Object-Oriented Software Maintainability Measurement in the past Decade

Bajeh Amos Orenyi, *Member*, *IEEE Computer Society*, Shuib Basri, and Low Tan Jung
Computer and Information Science Department
Universiti Teknologi PETRONAS
Bandar Seri Iskandar, 31750 Tronoh, Perak Darul Ridzuan, Malaysia
bajeh_amos@yahoo.com, shuib_basri@petronas.com.my, lowtanjung@petronas.com.my

***Abstract*—The economic implication of the maintenance phase of software development process can be managed by using software metrics to build-in maintainability quality into software products. Several software quality models and maintainability measurement models/methods have been proposed in the literatures; still the evaluation of Object-Oriented Software (OOS) maintainability has not attained uniformity and consistency in measurement results. This paper reviews existing studies in OOS maintainability measurement within the past decade. Results show that elements of subjectivity in the measurement methods threaten the attainment of uniformity, consistency and potential wide acceptability of the identified models in this review.**

Keywords-measurement; maintainability; Object-Oriented Software; quality models; subjectivity

I. INTRODUCTION

In engineering disciplines, measurement is one of the fundamental aspect without which there is no engineering per se. It has been one of the approaches used to control and manage the engineering of quality into products. Several authors have highlighted the fundamental importance of measurement in Software Engineering (SE)[1-4] . Basically, this is based on the assertion that, what you can measure, you can control and what you can control, you can manage.

The maintenance phase of the Software Development Life Cycle (SDLC) is the most costly of all the phases in terms of budget and programmers' effort [2, 5-7]. Thus developing maintainable Software products has economic significance of saving cost in terms of budget and effort put into software maintenance. The use of software metrics facilitates the process of developing highly maintainable software products. Software metrics equip developers with necessary information to make reasonable decisions that will reduce design complexities as much as possible during development. The use of metrics could uncover potential design errors and complexities during the design phase before they become more difficult and costlier to identify and correct at the implementation/testing phases or when the software is in operation.

Object-Oriented Software (OOS) have been arguably attributed to be more maintainable than software developed using the traditional method. Several Object-Oriented Metrics (OOM) for measuring the structural properties of OOS abound in the literatures [8-11] and several software quality models have been proposed for the evaluation of software quality characteristics such as maintainability [12-17]

This paper reviews existing studies in the area of OOS maintainability measurement or prediction over the past decade, 2003 - 2012. It covers the measurement of maintainability in terms of the internal and structural attributes/properties of OOS products (design and source code), and not maintainability effort estimation (usually quoted in man-hour, man-month or person-month) which involves the estimation of maintainers' effort while carrying out maintenance activity.

This paper is organized as follows: section II discuses the method used to access available studies within the period under consideration. Section III is the review proper, where data extraction from the collected studies is discussed. Section IV discusses the results of the review. Conclusion, recommendations and future work are in section V.

II. METHODOLOGY

A search string (stated below) was constructed from the keywords identified in some of the literatures collected before the commencement of this survey. Also the following online database, search engines and journals were used as the resources for this survey.

Search String: (Object Oriented Software OR OO Software OR OOS OR Object Oriented Application OR Object-Oriented Application OR OO Application OR OO system OR Object Oriented Software Application OR Object-Oriented Software Application OR OO Software Application OR OOS Application) AND (Software Quality Model OR Quality Model OR Quality Framework OR OO Metrics OR OOM) AND (subcharacteristics OR sub-characteristics OR Attributes OR sub-attributes OR subattributes) AND (measurement OR Estimation OR Prediction OR measure OR Estimate OR Predict) AND (Maintainability OR Analyzability OR Analysability OR Changeability OR Testability OR Complexity OR Stability OR Maintainability Compliance OR Understandability)

Resource for Publications:

i. IEEE Xplore
ii. Springer
iii. ACM DL
iv. Scopus
v. ProQuest

Other resources searched are individual journals and Conference proceedings:

