International Journal of Innovative Research in Computer Science & Technology (IJIRCST)
ISSN: 2347-5552, Volume- 8, Issue- 5, September- 2020
https://doi.org/10.21276/ijircst.2020.8.5.4
www.ijircst.org

Measuring Maintainability of Object Oriented Design: A Revisit

Ramesh Kumar, Dr. Abdullah, Abhishek Yadav

ABSTRACT- Maintainability has always been an elusive concept. Software maintainability is an external software quality attributes that estimate the complexity and effort required for maintaining software. The key concern of this review paper will be organized study on maintainability considering in view by its sub factors and metrics implementation of software maintainability keeping in mind to supports the maintenance process and facilitates the formation of improved quality software. In this paper studies accomplish a systematic literature review to have widespread facts of maintainability research and its feature factors and related measurements. Finally study does a comparative analysis on software maintainability models developed by various researchers/area experts including their contribution and limitation. In last our effort is to find the known wide-ranging and complete model or framework for quantifying the maintainability of object oriented software at an initial stage of development life cycle.

KEYWORDS- Modularity, Reusability, Analyzability, Modifiability, Testability, Maintainability, Maintainability Measurement, Object Oriented Design, Software Quality.

I. INTRODUCTION

Highlight Software systems are large, complex beset with maintenance problems, at the same time as users expect high quality product within time and budget [1]. However, it is tough to evaluate and assure software quality. In order to meet the changing requirements of customer or due to many other reasons, software needs to be changed or modified from time to time [10].

Manuscript received August 21, 2020

Ramesh Kumar, M.Tech (P), Department of Computer Science & Engineering, Bansal Institute of Engineering & Technology, Lucknow, India (e-mail: erramesh2007@gamil.com)

Dr. Abdullah, Assistant Professor, Department of Information Technology, Adigrat University (A Public University), Adigrat Tigray, Ethiopia-Africa.

Abhishek Yadav, Assistant Professor, Department of Computer Science & Engineering, Bansal Institute of Engineering & Technology, Lucknow, India.

For that reason, software design should be built in such a way so as to make them easily modifiable, maintainable, and if at all possible stable. One cause for this poor management is the lack of established measures for software maintainability [30, 9].With all cries and dissatisfaction, discipline is improving and maturing day by day. As an outcome, there is an imperative demand to put into Practice software engineering concepts, strategy, practices to avoid deviation, and to improve the software development process in order to deliver good quality maintainable software in time and within account. In conclusion, a lesser amount of consideration has been rewarded to the area of software maintainability. The design size and functionality of computer systems have full-grown for the duration of the past two decades in a very remarkable manner. Analyzability is a most important factor of system maintainability which increases the performance of maintenance process. Good analyzability makes the system more maintainable. Software maintainability can be improved by controlling object oriented characteristics such as coupling, cohesion, inheritance and polymorphism [22, 24, 26, 27].More complex systems always decrease the maintainability of the object-oriented software.

II. SOFTWARE MAINTAINABILITY

The key word of “software maintainability” first appeared in the categorization of maintenance. It is also planned as the first key attribute of good designed software by Sommerville [45] in the starting to his book. Maintainability is an essential and precious quality characteristic of software. Software maintainability always supports the maintenance process and assists the creation of superior quality software; an accurate measure of software quality totally depends on maintainability measurement. A lack of maintainability constantly contributes to a higher maintenance charge and effort [7, 8]. The aspiration of increasing the maintainability of object oriented design is not just to detect defects but more significantly, to detect defects as soon as they are introduced [9, 11].

