Volume 4, Issue 1, January 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

A Novel Software Metrics and Software Coding Measurement in Software Engineering

K.P. Srinivasan
Associate Professor of Computer Science
C.B.M. College, Kovaipudur,
Coimbatore – 641 042, India

Dr T. Devi
Head, Department of Computer Applications,
School of Computer Science and Engineering
Bharathiar University, Coimbatore – 641 046, India

Abstract—***The software measurement activity adds values and is kept as a part of every phase of the software development process. The software metrics can help the software professionals to make specific characteristics of software processes and products more visible. Further, software measurement includes quantitative evaluations of software and usually metrics can be used directly to determine achievements of quality goals quantitatively. At present, software metrics research tries to prove that the software metrics is playing an important role in understanding and controlling the software processes and products in software industry. This research paper introduces a new kind of software metrics called Program Keyword Metrics (PKM) for source code measurement and two keyword metrics named Program Keyword Vocabulary (PKV) and Program Each Keyword Length (PEKL). This metric measure the keywords of a programming language of coding phase. The keyword measurements are important in programs and presently this paper proposes two keyword metrics called Program Keyword Vocabulary and Program Each Keyword Length metrics. These metrics give the keyword usage in the program for judging the usage of keywords in the program. In order to quantify the effectiveness of program, these new software metrics are introduced.***

Keywords—Program Keyword Metrics, Software Coding Measurement, Software Quality Assessment, Result Based Software Metrics, Software Measurement Procedure, Software Metrics.

I. INTRODUCTION

The software measurement and well defined metrics have become a key component and good principles of software engineering [6, 9, 10, 14]. Further, software metrics are quantitative and qualitative measures that facilitate software developers to increase the "Process Efficiency and Product Effectiveness (PEPE)" [16, 18, 20]. The word "metrics" refers to a wide range of activities that are related to software measurement in software engineering. The definition of software metrics has taken various forms since its beginning in the software engineering. The IEEE standard glossary of software engineering defines metrics as 'a quantitative measure of the degree to which a system, component, or process possesses a given attribute' [14]. In software metrics the terms, "measure, measurement, and metrics" are often used interchangeably, but, there are differences between them. In software engineering, 'measurement is the process by which numbers or symbols are assigned to attributes of entities in the real world in such a way as to describe them according to clearly defined rules [6]. The measurement is act of determining a measure. In the software engineering, 'a measure is the result of the measurement process, so it is the assignment of a value to an entity with the goal of characterizing a specified attribute [11, 15]. In software engineering context, a measure provides quantitative indication of the extent, amount, dimension and capacity of some attributes of a product or process. Therefore, a measure is not just a value, but it is a function that associates a value with an entity. These values cannot be used for arithmetic, algebraic or geometric manipulations, but they can be used to obtain useful results through a variety of statistical techniques. Because of the distinction between internal and external attributes, measures for internal attributes of an entity can be computed based only on the knowledge of the entity, while measures for external attributes also require the knowledge of the context of the entity. Not all functions that associate a value with an entity are sensible measures for an attribute of that entity. Measures that make sense on an intuitive level are an obvious precondition in every measurement application. The measures are used in many scientific fields especially in the hard sciences they are usually taken for granted.

That is, they are considered naturally valid for the attributes they claim to measure and the measures are used in industry applications and everyday productivity activities of hard industry in order to finish their products. However, still this is not the case of software product development industries and organizations. In software engineering, the software metrics have been used for over four decades to assess or predict properties of software attributes, but success has been limited by factors such as lack of sound models, the difficulty of conducting controlled repeatable experiments in commercial context, and the inability to compare data obtained by different researchers. Presently, software metrics is an essential term and topic in software measurement, but not yet well-formed and as well as not matured. Further, there is no true international standard and method for software metrics. Section II explains program keyword software metrics. In Section III, procedure for program keyword metrics has been discussed in detail. In Section IV, results of keyword metrics are analysed and conclusion includes future directions of the research.