978-0-7695-4959-0/13 $25.00 © 2013 IEEE
DOI 10.1109/ACSAT.2012.54

i. Journal of Object Technology (JOT): http://www.jot.fm
ii. Journal of Engineering Trends in Computing and Information Science (http://www.cisjournal.org)
iii. Int'l Journal of Computer Application (http://www.ijcaonline.org)
iv. Journal of Computer Science
v. Int'l Journal of Computer Science and Security
vi. Journal of Software Maintenance and Evolution : Research and Practice
vii. Int'l Journal of Computer Science and Information Technology
viii. Journal of Computing
ix. World Academy of Science, Engineering and Technology
x. Int'l Conference on Advance Computing and Communications

The search string was modified for some of the online search engines to fit their requirement so that they can be searched. The advance search of the search engines was used to specify for publications within the period in view (2003-2012) - a total of 15,345 publications were returned. Firstly, the titles and abstract of the publications were use to identify relevant publications that discuss OOS maintainability – 76 publications were identified; secondly, the selected publications were further pruned down by reading the entire article and excluding those that discuss maintainability in terms of effort estimation quoted in man-hours, man-month or person-month- this yielded 36 relevant publications for the survey.

III. THE LITERATURE SURVEY

The pertinent concern of this survey is to investigate the type of measurement method (objective or subjective) used in the literature, and the potential acceptability of the existing models or methods. This is spurred by the submission of Abreu and Carapuca [18]; and Genero et al.[19]. Abreu and Carapuca stated that "Different people at different times or places should yield the same values when measuring the same system. Subjectivity makes metrics comparison throughout software industry an impossible mission. Subjective ratings....., are copious in the metric literatures. That is undoubtedly one of the reasons that lead to metrics suspicion among software practitioners and the Computer Science community in general". Genero et al. pointed out that there is the need for objective and valid measures for the evaluation and improvement of product quality.

Table I is the tabulation of the data extracted from the reviewed publication.

IV. RESULTS AND DISCUSSION

The result of the review shows that several OOS maintainability measurement models and methods have been proposed within the period in view. Fig. 1 depicts in a bar chart the distribution of the studies among the various methods of prediction identified in the publications. It shows that Regression Analysis (RA) methods are the most employed method of aggregation/evaluation used in the decade. The methods used in PS36 [20] are grouped as RA because of their close similarity with RA. Substantial differences are noted in the way and manners the OOM (base/derived) are grouped together (third column of Table I) as the metrics that measure the attributes that influence the maintainability of OOS. The use of Confidence Assessment of models/methods to measure their reliability was, to the best of our knowledge, used for the first time in OOS metrics in PS31 [21]. New aggregation approaches to maintainability evaluation are also seen in PS35 [22] and PS36 [20]

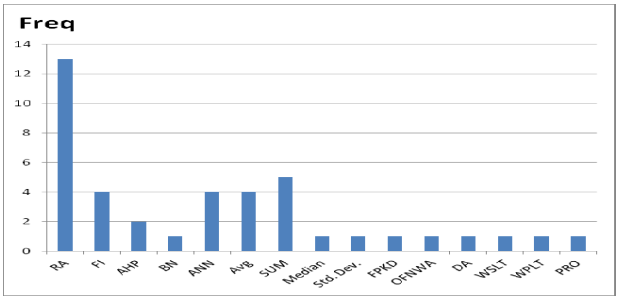

Figure 1. Aggregation Method
(Acronyms are defined under Table I)

Within the period in view, RA method dominates in the literatures and existing Software Quality Models (SQM) such as ISO/IEC 9126 [12], McCall's [13], etc are scarcely used in the development of maintainability models. Majority of the studies, except few, used the existing OOM without a critical review and adaptation of these metrics before they are used to develop the models. This makes the developed models to inherit the inconsistencies and ambiguities observed in the OOM [23, 24].

The OOM used in the models/methods are mostly measured objectively, but the methods used for aggregating metrics or predicting maintainability from the metrics have some element of subjectivity. Though the models developed using the different types of regression analysis have their base/derived metric measured objectively, they are subjective to the peculiar characteristics of the empirical/historical data used to develop the regression equations. The coefficients of the equations are determined from and validated with the data collected from some selected software. The values of the coefficients of the models partly determine the results from such models. Thus, results derived from such models are subjective to the coefficients and by extension, to the characteristics of the data used to determine the coefficients. In other words, different software products have different structural properties and complexities (as shown by descriptive statistic in some studies[25]); thus using a model developed from a set of software "A" to evaluated another software "B"(that is not used in the development of the model) will yield result that will be partly determined by the properties of "A".