III. REVIEW OF CLOSELY RELATED WORK

Broad range of maintainability calculation models have been planned in the literature within last two decades. A number of maintainability models/methodologies were proposed to facilitate the designers in measuring the

maintainability of object oriented software so as to produce superior and improved software systems. Opening from 1970s to 2020 a variety of maintainability Estimation models or techniques was developed. It is adopted in Jim McCall and Boehm quality model, which build the basis of ISO 9126 software quality model. Muthanna et al. (2000) developed a maintainability assessment model by the use of polynomial linear regressions. But this model could be helpful only for procedural software and not suitable for object oriented software. Study highlights software maintenance is a time consuming and costly phase of a software development life cycle. This paper examines the use of software design metrics to evaluate the maintainability of software systems. Study done by Di Lucca et.al (2004) providing web application centered maintainability model accurate to web applications only. Authors stated the increasing distribution of web based services in many and diverse business domains have triggered the need for new web applications. The urgent market demand enforces very short time for the development of new web applications and recurrent modifications for existing ones. In this paper Lucca et.al introduce a first idea for a web application based maintainability model the proposed model considers those peculiarities that makes a web application special from a traditional software system and a group of metrics allowing a valuation of the maintainability is acknowledged. Results from a few initial case studies to confirm the usefulness of the proposed model are presented in the paper. Study done by Hayes Zaho (2005) suggested a maintainability estimation model that considered software modules as easy to maintain and not easy to maintain. Such categorization can assist to recognize the modules, which are not easy to maintain Van Koten (2006) presents a Bayesian network maintainability prediction model for object oriented software design. The model is developed with the help of object oriented metric data presented in Li and Henry's datasets; which were composed from two dissimilar object oriented software systems. Prediction correctness of the model is calculated and compared with existing models. The outcomes mention that the Bayesian network model calculates maintainability not correctly than the regression centered models for one system. Study done by Zhou Leung MARS (2007) make use of a novel exploratory modeling method, Multiple Adaptive Regression Splines (MARS) to construct software maintainability assessment model using the data collected from two different object oriented software systems. Still the MARS model is not validated and not clears about the cause effect relation between object oriented metric and maintainability. Work done by MO. Elish & KO Elish (2009) recommended TreeNet model for maintainability calculation can be concern of as a series expansion alike to the appropriate functional relationship. TreeNet model uses two famous object oriented software datasets published by Li and Henry: UIMS and QUES datasets. The proposed model results designate that competitive prediction accuracy has been achieved when applying the TreeNet model. Study shows future work would be conducting additional studies with other datasets to extra support the findings of this paper, and to understand the full potential and probable limitation of TreeNet Study done by C Jin & JA Liu (2010) offerings the applications of support vector machine and unconfirmed learning in object oriented software maintainability estimation through metrics. In this study, the software maintainability predictor is performed at the source code level of development life cycle. The proposed dependent variable was software maintenance effort. Similarly the independent variables were five object oriented metrics determined clustering method. The results showed that the mean absolute relative error was 0.218 of the predictor. Subsequently, we found that support vector machine and clustering technique were supportive in emerging software maintainability predictor. Novel predictor can be used in the related software developed in the same background. Work done by Gautama Kang (2011) emphasized dimension of the software maintainability close the beginning in the software development life cycle, mainly at the design period is very significant, and it support designers to integrate required improvement and corrections at design phase for improving software maintainability of the delivered software. Earlier MEMOOD model was developed which estimates the maintainability of the software system on the basis of object oriented metrics of software system. This work has suggested a multivariate linear model Compound "MEMOOD", which assessments the maintainability of class diagrams of software systems. Subsequently study make a comparison of MEMOOD model and Compound MEMOOD model through regression analysis and it is found that Compound MEMOOD Model gives better results with the given dataset. Moreover, no quantitative comparisons have been presented in this study. Study done by Alisara Hincheeranan et.al (2012) evaluated maintainability seeing maintainability and extensibility as two sub factors of maintainability. He stated measuring maintainability of software system at the design stage may facilitate a software designer must improves the maintainability of software before deliver to a customer. In this paper author developed the Maintainability Estimation Tool (MET) for a maintainability estimation of software system. This tool assist a software designer for improves the maintainability of class diagram in design phase and facilitate reduces the growing high cost of software maintenance phase. Moreover, no quantitative validation has been presented in this study. Al Dallal, J. (2013) considers classes of three open source software systems. For every class, study accounts for two real maintainability indicators; (1) the number of revised lines of code (2) the number of revisions in which the class was concerned. Through 19 internal quality estimations, novelists empirically discover the effect of size, cohesion and coupling on class level maintainability. Acquired outcomes display that classes with enhanced qualities (greater cohesion values and lesser coupling and size values) have continuously improved maintainability (i.e. are more possible to be effortlessly modified) than those of inferior qualities. The proposed prediction models can help software designers to find classes with low maintainability. In the work done by R. & Chug A. (2014) offered a novel metric suite to overwhelmed the shortages and redefine the relationship amongst design metrics through software

