. Various metrics such as cyclomatic complexity, method length and token count have direct impact on the design. If properly measured we can use the information to interpret various results that can be useful in decision making.

REFERENCES

[1] Paul Goodman, " Practical Implementation of Software Metrics", McGrawHill, UK, 1993
[2] Cagdas Basaraner, " Java DZone articles 5 common metrics for atomised software"2012
[3] Shyam R.Chidamber and Chris F. Kemerer, " A Metrics S u i t e f o r O b j e c t O r i e n t e d D e s i g n " , I E E E TRANSACTIONS ON SOFTWARE ENGINEERING, VOL 20 NO. 6, JUNE 1994
[4] Singh Yogesh & Pradeep Bhatia, " Module Weakness—A NewMeasure", ACM SIGSOFT Software Engineering Notes,81,July,1998
[5] Henry S. & Kafura D., "Software Structure Metrics Based on Information Flow", IEEE Trans. On Software Engineering SE-7, 5, 510-518, Sept. 1981