Hence, the value of "B" will be subjective to "A". This form of subjectivity, to the best of our knowledge, has not been identified in Software metrics. This could be one of the fundamental reasons why several forms of regression models yield different results when applied to the same software [26]. Also, this is a potential threat to the wide applicability and acceptability of the various regression models.

TABLE I. SURVEY RESULT

Primary Study ID	Software Quality Model	Subcharacteristics & base/derived metrics	Measurement Method	Aggregation Method
PS1 [27]	None	DIT, MPC, RFC, LCOM, DAC, WMC, NOM, SIZE1 and SIZE2	Objective	RA
PS2 [28]	None	ALV, ALS, ACC, CR	Objective	FI
PS3 [29]	ISO/IEC 9126	WMC, NPM, DAM, CBO, NOP, DIT, NOC, LCOM and Ca.	Partly subjective	AHP
PS4 [30]	None	WMC, DIT,RFC, NOC, LCOM, MPC, DAC, NOM, SIZE1, SIZE2 and CHANGE	Objective	RA
PS5 [31]	None	RSC using CR, DQ using Fog Index, UoS using Laitnen tool, & ACC	Partly subjective	FI
PS6 [32]	None	1. Size - NumCls, NumCls_tc, NumOpcls, 2. Complexity - No. of relationship between classes, interfaces 3. Cohesion - Relational & Connected component; 4. Ease of change (Instability); 5. Abstractness; 6. Nesting level; 7. Coupling (import)	Partly subjective	RA
PS7 [33]	None	CK Metrics: WMC, LCOM, CBO, RFC, DIT and NOC	Objective	PRO
PS8 [34]	None	CK Metrics Suite: DIT, WMC, NOC, LCOM, RFC and CBO	Objective	RA
PS9 [35]	None	1. CK metrics: DIT, NOC, MPC, RFC, LCOM, WMC and 2. L&H metrics: DAC, NOM, SIZE1 and SIZE2, CHANGE	Objective	BN
PS10 [36]	None	64 base metrics automatically measured from source code	Partly Subjective	FI
PS11 [37]	None	1. Structural measures - LOC, NOC, WMC, OMMIC, TCC; 2. Experts Assessment - Choice of classes, Design adapted to systems, 3-layer architecture, Good use of components, Encapsulation, Inheritance, Good use of class library, Simplicity, Naming, comments, Appropriate technical platform.	Partly Subjective	N/A
PS12[38]	None	ACLOC, AHF, AMLOC, AVPATH, CDENS, COF, DIT, LCOM, MHF, MIF, N, n, WMC, NCLASS, NMETH, PPPC, RFC, SLOC	Objective	RA
PS13[26]	1. ISO/IEC 9126 2. Regression Model – Multivariate Linear Regression	1. ISO/IEC 9126 - Base/derived metrics: NOC, NAM, NOM, WMC, RFC, DIT, LOC, CBO, DAC, LD, MPC, LCOM, ILCOM, TCC, LOC, LEN and CYC 2. Regression Model – Base/derived metrics: M_{SIZE}, M_{WMC}, M_{CBO}, M_{DIT},M_{NOC}, M_{CBO}, M_{RFC}, M_{LCOM}, M_{DIT}, M_{NOM}	1. Partly subjective 2. Objective	1. AVG 2. RA
PS14[39]	ISO/IEC 9126	Size and Coupling	Objective	N/A
PS15[40]	None	V, MCC, LOC and perCM - ratio of line of comment to total number of lines.	Objective	RA
PS16[41]	None	LCOM, NOC, DIT, WMC, RFC, DAC, MPC and NOM	Objective	ANN
PS17[17]	ISO/IEC 9126	1. Analysability - volume, duplication, unit size and unit testing 2. Changeability - complexity per unit, duplication 3. Stability - unit testing and 4. Testability - complexity per unit, unit size, unit testing	Partly Subjective	N/A
PS18[42]	None	TNOS, NICMIC, NIMMIC and NIIC	Objective	RA
PS19[43]	None	1. PC1: RFC, LCOM, DAC, WMC, NOM, SIZE1, SIZE2 2. PC2: MPC 3. PC3: DIT	Objective	ANN
PS20[44]	None	1. PC1: RFC, LCOM, WMC, NOM & SIZE2 2. PC2: DAC & SIZE1 3. PC3: DIT & NOC	Objective	ANN
PS21[16]	Understandability & Modifiability	1. Understandability - NC and NGenH 2. Modifiability - NC,Ngen, NGenH, NAggH, MaxDIT,	Objective	RA
PS22[45]	Maintainability:- Documentation, Source code & Implementation	1. Software Development – organisation property 2. Documentation - description clarity, traceability, clarity, modular, consistency) 3. Source code - simplicity, scalability, testability, modular, traceability 4. Implementation - customary, simplicity, testability	Subjective	FI
PS23[46]	As in PS22	1. Understandability: a) PC FNAS (NASSOC and MAXDIT)	Objective	RA