maintainability in data intensive applications. The proposed metric suite is estimated, analyzed using five proprietary software systems. The outcomes display that the suggested metric suite is very supportive for maintainability calculation of software systems in common and for data intensive software systems in specific. The proposed metric suite may be considerably useful to the developers in studying the maintainability of intensive software systems their research paper assembled a methodical analysis of studies on software maintainability amongst the years 1991 to 2015[31]. The authors organized and scrutinized the effort on maintainability by tangents of design metrics, tools, algorithms, data sources and so on. They concise that design metrics was still the greatest preferred choice to capture the features of any given software before installing it additional in prediction model for formative the software before deploying them. Work done by Rajendra et. al. (2015) evaluated and authenticated the model for software maintainability based on quality factors flexibility and extendibility [39]. The outcomes they arrived stood important but by other factors newer models for maintainability with better-quality outcomes could be proposed. Study done by Ruchika Malhotra et.al. (2016), in corresponding software maintainability. Celia Chen et al. (2017) in their work stressed the vast level of cost saving in software by understanding the significance of maintainability, and recommended replies to queries of decision concerning what portions of software to be reused, what portions to be redeveloped, the theoretical valuation of effort requisite to do so and thus giving pointers as how to decrease overall budgets [32].

IV. COMPARATIVE STUDY

A Critical Look of Maintainability Models Consider by Various Expert has been prepared in Table 1.

Table 1: A Critical Look of Maintainability Models Consider by Various Expert.

Researcher	Year	Maintainability Measurement Approach	SDLC Stage	Validation
Geoferry and kemere	1991	Cyclomatic Complexity Density	Code Level	Yes
Oman Hagemeister	1992	Halstead's Effort (aveE), McCabe' Cyclomatic Complexity (G), LOC (Lines of Code)	Code Level	No Validation
Li -Henry	1993	Henry model based on coupling between classes	Code Level	Yes
Coleman Oman	1994	Oman model	Code Level	Yes
Welker Oman	1995	(Improved Oman Model) CyclomaticComplexity V(g'),LOC (Lines of Code)	Code Level	Not Validated
Dromey's Quality Model	1995	Quality Model	Code Level	Theoretical justification
Muthanna et al.	2000	Model based on Polynomial Linear Regression	Design Phase	No Validation
Huffman Hayes et al.	2003	Observe Mine Adopt (OMA) Based on Maintainability product	Code Level	Yes
Lucca- Fasolino WAMM	2004	Web Application Maintainability Model	Web based Approach	Web based Approach
Hayes Zaho	2005	(Main Pred Model) LOC (Lines of Code), TCR (True Comment Ratio)	Code level	No Validation
Koten-Gray	2006	Bayesian Network Maintainability Prediction Model	Code level	Yes
Zhou -Leung MARS	2007	Multiple Adaptive Regression Splines	Design Phase	No Implementation.
Prasanth Ganesh & Dalton	2008	With the help of FRT(Fuzzy Repertory Table)	Design Phase	Not Validated

MO. Elish & KO Elish	2009	Produced Treenet model using stochastic gradient boosting	Code level	Not Validated
C Jin & JA Liu	2010	Based on Support vector machine	Code level	Based on vector machine
S. Rizvi et al.	2010	MEMOOD Model	Design Phase	No Validation
Gautama Kang	2011	Compound Memood Model	Design Phase	Not Validated
Alisara et al.	2012	Maintainability Estimation Tool (MET)	Code level	No Validation
Al Dallal, J.	2013	Object oriented class maintainability calculation via internal quality attribute.	Design and code level	Not Validated
R. & Chug A.	2014	A Metric Suite for Predicting Software Maintainability in Data Intensive Applications.	Design Phase	Based on Metrics
Rajendra et. al.	2015	Maintainability based on quality sub factors	Design Phase	Based on regression line
Ruchika Malhotra et.al.	2016	Maintainability by tangents of design metrics	Not clear	Not Validated
Celia Chen et al.	2017	Importance of software maintainability	SDLC	Theoretical estimation
Hadeel Alsolai et al.	2019	Maintainability in Object Oriented Systems Using Ensemble Techniques	SDLC	Validated

