IJITKM Volume 8 • Number 1 June-Dec 2014 pp. 41-45 (ISSN 0973-4414)

Selection, Classification and Computation of an Optimal Set of Software Metrics for Requirement, Design and Testing Phases of SDLC

Shefali Singla1 , Dr. Dheerendra Singh2, Yogesh Verma3, Dr. R. NandaKumar4
1, 2Shaheed Udham Singh College of Engg, Tangori,Mohali, Punjab, India
3,4 Space Applications Centre, Ahmedabad, India
shefalis51@gmail.com, professordsingh@gmail.com, yogeshverma@sac.isro.gov.in, nandakumar@sac.isro.gov.in

ABSTRACT

This paper presents a comparative study is on testing and design tools such as Sonar Qube, Simple code and Eclipse metric. This delivers tool-dependent metrics results and has even implications on the results of analyses based on these metrics results. In short, the metrics based assessment of a software system and measures taken to improve the quality of software. To support our case, we conducted an experiment with a 11 free metrics tools. We found that we get different metrics form open source tools. By analyzing the results of all free source tools we conclude that Sonar Qube provides best result from all open source tools.

Keywords: Software metrics, requirements metrics, design metrics, testing metrics, open source tools, sonar.

1. INTRODUCTION

Accurate measurement is a prerequisite for all engineering disciplines, and software engineering is not an exception. For decades seek engineers and researchers to express features of software with numbers in order to facilitate software quality assessment. A large body of software quality metrics have been developed, and numerous tools exist to collect metrics from program representations. This large variety of tools allows a user to select the tool best suited, e.g., depending on its handling, tool support, or price.

The definition of the metrics we first selected refers to the standard literature. During implementation, we found that the metrics definitions are ambiguous and there are lots of metrics which is used to improve the quality of software, as we are working on the three phases of software development life cycle SDLC, we make classification of metrics according to their category, here we provide its description, formula, category refer in the literature study part. After classification of metrics, we have selected open source tools which are used to compute this metrics.

2. METRICS SELECTION

We composed a list of metrics based on the literature survey from various sources like web and papers to select a set of metrics for requirements, design and testing phase of software development. These metrics were further analysed and categorized from an organization point and converted in to an optimal set of metrics.

Under requirement phase , total 27 quantitative metrics were selected and categorized under sever different categories.

Table-1 Summary of Requirement Metrics

SN	Category Name	No of selected metrics
1.	Productivity	4
2.	Quality	6
3.	Volatile	3
4.	Understand ability	4
5.	Traceability	2
6.	Effort	3
7.	Miscellaneous	5

In table 1 , there are 4 metrics for the productivity of an software, 6 metrics which gives the value of quality, 3 metrics which tells the volatility of an software, 4 metrics estimates whether the given software is understandable or not, and the 3 metrics give insight about how much effort will be consumed while work on that particular software.

Some of the important metrics are requirements traceability, changed, Unambiguous, consistency and not redundant.

Similarly in design phase, in total 42 metrics were considered as per the table 2. Here, 14 metrics specify that how many no of components depend upon the input of another component, 4 metrics defines about the reusability of metrics, 1 metric is about the progress of the software and 4 metrics are the package metrics.

The set of metrics includes no of methods, McCabe's complexity, No of attributes, no of children, abstractness, cohesion, coupling etc.

Table-2 Summary of Design Metrics

SN	Category	Selected metrics
1.	Component Interaction Metrics	14
2.	Connector Metrics	2
3.	Composite Metrics	1
4.	Reusability Metrics	4
5.	Functional Coverage Metrics	4
6.	Progress Metric	1
7.	Package Metric	4
8.	Coupling metric	2
9.	Un Categorized	10

In testing phase, total 46 metrics were collected, analysed and categorized under different categories as per table -3

Table-3 Summary of Testing Metrics

S N	Category	No of metrics
1.	Defect	15
2.	Effort	4
3.	Efficiency	3
4.	Quality	9
5.	Traceability	1
6.	Stability	1
7.	Progress	12
8.	Productivity	1

The set of metrics consist of defect density, defect pattern, time to fix a defect, no of defects, test effort percentage, test efficiency , quality of fixes and so on.