II. PROGRAM KEYWORD SOFTWARE METRICS FOR SOFTWARE MEASUREMENT

The software measurement is the process by which numbers or symbols are assigned to attributes of entities in the real world in such a way as to characterize the attributes by clearly defined rules and scales. Further, software metrics are used for three basic activities of software engineering i.e., understanding, control and improvement [6]. The software metrics are recognized as popular in some companies because they are helpful in decision-making on a process and product and the useful software metric indicates whether an organization is achieving software goals. The software metrics proposed by eminent researchers are called C-K metrics [4, 17], MOOD metrics [1, 8, 19], L-K metrics [12, 14], QMOOD metrics [3] for object-oriented design, Halstead metrics [7] and McCabe's metrics [13] for traditional software measurement. A new kind of software metrics called Program Keyword Metrics has been introduced (Figure 1) for software measurement and named Program Keyword Vocabulary and Program Each Keyword Length.

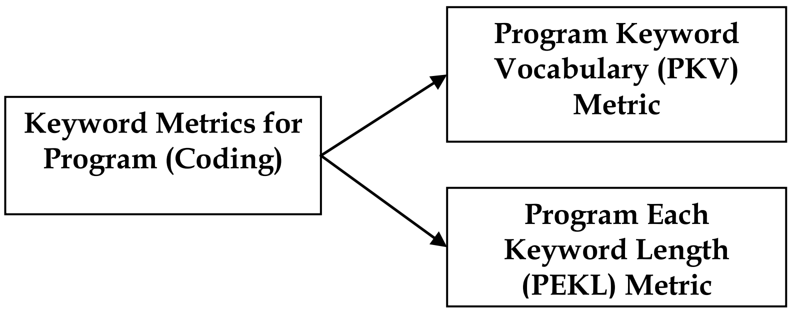
Keyword Metrics for Program (Coding)
Program Keyword Vocabulary (PKV) Metric
Program Each Keyword Length (PEKL) Metric

Figure 1. New Kind of Keyword Software Metrics for Program Measure

In order to judge the usage of keywords in the program in the coding phase of software development, it is desirable to have a figure of merit for the usage of keyword; hence, Program Keyword Vocabulary (PKV) metric is proposed here to calculate the Program Keyword Vocabulary. A large value of PKV shows that ① the program has all keywords and ② the program uses most of the features of a particular language.

Let AT_i denote Appeared keyword Tokens of the ith Keyword and 'TK' denotes the Total number of Keywords (TK) in the programming language. Then the Program Keyword Vocabulary (PKV) of the program can be defined as:

$$PKV = \frac{\sum_{i=1}^{TK} AT_i}{TK} * 100$$

Here, AT = 1 if a Keyword has occurred
= 0 otherwise

In order to find the usage of each keyword in the program during coding phase of the software, it is desirable to have a figure of merit for each keyword. This can be used to find the abnormal usage of keywords in the program. A large value of PEKL shows that ① the particular keyword functionality is high and ② the particular keyword is used to the maximum than other keywords.

Let TEKL= Total number of Each Keyword Length and TKL=Total Keyword Length of the program. The TEKL can be calculated based on number of times each keyword is used in the program. The TKL can be calculated as total number of keyword tokens appeared in the program. Then the Program Each Keyword Length (PEKL) metric can be defined as:

$$PEKL = \frac{TEKL}{TKL} * 100$$

III. PROCEDURE FOR PROGRAM KEYWORD SOFTWARE METRICS

To measure the abnormal usage of keywords in the program, it is required to measure the keywords by metrics and hence the PKM metrics are suggested. This section proposes the procedure for keyword metrics for effective execution of metrics in software measurement. The procedure for keyword metrics and usage of the Program Keyword Vocabulary (PKV) and Program Each Keyword Length (PEKL) metrics are given here. The procedure based metrics system [16, 18, 20] has been introduced and it leads to easy execution of any type of metrics (Figure 2). The execution of each keyword metrics steps is detailed below:

Execution of Step 1: Select a source code to measure the keyword metrics. Normally, the formal program is desirable for measuring the PKV and PEKL metrics.

Execution of Step 2: Find all keyword tokens in the program to measure the utilization of keywords. The collection of tokens is different for different languages.