After an in systematic review of related work, it seems that maintainability measurement should be done at design stage of software development life cycle. To evaluate maintainability at design phase it is important to discover maintainability factors that have direct impact on maintainability evaluation. It is obvious from comprehensive literature review that Changeability and Stability is a most important factor for object oriented software maintainability which increases the performance of maintenance process.

V. MAINTAINABILITY FACTORS

It is evident from systematic literature survey that there is an opposition among researchers and practitioners in taking into consideration the maintainability factors for evaluating maintainability of object oriented software in general and at design phase. A consolidated table for the maintainability factors recognized by area experts is concluded in Table 1. It is noticeably highlighted from the table that Changeability and Stability are the key maintainability factors.

Table 2: Maintainability Factors Consider by Various Experts

Maintainability Factors ➔ / **Study/Source ⬇**	**Modifiability**	**Readability**	**Reusability**	**Simplicity**	**Maintainability**	**Analyzability**	**Conciseness**	**Modularity**	**Testability**
McCall [40]	✓		✓	✓			✓		
Boehm [41]	✓								✓
ISO 9126 [42]	✓		✓		✓	✓	✓		
ISO/IEC25010: 2011[47]	✓		✓			✓		✓	✓
FURPS [43]	✓		✓		✓				✓
Fuzzy Model [44]		✓						✓	
Rizvi et al. [45]	✓		✓			✓			
Wiebe et al. [46]	✓			✓					

International Journal of Innovative Research in Computer Science & Technology (IJIRCST)
ISSN: 2347-5552, Volume- 8, Issue- 5, September- 2020
https://doi.org/10.21276/ijircst.2020.8.5.4
www.ijircst.org

VI. DESIGN PROPERTIES THAT INFLUENCES MAINTAINABILITY

Object oriented design properties overcome the negative aspect of procedure oriented design. In order to design the Software through an object oriented approach, the three essential properties are considerably being used i.e. encapsulation, inheritance and coupling. Object oriented design properties that have positive impact on maintainability evaluation has been identified and consolidated chart for the same is given in Table 3.

Table 3: Object oriented design properties contributing in maintainability measurement: a critical look

Design Properties➔ Source/Study⬇	**Cohesion**	**Coupling**	**Encapsulation**	**Inheritance**	**Polymorphism**
Gregor et al. (1996)		✓	✓	✓	
Bruce & Shi (1998)	✓	✓		✓	✓
B. Pettichord (2002)	✓	✓		✓	
Baudry et al. (2002)		✓	✓		✓
M Bruntik (2004)	✓			✓	
S .Mouchawrab (2005)	✓	✓		✓	
E Mulo(2007)		✓	✓	✓	✓
Sujata et al. (2011)	✓			✓	
P. Malla et al. (2012)	✓	✓	✓	✓	
Nikfard et al. (2013)		✓	✓	✓	✓

VII. CRITICAL OBSERVATIONS

After successful completion of the literature review a number of important explanations can be enumerated as follows.

- If we measure the software maintainability at an early stage that is design phase in the software development process may significantly improve the software quality and as well as client happiness, and decrease overall cost, time and effort of rework.
- In order to reducing effort in measuring maintainability of object oriented design we require to recognize a minimal set of maintainability factors for object oriented development procedure, which have optimistic impact on maintainability estimation.
- Object oriented software characteristics are required to be recognized and after that the set of maintainability factors appropriate at the design phase should be finalized.
- Further, maintainability metrics have to be chosen at the design phase for the reason that metric selection is an important step in maintainability measurement of objects oriented design.