3. OPTIMAL SET OF METRICS

The table 4 presents the optimal set of metrics of three phases based on the literature survey

Table-4 List of optimal metrics for Organization

Phase	Metrics
Requirements phase	Metric Name, Uniqueness, Correctness, Changed Requirements, Misinterpreted Requirement, Understandable Requirements, Modifiable, Traced, Requirement Testing SRS Quality, Unambiguous, Complete, Understandable, Verifiable, Internal Consistent, Annotated By Relative Importance, Annotated By Relative Stability, Annotated By Version, Precise, Traceable, Not redundant, At Right level of detail Organized, Achievable, Electronically stored, concise, Design independent, reusable
Design phase	Total Number of Component (TNC), Average No of Methods Per Component (ANMC), Total Number of Implemented Components (TNIC) Total Number of Links (TNL), Average no of Links Between Component (ANLC), Width of the Composition Tree(WCT), Component dependency Metrics, Component Interaction dependency Metrics (CIDM), Size of the Architecture, Component Density of Architecture, Average Distance of Architecture, Component Customization Metrics, Component Reusability Metrics, Existence of Meta Information(EMI), Rate of Component Observability (RCO), Rate of Component Customizability (RCC), Self Completeness of Component's Parameter, Function Coverage, Compliance Familiarity, Support, Proximity Metric, Data Model Compliance Index (DCMI), Functional Model Compliance Index (FMCI), Component Usage Strength (CUS), Component Backward Dependency Strength (CBDS), Abstractness, Instability, Distance from the Main Sequence, Attribute Hiding Factor (AHF), Method Hiding factor (MHI), Polymorphism Factor (PF) Structure Complexity of a Module, Data Complexity of Module, Module Coupling Indicator Mc, Revised Coupling Metric, Function Point, Bang Metric
Testing phase	Defect Category, Defect Cause Distribution, Defect Finding Rate, Defect Removal Efficiency, Defect Severity Index, Number of Defects, Return of Test Effort(RTE), Test Case Effectiveness Time To Fix A Defect, Defects By Injection Phase, Defect By Cause, Defect Pattern, Defect Detection Difference, Defect Closure Period, Defect Density, Effort Adherence, Test Effort Percentage, Percent Automatable, Test Effort Performance, Review Efficiency, Test Case Execution Statistics, Automation Progress, Actual Impact On Quality, Function Point, Current Quality Ratio, Quality Of Fixes, Problem Report, Test Efficiency, Test Case Defect Density Effectiveness Of Smoke Test, Customer Problem Metric, Traceability, Scope Changes, Test Procedure Execution Status, Error Discovery Rate, Defect Aging, Defect Fix Retest, Defect Acceptance, Bad Fix Defect, Schedule Adherence, Suspension Criteria For Testing, The Exit Criteria, Scope Of Testing, Monitoring Of Testing Status, Test Case generation Productivity

4. OPEN SOURCE TOOLS EVALUATION & METRIC COLLECTION

Further to automate metrics collection task, following set of tools were considered from open source domain particularly for design and implementation phase:

I. Sonar
II. CCCC
III. Chidamber &Kemerer Java Metrics(ckjm)
IV. Eclipse Metrics Plug-in 1.3.6
V. Simple Code Metrics
VI. Squale
VII. SourceMonitor
VIII. LocMetrics
IX. LineTally
X. Unified Code Count
XI. Universal Code Line Counter

Further, Tools were installed, configured and evaluated up to the criteria set by the organization i.e. set of metrics.

Tools gives us the metrics namely Lines of Code(LOC), Number of methods(NOM), McCabe's Complexity(MV), Weighted Methods per class (WMC) /Total Complexity, Depth Inheritance Tree(DIT), Number Of Children(NSC), Number Of Attributes(NOA), Lack Of Cohesion in Methods(LCOM) , Afferent Coupling(Ca) , Efferent Coupling(Ce) , Number of static methods(NSM) , Normalized Distance(RMD) , Specialized Index(SIX) , Instability(RMI) , Number of Overridden methods(NORM) , Number Of interfaces(NOI) , Number Of Parameters(PAR) , Abstractness(RMA) Nested Block depth(NBD), Number of static attributes(NSF) , Number of Classes(NOC) , Method Lines of Code(MLOC) , Documented API , Undocumented API , Number Of Assessors, Number Of Files(NOF), Complexity per function, Lines of Comments(COM), Cyclomatic complexity per class, Cyclomatic complexity per file, Line of Imports, Number of Public Methods(NPM), Number of Source files(Nsou), Number of Statements, Percentage of Duplicate Lines, Duplicated Lines, Duplicated Files, Duplicated Blocks. A total set of around 38 metrics involving various phases of software development life cycle.