		b) Measures: MaxDIT, NA, NDep, NASSOC 2. Modifiability: a) Principal Component FNAS (NASSOC and MAXDIT) b)Measures: MAXDIT, NM, NC & NGenH		
PS24[47]	None	LCOM, WMC, RFC, DAC and NOM	Objective	RA
PS25[48]	None	TCC, NOC, WMC, DIT, LOC, OMMIC, OMMEC, God method, Data class and 24 other metrics subjectively defined	Partly Subjective	N/A
PS26[49]	ISO/IEC 9126	Doc quality, Programming quality, System Requirement, Personal Resource and Process Management problems	Subjective	AVG
PS27[50]	None	3 PC from DIT, NOC, CBO, RFC, IC, CBM, WMC, NOMA	Objective	ANN
PS28[25]	None	WMC, DIT, NOC, RFC, and LCOM, MPC, DAC, NOM, and SIZE2; and the traditional SIZE1 (LOC)	Objective	RA
PS29[51]	ISO/IEC 9126	Number of Statement, WMC, NOC, DAM, CBO, LCOM, No. of Messages, NPM, Measure of Aggregation & DIT	Partly Subjective	AHP
PS30[52]	ISO/IEC 9126	All subcharacteristics measured from: LOC and MCC except Stability measured as CBO	Objective	Sum
PS31[21]	None	WMC1, OMMIC, OMMEC, NOC, DIT, TCC	Partly Subjective	Mean, Median, Std Dev., Sum
PS32[53]	As in PS22	NC, ANAUW, ANMUW, ANAsso, NaggH, MaxHAgg, NGenH, MaxDIT, NOS, WMBO, ANRM, ANDM, ANET, ANCM,	Partly Subjective	1. Discriminant Analysis 2. Weighted-Score-Level Technique 3. Weighted-Predicted-Level Technique
PS33[54]	None	Interaction Level(No. of Interactions, sum of strength of interaction); Interface size(No. of parameters + sum of size of parameters); Operation Argument Complexity(sum of the size of each parameter)	Partly Subjective	Sum
PS34[55]	None	LOC, Comments, Classes, WMC, OMMIC, OMMEC, NOC, DIT, TCC	Objective	Sum
PS35[22]	None	NC, NA, NM, NAssoc, NAgg, NGen, NAggH, NGenH, MaxDIT	Objective	Fuzzy Prototype Knowledge Discovery
PS36[20]	Maturity, Source Code and Documentation	Over 32 based/derives metrics are enlisted	Partly Subjective	Ordered Fuzzy Number Weighted Averaging