Step 1: Select a source code to measure the keyword metrics.
Step 2: Find and collect all the keyword tokens in the program.
Step 3: Obtained keyword tokens are tabulated for the entire program for easy usage and calculations of metrics values.
Step 4: Calculate the Program Keyword Vocabulary (PKV) using the defined metrics formula. Let AT_i denote Appeared keyword Tokens of the i^{th} Keyword and 'TK' denotes the Total number of keyword Tokens in the programming language. Then Program Keyword Vocabulary (PKV) of the program can be defined as:

$$PKV = \frac{\sum_{i=1}^{TK} AT_i}{TK} * 100$$

AT = 1 if Keyword token occurred ;

= 0 otherwise;

Step 5: Obtain Program Each Keyword Length (PEKL) using the defined metric formula for PEKL as shown below: Let TEKL= Total number of Each Keyword Length, TKL=Total Keyword Length of the program.

$$PEKL = \frac{TEKL}{TKL} * 100$$

Step 6: Closely examine the PKV and PEKL values of the program. If PKV is 100% in the program then it clearly shows that all token appeared from language. If program have a highest value of PEKL of each keyword then particular keyword has high functionality and modify if necessary.

Figure 2. Procedure Based Metrics System for Keyword Metrics

Execution of Step 5: Calculate Program Each Keyword Length (PEKL) using the defined metric formula. Find the keyword usage using the obtained metric values and check the range of values as shown in Table 1. The obtainable conclusions from keyword metric values are given in Table 2.

Table 1. Ranges Values for Keyword Metrics

Keyword Metric	Range Values (%)
PKV	**0 to 100**
PEKL	**0 to 100**

Table 2. Obtainable Conclusions of Keyword Metrics

Metrics	Obtainable Conclusions
PKV	If PKV is 100%, it shows that the program uses all keyword tokens.
PEKL	If PEKL is 100%, it shows that the corresponding keyword is used to a maximum extent in program.

A good program code will be tested using the keyword metric. If a program has abnormal values for PKV and PEKL metrics then the program needs to be revised and improved. Further, evaluating the keywords values based on the metric values which are defined by the coding experts using domain environments and applications.

IV. EMPIRICAL ANALYSIS RESULTS OF KEYWORD SOFTWARE MEASUREMENT

The proposed keyword metrics are computed for four different projects and the projects are developed in C++ programming language. The Banking System is software that helps to maintain data related to customers and performs the typical banking transactions. The menu based calculation system software aims at performing different types of calculations including normal and scientific calculations. The projects are developed in C++ language and will be referred herein as Project 1, and Project 2 [2]. The proposed metrics are also applied on another two projects referred here as Project 3 and Project 4. The project 3 is developed for Supermarket Billing System using C++ programming. Further, keyword metrics are also applied for another project called Project 4 that is developed for Library called Library Management System using C++ programming language.

The descriptive statistics (Min, Max, Mean, and Std.) calculated for the keyword metrics [5] for project 1, project 2, project 3 and project 4 are presented analysed and the following observations are made. The Table 3 provides descriptive statistics for the PKV metric distributions of Project 1, 2, 3 and 4. Figure 3 shows the distributions of analysed PKV keyword metrics of projects. The PKV metric accurately gives the keyword vocabulary used in the program. The PKV results eliminate the ambiguity criticism of Halsted metrics. The “accuracy on results” in software measurement is achieved i.e., ambiguity is not raised and accurate results are obtainable.

Table 3. Descriptive Statistics for PKV Keyword Metric

Projects	**PKV Metric Percentage**
Project 1	26.9%
Project 2	23.8%
Project 3	25.4%
Project 4	19.5%

The PKV value for Project 1 is 26.9%, Project 2 is 23.8%, Project 3 is 25.4% and Project 4 is 19.5% and it is shown in Table 3. These values that are less than 30% reveal that there in normal usage of keywords in the projects. The Project 4 has less PKV value among these four projects i.e., 19.5%. This analysis is helpful in testing the features of the programming languages incorporation in the programs of the Project.

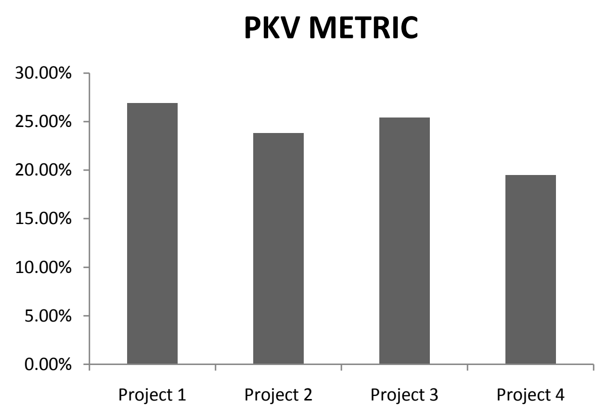

Figure 3. PKV Metric of Projects 1, 2, 3, and 4

The Table 4 provides descriptive statistics for the PEKL metric distributions of Projects 1, 2, 3 and 4. Figure 4 shows the distribution of analysed PEKL keyword metrics of all four projects.