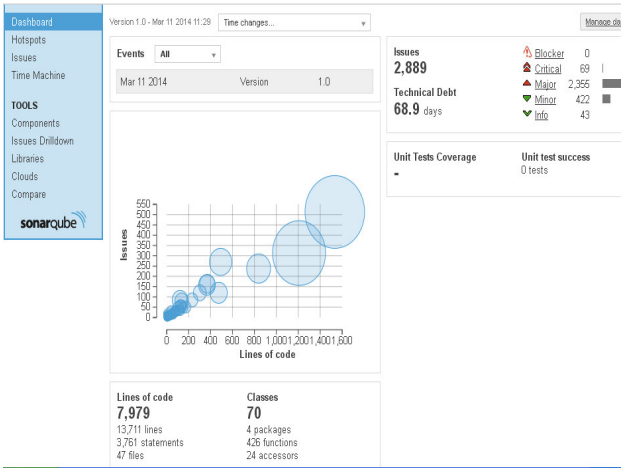

Fig 1 Sonar output on an open source java code

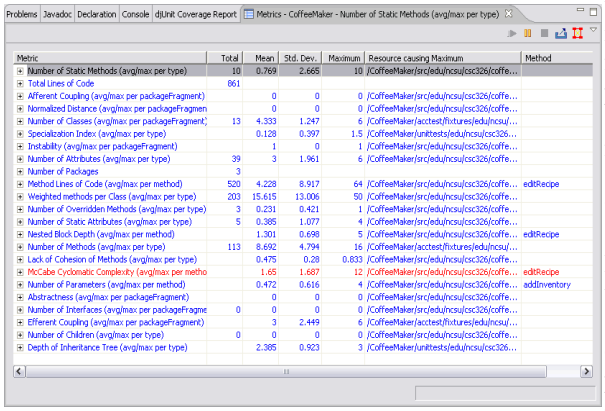

Fig 2 Eclipse output on an open source java code

Basically on comparing the results on these three tools, we see that for calculating the number of methods both eclipse and sonar consider the constructors as methods only. But in eclipse, the final running class is not counted among the number of methods. In Net beans regarding this issue we see that only public methods are counted. Even regarding the complexity factor, we see that in Net beans the average complexity is counted only for the public methods of the main class. The total complexity shown in Sonar actually conforms to the weighted methods per class metric.

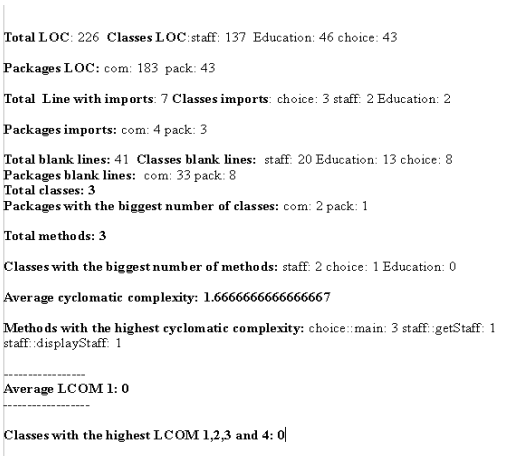

Fig 3 Net beans output on an open source java code

Out of all the evaluated tools, Sonar gives us a good set of metrics as well rich features from testers, developers, and management point of view.

5. CONCLUSION & FUTURE WORK

Based upon the literature study, the list of software metrics for requirement, design and testing phases has been evaluated, selected and classified to be used as listed: A set of 27 initial metrics are selected for the first phase (Requirement Phase), among them there are4 productivity metrics, 6 quality metrics, 3 volatile metrics, 4 understand ability metrics, 2 traceability metrics, 3 effort metrics and 5 miscellaneous metrics. A set of 42 unique metrics are selected for the second phase (Design Phase), among them there are 14 component interaction metrics, 2 connector metrics, 1 composite metrics, 4 reusability metrics, 4 functional coverage metrics, 1 proximity metric, 1 progress metrics, 4 package metrics, 2 coupling metrics, 10 un categorized. A set of 46 unique metrics are selected for the third phase (Testing Phase), among them there are 15 for defect metrics, 4 effort metrics, 3 efficiency metrics, 9 quality metrics, 1 traceability metrics, 1 stability metrics, 11 progress metrics and 1 is productivity metrics.