M_{DIT}/DIT (Depth of Inheritance Tree) , MPC(Message Passing Coupling), M_{RFC}/RFC(Response for a Class), M_{LCOM}/LCOM(Lack of Cohesion in Method), DAC(Data Abstraction Coupling), M_{WMC}/WMC(Weighted Method per Class), M_{NOM}/NOM(No. of Method), M_{SIZE}/SIZE1(No. of LOC)& SIZE2(No. of attribute & property), ALV(Average No. of Live Variable), ALS(Average Life Span of Variable), ACC(Average Cyclomatic Complexity), CR(Comment Ratio), NPM(No. of Public Method), DAM(Data Access Metrics), M_{CBO}/CBO(Coupling Between Object), NOP(No. of Polymorphic Method), M_{NOC}/NOC(No. of Children), Ca(Afferent Coupling), RSC(Readability of Source Code), DQ(Document Quality), UoS(Understandability of Software), CHANGE(No. of Lines changed), TCC(Tight Class Cohesion), Average Class size(ACLOC), Attribute Hidding Factor(AHF), Average Method Size(AMLOC), Average Depth of Path(AVPATH), Control Density(CDENS), Coupling Factor(COF), Method Hiding Factor(MHF), Method Inheritance Factor(MIF), Program Length(N), Program vocabulary(n), Number of Class(NCLASS), Number of Method(NMETH), Percentage Public/Protected member(PPPC), Source lines of code(SLOC), Number of class(NC), No. of Attributes(NA), No. of Methods(NM), No. of Association(NAssoc), No. of Aggregation(NAgg), No. of Dependency(NDep), No. of Generalisation(NGen), No. of Generalization Hierarchy(NGenH), Maximum DIT(MaxDIT), No. of Association Vs Classes(NassVC), No. of Dependency Vs Classes(NDepVC), No. of Aggregation Hierarchy(NAggH), No. of Aggregation Vs Classes(NAggVC), No. of GeneralizationVsClasses(NGenVC), NAM, LD, ILCOM, LEN, CYC, MCC(McCabe's Cyclomatic Complexity), Percentage line commentd(perCM), TNOS(Total No. of Statement), NICMIC(Non-inheritance Class-Method Import Coupling), NIMMIC(Non-Inheritance Method Method Import Coupling), NIIC(Non-Inheritance Import Coupling), FNAS(),OMMIC(Call to methods in unrelated class), OMMEC(Call from methods in an unrelated class), IC(Inheritance Coupling), NOMA(Number of Object/Memory Allocations), ANAUW(Average No. of Attributes-Unweighted), ANMUW(Average No. of Methods-Unweighted), ANAsso(Average No. of Association), NOS(No. of Scenarios), WMBO(Weighted Messages Between Objects), ANRM(Average No. of Return Messages), ANDM(Average No. of Directly dispatched Messages), ANET(Average No. of Elements), ANCM(Average No. of Condition Messages), CM(Line of Comment), SP(Average no. of Statements between two successive reference to the same variable), NumCls(No. of Class in package), NumCls_tc(No. of class in package, subpackage, and so on), NumOpcls(No. of Operations in the classes of the package), V(Halstead's Volume), perCM(Percentage line commented), TNOS(Total No. of Stmt), NICMIC(Non-inheritance Class-Method Import Coupling), NIMMIC(Non-Inheritance Method Import Coupling), NIIC(Non-Inheritance Import Coupling) PC(Principal Component), RA(Regression Analysis), FI(Fuzzy Inference), Sum(Summation), AHP(Analytic Hierarchy Process), PRO(Proportionality), BN(Bayesian Network), AVG(Average), ANN(Artificial Neural Network), FPKD(Fuzzy Prototype Knowledge Discovery), OFNWA(Ordered Fuzzy Number Weighted Averaging) DA(Discriminant Analysis), WSLT(Weighted-Score-Level Technique), WPLT(Weighted-Predicted-Level Technique)

V. CONCLUSION AND FUTURE WORK

This review has shown that in the recent times RA methods have been predominantly used in the development of OOS maintainability models and we have identified the form of subjectivity associated with this method (and other methods) in section IV. This subjectivity potentially threatens the uniformity of measurement results across the models and thus threatens the possibility of wide applicability and acceptability of the models because their results will be subjective to the values of the regression equations coefficients determined from the dataset of some collected OOS products. This is also in line with the work of Lincke et al [26] which shows that different existing models yield different results which

lead to different conclusions when used on the same software. Thus, there is the need to develop maintainability measurement model that will use objective measurement method to yield consistent result anytime anywhere and by anybody (in this case software developer). The model should possess the potential of wide applicability and acceptability by not subjecting its result to datasets from some selected software as in the RA models. Such model will equip developers with objective information to guide them in engineering quality into OOS product.