The PEKL is high for some specific keywords for all the projects and it shows that the particular keyword features incorporated in software program is high while looking at the maximum usage of keywords. The “int” keyword occupies 16.5% of total keywords for project 1, the “double” keyword occupies 34.4% on project 2, the “char” keyword occupies 15.7% on project 3, and “char” keyword occupies 28.1% on project 4. The values for PKL and PEKL are generally considered to be good in software coding and these metrics are easy to measure. PKV measures the complexity of the software by counting the number of keywords in the program coding. The minimum value of PEKL observed for the projects are: “private” keyword occupies 0.3% in project 1, “default and void” keyword occupies 0.5% in project 2, “case, const and public” keyword occupies 1.1% in project 3, and “char” keyword occupies 1.3% in project 4.

Table 4. Descriptive Statistics for PEKL Keyword Metric

	Min	Max	Mean	Median	STD.
Project 1	0.3%	16.5%	5.9%	5.3%	5.1%
Project 2	0.5%	34.4%	6.6%	2.7%	8.7%
Project 3	1.1%	15.7%	6.2%	4.5%	4.9%
Project 4	1.3%	28.1%	8.3%	8.5%	8.1%

Among the four projects (1, 2, 3, and 4), the PEKL of Project 2 has the maximum value of 34.4% (double) and PEKL of Project 3 has the least value of 0.3%. The empirical studies of four projects suggest that programs with more PKV must be thoroughly checked during testing because they incorporate all the features. The metrics of all four projects show that maximum keywords are not used in their programs. This analysis allows the project team to gather information about the cause of “normal” and “standard” keyword usage during program development. The keyword metrics is given to the project team or decision-making officials for vocabulary of “unwanted” keyword usage.

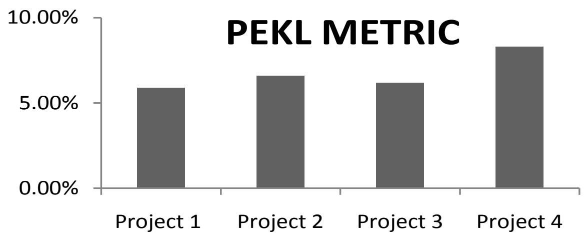

Figure 4. PEKL Metric of Projects 1, 2, 3, and 4

The proposed program keyword metrics has the following achievements and improvements over previous metrics available in literature: ①The keyword metrics are unambiguous and accurate in program/coding measurement. ②The keyword metrics values can be obtained easily and faster. ③The keyword metrics support making process in coding phase. ④The keyword metrics support in solving software development issues. ⑤The keyword metrics are attractive within the metrics researchers and programmers. ⑥The keyword metrics are useful in identifying the complexity of projects. ⑦The keyword metrics gives a new direction for research. ⑧The keyword metrics are used to project the quality of the program. ⑥These metrics can be used for small, medium and large software. ⑩The metrics values are easily obtainable hence student can check their software.

V. CONCLUSION

All the engineering systems are using the measure and measurement systems in day-to-day activities for the production of quality products. Measurement system is an important requirement to run any organization effectively. In case of software engineering, most of the organization produces their products without perfect measurement system. The software metrics which had been proposed till this date for measurement are without proper execution and perfect results. In order to improve the software development process and product, the software measure and metrics and procedure for execution are required. This research paper introduced a new kind of keyword metrics for source code that improves the quality of source code assessment in software engineering. This research can be further extended to design new program keyword metrics based on volume and difficulty.

REFERENCES

[1] F.B. Abreu and W. Melo, “Evaluating the Impact of Object-Oriented Design on Software Quality,” Proceedings of the 3rd International Software Metrics Symposium, IEEE, Berlin, Germany, 1996.

[2] E. Balagurusamy, E., 2010, Object-Oriented Programming with C++, Tata McGraw-Hill Edition, India, 2010.

[3] J. Bansiya and C.G. Davis, “Hierarchical Model for Object-Oriented Design Quality Development,” IEEE Transactions on Software Engineering, Vol. 28, No. 1, pp. 4-17, 2002.