In this research a comparative study is done on testing tools such as Sonar Qube, simple code (Net beans plug-in) and Eclipse metrics plug-in 1.3.6, for computing the optimal set of metrics for requirement, design and testing phase of software development life cycle. Sonar Qube computes the 30 optimal metrics, where the Eclipse metrics plug-in 1.3.6 computes 23 optimal metrics and

simple code (Net beans plug-in) computes 5 optimal metrics. Sonar has a provision for integration with other plugins / tools namely configuration management, PDF report generation and JMeter Testing. Sonar Qube gives the graphical representation which tells about the critical, major and minor issues of metrics whereas, rest of two not show the graphical representation of metrics that are only text based. Further, we find more results on open source tools for computing, evaluating and presenting of metrics at each phase of software development life cycle (SDLC).

6. ACKNOWLEDGMENTS

The author is highly grateful to Dr. Manjit Singh Grewal, Director Shaheed Udham Singh College of Engineering and Technology (SUSCET), Tangori (Mohali), for providing this opportunity to carry out the present thesis work. I would like to express a deep sense of gratitude and thanks profusely to my Supervisor Dr. Dheerendra Singh, Professor and Head, Department of Computer Science & Engineering,. Without the wise counsel and able guidance, it would have been impossible to complete the thesis in this manner. The constant guidance and encouragement received from Shri. Yogesh Verma, Sci./Engr-SD, GSQAD-RQAG-SRA, SAC-ISRO and Dr. R. Nandakumar, Head, GSQAD-RQAG-SRA, SAC-ISRO, AHMEDABAD has been of great help in carrying out the present work and is acknowledged with reverential thanks. I express gratitude to other faculty members of Department of CSE, for their intellectual support throughout the course of this work.

7. REFERENCES

[1] [3 April, 2013], Cecilia Rodriguez Matrinez "Software quality studies using analytical metric analysis"

[2] [6Dec,2013],DavidRacodon"http://docs.codehaus.org/display/SONAR/metric+definitions"

[3] [April, 2013], Rudiger Lincke, Jonas Lundberg and Welflowe "Comparing Software Metrics Tools"

[4] [2005], QaiserDurrani , "Role of Software Metrics in Software Engineering and Requirements Analysis"

[5] [May 2013], Mohd.Haleem, Prof (Dr) Mohd.Rizwan Beg, Sheikh Fahad Ahmad, "Overview of Impact of Requirement Metrics in Software Development Environment"

[6] [January 2012], Shahid Iqbal and M. Naeem Ahmed Khan Shaheed Zulfiqar Ali, "Yet another Set of Requirement Metrics for Software Thesiss Bhutto Institute of Science and Technology (SZABIST) Islamabad"

[7] [2 March, 2009], SubhajitDatta , " Metrics And Techniques To Guide Software Development" Florida State University College Of Arts And Sciences, P.H.D

[8] [31 March, 2005], Ganesh Kandula, Vinay Kumar Sathrasala, "Product and Management Metrics for Requirements" Master Thesis

[9] [June 15, 2006], Mohammed Javeed Ali, "Metrics for Requirements Engineering Master's Thesis" , University Department of Computing Science Sweden

[10] [10 March, 2011], Mohammad UbaidullahBokhari and Shams Tabrez Siddiqui , "Metrics for Requirements Engineering and Automated Requirements Tools", Department of Computer Science, AMU, Aligarh

[11] [May, 2003], Hilda B. Klasky , "A Study of Software Metrics " , Graduate School-New Brunswick Rutgers, The State University of New Jersey

[12] [20 Dec., 2001], Vanita Shroff , "Requirements Management Metrics", Master's in Software Engineering Department of Electrical and Computer Engineering, University of Calgary, Canada

[13] [2003], James R McCoy, "Requirements use case tool", Conference on Object Oriented Programming Systems Languages,

[14][Visited 2013], "Teleology Requirements Engineering Tools", http://www.telelogic.com/com/products.doors analyst/index.cfm

[15] [Oct 1996], Victor R. Basili, Fellow, IEEE, "A Validation of Object-Oriented Design", IEEE Computer Society IEEE Transactions On Software Engineering, Vol. 22, No. 10.