Recommendations: due to the elusiveness of quality attributes such as maintainability, the use of SQM (e.g. ISO 9126-1) should be encouraged; complete reliance on historical data for the development of maintainability models should be minimized, this will reduce the form of subjectivity identified with RA models in this review; and root-cause traceability needs to be emphasized in future models in order to facilitate the engineering of quality into OOS products during development.

This work is part of an ongoing research with the objective of developing an ISO/IEC 9126 standard-based maintainability measurement model/scheme that will be objective and have potential of wide applicability. The above recommendations are some of the issues the research will address. The choice of ISO/IEC 9126 standard is informed by its status as an international standard which is developed in consensus. This standard had been harmonized with ISO/IEC 14598 to form the SQuaRE 2005 Software quality model [56].

REFERENCES

[1] F. T. Sheldon, K. Jerath and H. Chung, "Metrics for maintainability of class inheritance hierarchies," *Journal of Software Maintenance and Evolution: Research and Practice,* vol. 14, pp. 147-160, 2002.

[2] R. S. Pressman, *Software Engineering: A practitioner's Approach.* New York, USA: McGraw Hill, 2005.

[3] L. Badri, M. Badri and A. B. Gueye, "Revisiting Class Cohesion: An empirical investigation on several systems," *Journal of Object Technology,* vol. 7, pp. 55-75, 2008.

[4] M. Genero, M. Piattini, E. Manso and G. Cantone, "Building UML class diagram maintainability prediction models based on early metrics," in *Software Metrics Symposium, 2003. Proceedings. Ninth International,* 2003, pp. 263-275.

[5] K. Aggarwal, Y. Singh, P. Chandra and M. Puri, "Measurement of software maintainability using a fuzzy model," *Journal of Computer Sciences,* vol. 1, pp. 538-542, 2005.

[6] B. P. Lientz, E. B. Swanson and G. E. Tompkins, "Characteristics of application software maintenance," *Commun ACM,* vol. 21, pp. 466-471, 1978.

[7] G. Parikh, "The Guide to Software Maintenance," 1982.

[8] S. R. Chidamber and C. F. Kemerer, "A metrics suite for object oriented design," *Software Engineering, IEEE Transactions on,* vol. 20, pp. 476-493, 1994.

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

[10] W. Li and S. Henry, "Object-oriented metrics that predict maintainability," *J. Syst. Software,* vol. 23, pp. 111-122, 1993.

[11] J. Bansiya and C. G. Davis, "A hierarchical model for object-oriented design quality assessment," *Software Engineering, IEEE Transactions on,* vol. 28, pp. 4-17, 2002.

[12] International Organization for Standardization/International Electrotechnical Commission, "ISO/IEC 9126-1 Standard, Software Engineering, Product Quality, Part 1: Quality Model," *Author, Geneva,* 2001.

[13] J. A. McCall, P. K. Richards, G. F. Walters, Rome Air Development Center and United States. Air Force. Systems Command. Electronic Systems Division, *Factors in Software Quality.* Rome Air Development Center, Air Force Systems Command, 1977.

[14] R. G. Dromey, "A model for software product quality," *Software Engineering, IEEE Transactions on,* vol. 21, pp. 146-162, 1995.

[15] B. W. Boehm, J. R. Brown, H. Kaspar, M. Lipow, G. J. MacLeod and M. J. Merrit, "Characteristics of software quality," 1978.

[16] S. Rizvi and R. Khan, "Maintainability Estimation Model for Object-Oriented Software in Design Phase (MEMOOD)," *Journal of Computing,* vol. 2, pp. 26-32, 2010.

[17] I. Heitlager, T. Kuipers and J. Visser, "A practical model for measuring maintainability," in *Quality of Information and Communications Technology, 2007. QUATIC 2007. 6th International Conference on the,* 2007, pp. 30-39.

[18] F. B. Abreu and R. Carapuça, "Object-oriented software engineering: Measuring and controlling the development process," in *Proceedings of the 4th International Conference on Software Quality,* 1994, .

[19] M. Genero, M. Piattini and C. Calero, "A survey of metrics for UML class diagrams," *Journal of Object Technology,* vol. 4, pp. 59-92, 2005.