[4] S.R Chidamber and C.F. Kemerer., “A Metrics Suite for Object-Oriented Design,” IEEE Transactions on Software Engineering, Vol. 20, No. 6,June, pp. 476-493,1994.

[5] T. Devi and K.P. Srinivasan., “Statistical Techniques in Software Quality Measurement and Metrics”, Proceeding of UGC Funded National Conference on Recent Advances in Statistics and Computer Applications, Bharathiar University, Coimbatore, 2010.

[6] N.E. Fenton and S.L. Pfleeger., Software Metrics: A Rigorous and Practical Approach, Thomson Asia, 2004.

[7] M.H. Halstead., Elements of Software Science, Elsevier, 1977.

[8] R. Harrison, S.J. Counsel and R.V. Nithi., "An Evaluation of the MOOD Set of Object-Oriented Software Metrics," IEEE Transactions on Software Engineering, Vol.24, No.6, pp.491-496, 1998.

[9] C. Jones., Applied Software Measurement: Global Analysis of Productivity and Quality, Tata McGraw-Hill Edition, India, 2008.

[10] S.H. Kan., Metrics and Models in Software Quality Engineering, Pearson Education, India, 2006.

[11] D.S. Kushwaha and A.K. Misra, "Cognitive Complexity Metrics and its Impact on Software Reliability Based on Cognitive Software Development Model," ACM SIGSOFT Software Engineering Notes, Vol. 31 No. 2, pp. 1-6, March, 2006

[12] M. Lorenz and J. Kidd., Object-Oriented Software Metrics, Prentice Hall, 1994.

[13] T.J. McCabe., "A Complexity Measure," IEEE Transactions on Software Engineering, Vol. se-2, No. 4, pp. 308-320,1976.

[14] R.S. Pressman, Software Engineering a Practitioner's Approach, 5th Edition, McGraw Hill, India, 2001.

[15] S. Purao and V. Vaishnavi, "Product Metrics for Object-Oriented Systems," ACM Computing Surveys, Vol. 35, No. 2, June, pp. 191–221, 2003.

[16] K.P. Srinivasan and T. Devi., "Procedure for Selection of an Efficient Object-Orientated Design," Proceeding of UGC Funded National Conference on Recent trends in Software Engineering, Sullamussalam Science College, Mallapuram, Kerala, 2009. (Best Paper of the Conference).

[17] K.P. Srinivasan, T. Devi, and V.Thiagarasu., "Analysis of Chidamber - Kemerer Metrics for Object-Orientated Design," Proceeding of National Conference on Emerging trends in Computer Science., Avinasilingam University, Coimbatore, 2009.

[18] K.P. Srinivasan and T. Devi.,"Design and Development of a Procedure to Test the Effectiveness of the Object-Oriented Design," International Journal of Engineering Research and Industrial Applications, Vol. 2, No.VI, 2009.

[19] K.P. Srinivasan and T. Devi., " A Case Study Approach for Application of MOOD Metrics in Object-Oriented Design," Proceedings of the International Conference on Global Computing and Communication, Hindustan University, Chennai, 2009.

[20] K.P. Srinivasan and T. Devi., "Design and Development of a Procedure for new Object Oriented Design Metrics," International Journal of Computer Applications, Vol. 24, No.8, pp.30-35, 2011.

AUTHORS

K.P. Srinivasan received his Master of Computer Applications degree from Bharathiar University, Coimbatore, India in 1993 and M.Phil degree in Computer Science from the Bharathiar University, Coimbatore, India in 2001. Presently, he is working as an Associate Professor of Computer Science in C.B.M. College, Kovaipudur, Coimbatore under Bharathiar University, Coimbatore, India since 1997. He is doing his research work in Software Engineering. He has published five conference papers and four journal papers. His current research interests are in the areas of Software Engineering and Object-Oriented Systems.

Dr T. Devi received Master of Computer Applications Degree from P.S.G. College of Technology, Coimbatore, India in 1987 and the Ph.D. degree from the University of Warwick, United Kingdom in 1998. Presently, she is working as an Associate Professor and Head of the Department of Computer Applications, School of Computer Science Engineering, Bharathiar University, Coimbatore, India. Prior to joining Bharathiar University, she was an Associate Professor in Indian Institute of Foreign Trade, New Delhi, India. She has contributed more than 140 papers in various Journals and Conferences. Her current research interests are in the areas of Software Engineering and Concurrent Engineering.