[16] [June 1994], Marian Jureczko And Diomidis D. Spinellis, "Using Object-Oriented Design Metrics To Predict Software Defects "

[17] [Oct 1996], Victor R. Basili, Fellow, IEEE, "A Validation of Object- Oriented Class Metrics As Quality Indicators". IEEE Transactions On Software Engineering, Vol. 22.

[18] [2009], Professor D. Vernon, "Khalifa University Course Notes "

[19] [June 1994] ,Shyam R. Chidambaram and Chris F. Kemmerer, "A Metrics Suite for Object Oriented Design" , IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 20, NO. 6

[20] [Sept 1993], Dewayne E. Perry and Carol S.Steig, "An Empirical Approach to Design Metrics and Judgments"

[21] Karin Erni , "Applying Design-Metrics to Object-Oriented Frameworks"

[22] [April 2003], Ramanath Subramanyam and M.S. Krishnan , "Empirical Analysis of CK Metrics for Object-Oriented Design Complexity: Implications for Software Defects" , IEEE Transactions On Software Engineering, Vol. 29, No. 4

[23] [Jan 2012], Mrinal Singh Rawat, "Survey on Impact of Software Metrics on Software Quality", Department of Computer Science MGM's COET, Noida, India International Journal of Advanced Computer Science and Applications, Vol. 3, No.

[24] [2008], Jie Xu , "An Empirical Validation of Object-Oriented Design Metrics for Fault Prediction by Department of Electrical and Computer Engineering", University of Western Ontario, London, Ontario, Journal of Computer Science 4 (7): 571-577, 2008 ISSN 1549-3636 © 2008 Science Publications

[25] [2006], Denys Poshyvanyk, "The Conceptual Coupling Metrics for Object-Oriented Systems by Department of Computer Science", Wayne State University Detroit Michigan

[26] [2001], Daniel Rodriguez and Rachel Harrison, "An Overview of Object-Oriented Design Metrics"
[27] [Jan 2007], Wasif Afzal , "Metrics in Software Test Planning and Test Design Processes", School of Engineering Blekinge Institute of Technology Sweden
[28] [May 2003] , Hilda B. Klasky, "A Study Of Software Metrics " A thesis submitted to the Graduate School-New Brunswick Rutgers, The State University of New Jersey
[29] NachiappanNagappan , "Using Historical In-Process and Product Metrics for Early Estimation of Software Failures", Microsoft Research
[30] [1999], Thom Garrett IDT, "Useful Automated Software Testing Metrics"
[31] [1999], Elfriede Dustin, Thom Garrett, Bernie Gauf, "Implementing Automated Software Testing"
[32] [Aug 2006], Ramesh pusala, "Operational Excellence Through Efficient Software Testing Metrics"
[33] [Aug2006], "INFOSYS operational excellence through efficient software testing metrics".
[34] [2009], "Software Measurements And Metrics: Role In Effective Software Testing
[35] [Jan 2011] Sheikh Umar Farooq, Department of Computer Sciences, University of Kashmir Srinagar, J&K
[36] [May 2011], Vikas Verma and Sona Malhotra "Applications Of Software Testing Metrics In Constructing Models Of The Software Development Process", M.Tech Scholar, CSE Department, U.I.E.T. Kurukshetra University, Kurukshetra, India
[37] "www.mindlance testing.com."
[38] [Jan 2007], Wasif Afzal, "Metrics in Software Test Planning and Test Design Process", School of Engineering Blekinge Institute of Technology Sweden
[39] [1994], J. R. Horgan,Bellcore and Saul London,Bellcore and Michael R. LyuBellcore, "Achieving Software Quality with Testing Coverage Measures: Metrics, Tools, and Applications", IEEE
[40] [2005], Linda Westfall, "12 Steps to Useful Software Metrics"
[41] [2011], Muhammad Shahid1 and Suhaimi Ibrahim and Mohd Naz'riMahrin, "A Study on Test Coverage in Software Testing".
[42] [Aug 1979] , John B. Bowen and Hughes-Fullerton, "A Survey of Standards and Proposed Metrics for Software Quality Testing"
[43] [Aug, 1991], U.S. Department of Transportation Federal Aviation Administration, "Software Quality Metrics Overview"
[44] [Jan 2007], Wasif Afzal, "A Critique of Software Defect Prediction Norman Fenton and Martin Neil Metrics in Software Test Planning and Test Design Processes".