[20] G. Canfora, L. Cerulo and L. Troiano, "Can fuzzy mathematics enrich the assessment of software maintainability?" in *Proceeding of the First International Workshop on Software Audit and Metrics, Porto, Portugal, April,* 2004, pp. 13-14.

[21] H. Benestad, B. Anda and E. Arisholm, "Assessing software product maintainability based on class-level structural measures," *Product-Focused Software Process Improvement,* pp. 94-111, 2006.

[22] M. Genero, J. Olivas, M. Piattini and F. Romero, "Assessing object-oriented conceptual models maintainability," *Advanced Conceptual Modeling Techniques,* pp. 288-299, 2003.

[23] M. El-Wakil, A. El-Bastawisi, M. Boshra and A. Fahmy, "Object-oriented design quality models a survey and comparison," in *2nd International Conference on Informatics and Systems,* 2004, .

[24] S. K. Dubey and A. Rana, "A Comprehensive Assessment of Object-Oriented Software Systems Using Metrics Approach," *International Journal of Computer Science and Engineering (IJCSE),* pp. 2726-2730, 2010.

[25] M. O. Elish and K. O. Elish, "Application of TreeNet in predicting object-oriented software maintainability: A comparative study," in *Software Maintenance and Reengineering, 2009. CSMR'09. 13th European Conference on,* 2009, pp. 69-78.

[26] R. Lincke, T. Gutzmann and W. Lo we, "Software quality prediction models compared," in *Quality Software (QSIC), 2010 10th International Conference on,* 2010, pp. 82-91.
[27] W. Li-Jin, H. Xin-Xin, N. Zheng-Yuan and K. Wen-hua, "Predicting object-oriented software maintainability using projection pursuit regression," in *Information Science and Engineering (ICISE), 2009 1st International Conference on,* 2009, pp. 3827-3830.
[28] H. Mittal and P. Bhatia, "Software maintainability assessment based on fuzzy logic technique," *ACM SIGSOFT Software Engineering Notes,* vol. 34, pp. 1-5, 2009.
[29] P. Antonellis, D. Antoniou, Y. Kanellopoulos, C. Makris, E. Theodoridis, C. Tjortjis and N. Tsirakis, "A data mining methodology for evaluating maintainability according to ISO/IEC-9126 software engineering–product quality standard," *Special Session on System Quality and Maintainability-SQM2007,* 2007.
[30] Y. Zhou and H. Leung, "Predicting object-oriented software maintainability using multivariate adaptive regression splines," *J. Syst. Software,* vol. 80, pp. 1349-1361, 2007.
[31] Y. Singh, P. K. Bhatia and O. Sangwan, "Predicting software maintenance using fuzzy model," *ACM SIGSOFT Software Engineering Notes,* vol. 34, pp. 1-6, 2009.
[32] K. Kaur and H. Singh, "Determination of maintainability index for object oriented systems," *ACM SIGSOFT Software Engineering Notes,* vol. 36, pp. 1-6, 2011.
[33] S. K. Dubey and A. Rana, "Assessment of maintainability metrics for object-oriented software system," *ACM SIGSOFT Software Engineering Notes,* vol. 36, pp. 1-7, 2011.
[34] T. Nair, S. Aravindh and R. Selvarani, "Design property metrics to maintainability estimation: a virtual method using functional relationship mapping," *ACM SIGSOFT Software Engineering Notes,* vol. 35, pp. 1-6, 2010.
[35] C. Van Koten and A. Gray, "An application of Bayesian network for predicting object-oriented software maintainability," *Information and Software Technology,* vol. 48, pp. 59-67, 2006.
[36] N. J. Pizzi, "Software quality prediction using fuzzy integration: a case study," *Soft Computing-A Fusion of Foundations, Methodologies and Applications,* vol. 12, pp. 67-76, 2008.
[37] B. Anda, "Assessment of software system evolvability," in *Ninth International Workshop on Principles of Software Evolution: In Conjunction with the 6th ESEC/FSE Joint Meeting,* 2007, pp. 71-74.
[38] S. C. Misra, "Modeling design/coding factors that drive maintainability of software systems," *Software Quality Journal,* vol. 13, pp. 297-320, 2005.
[39] F. Losavio, L. Chirinos, A. Matteo, N. Lévy and A. Ramdane-Cherif, "ISO quality standards for measuring architectures," *J. Syst. Software,* vol. 72, pp. 209-223, 2004.
[40] L. G. Thomas, *An Analysis of Software Quality and Maintainability Metrics with an Application to a Longitudinal Study of the Linux Kernel,* 2008.
[41] K. Aggarwal, Y. Singh, A. Kaur and R. Malhotra, "Application of Artificial Neural Network for Predicting Maintainability using Object-Oriented Metrics," *Transactions on Engineering, Computing and Technology,* vol. 15, pp. 285-289, 2006.
[42] M. Dagpinar and J. H. Jahnke, "Predicting maintainability with object-oriented metrics-an empirical comparison," in *Proceedings of the 10th Working Conference on Reverse Engineering (WCRE),* 2003, pp. 155-164.
[43] M. M. T. Thwin and T. S. Quah, "Application of neural networks for estimating software maintainability using object-oriented metrics," in *SEKE,* 2003, pp. 69-73.
[44] Y. Dash, S. K. Dubey and A. Rana, "Maintainability Prediction of Object Oriented Software System by Using Artificial Neural Network Approach," *International Journal of Soft Computing and Engineering (IJSCE),* vol. 2, pp. 420-423, 2012.
[45] H. Jiang, "Fuzzy evaluation on software maintainability based on membership degree transformation new algorithm M (1, 2, 3)," in *Business Intelligence and Financial Engineering, 2009. BIFE'09. International Conference on,* 2009, pp. 79-83.
[46] M. Genero, E. Manso, A. Visaggio, G. Canfora and M. Piattini, "Building measure-based prediction models for UML class diagram maintainability," *Empirical Software Engineering,* vol. 12, pp. 517-549, 2007.
[47] C. Jin and J. A. Liu, "Applications of support vector mathine and unsupervised learning for predicting maintainability using object-oriented metrics," in *Multimedia and Information Technology (MMIT), 2010 Second International Conference on,* 2010, pp. 24-27.
[48] A. F. Yamashita, H. C. Benestad, B. Anda, P. E. Arnstad, D. I. K. Sjoberg and L. Moonen, "Using concept mapping for maintainability assessments," in *Empirical Software Engineering and Measurement, 2009. ESEM 2009. 3rd International Symposium on,* 2009, pp. 378-389.
[49] J. C. Chen and S. J. Huang, "An empirical analysis of the impact of software development problem factors on software maintainability," *J. Syst. Software,* vol. 82, pp. 981-992, 2009.
[50] M. M. T. Thwin and T. S. Quah, "Application of neural networks for software quality prediction using object-oriented metrics," *J. Syst. Software,* vol. 76, pp. 147-156, 2005.
[51] Y. Kanellopoulos, P. Antonellis, D. Antoniou, C. Makris, E. Theodoridis, C. Tjortjis and N. Tsirakis, "Code Quality Evaluation Methodology Using The ISO/IEC 9126 Standard," *Arxiv Preprint arXiv:1007.5117,* 2010.
[52] L. Ping, "A quantitative approach to software maintainability prediction," in *Information Technology and Applications (IFITA), 2010 International Forum on,* 2010, pp. 105-108.
[53] M. Kiewkanya, N. Jindasawat and P. Muenchaisri, "A methodology for constructing maintainability model of object-oriented design," in *Quality Software, 2004. QSIC 2004. Proceedings. Fourth International Conference on,* 2004, pp. 206-213.
[54] R. K. Bandi, V. K. Vaishnavi and D. E. Turk, "Predicting maintenance performance using object-oriented design complexity metrics," *Software Engineering, IEEE Transactions on,* vol. 29, pp. 77-87, 2003.
[55] B. Anda, "Assessing software system maintainability using structural measures and expert assessments," in *Software Maintenance, 2007. ICSM 2007. IEEE International Conference on,* 2007, pp. 204-213.
[56] D. Zubrow, "Measuring software product quality: The ISO 25000 Series and CMMI," *European SEPG,* 2004.