VIII. CONTRIBUTION

The most important contribution of this review paper is in the field of maintainability measurement. We have conducted a systematic review in this paper. The dissimilar factors maintainability and measure for these factors are identified. Overall contribution is listed as follows:

- Systematic literature review of closely related work
- A complete step by step improvement of the systematic review procedure is described. It will help to further study as a reference for undertaking SLR.
- Recognition of key research papers/chapters related to the maintainability study in software engineering domain
- Discovery of maintainability factors and measurement in the recent domain of OOD
- Identification and arrangement of different concepts about the software maintainability in the present software engineering domain.
- A proposed design property that influences maintainability to assist the self-assessment for designers to identify software maintainability factors.
- Structure and well defined assessment process for finding factors from high level to lower measurable level.

IX. CONCLUSION

A lot of maintainability approaches have been proposed in the existing literature for evaluating software maintainability. A review of the related literature shows that most efforts have been put at the later phase of software development life cycle especially at design phase. On the other hand, the lack of maintainability at early stage may not be compensated during subsequent development life cycle. In order to obtain consistent and correct measures of maintainability, it is advisable to recognize the factors that affecting maintainability directly. Though, getting a universally accepted set of maintainability factor is impossible, effort have been made to identify the key factors of maintainability for the same.

REFERENCES

[1] K.K. Aggarwal, Yogesh Singh. New Age International, Jan 1, 2005 - Software engineering.

[2] Singh, Hardeep, and Aseem Kumar. "A Novel Approach to Enhance the Maintainability of Object Oriented Software Engineering During Component Based Software Engineering." International Journal of Computer Sci. and Mobile Computing 3.3 (2014): 778-786.

[3] Al Dallal, Jehad. "Object-oriented class maintainability prediction using internal quality attributes." Information and Software Technology 55.11 (2013): 2028-2048.

[4] Singh, Pradeep Kumar, and Om Prakash Sangwan. "Aspect Oriented Software Metrics Based Maintainability Assessment: Framework and Model." (2013): 1-07.

[5] McCall, J.A., Richards, P.K., and Walters, G.F., (1977) "Factors in Software Quality", RADC TR-77-369, Vols I, II, III, US Rome Air Development Center Reports

[6] G. M. Berns. Assessing software maintainability.ACM Communications, 27(1), 1984.

[7] Bowen, T. P., Wigle, G. B., Tsai, J. T. 1985. Specification of software quality attributes. Tech. Rep. RADC-TR- 85-37, Rome Air Development Center.

[8] Sneed, H., Mercy, A. (1985), Automated Software Quality Assurance. IEEE Trans. Software Eng., 11Bi, 9: 909-916.

[9] Grady, Robert, Caswell, Deborah (1987), Software Metrics: Establishing a Company-wide Program. Prentic Hall. pp. p. 159.ISBN 0138218447.

[10] Gill Geoffrey K. and Chris F. Kemerer. (1991). "Cyclomatic Complexity Density and Software Maintenance Productivity, "IEEE Transactions on Software Engineering, Dec, pp.1284-1288.

[11] P. Oman and J. Hagemeister, "Metrics for assessing a software system's maintainability," Software Maintenance, 1992, pp. 337 - 344.

[12] W. Li and S. Henry, "Object-Oriented Metrics that Predict Maintainability", Journal of Systems and Software, vol 23, no.2, 1993, pp.111-122.

[13] D. Coleman, D. Ash, B. Lowther and P. Oman, "Using Metrics to Evaluate Software System Maintainability", IEEE Computer; 27(8), pages 44–49, 1994.

[14] Welker, K. and Oman, P.W., Software Maintainability Metrics Models in Practice, CrossTalk, Nov./Dec.1995, pp. 19-23 and 32

[15] Geoff R. Dromey's Model, (Feb 1995) (vol. 21 no. 2), IEEE Transaction on Software Engineering, A Model for Software Product Quality.

[16] Dromey, R.G.: Concerning the Chimera. IEEE Software 13 (1), pp. 33--43 (1996).

[17] S. Muthanna, K. Kontogiannis, K. Ponnambalaml and B. Stacey, "A Maintainability Model for Industrial Software Systems Using Design Level Metrics", In Working Conference on Reverse Engineering (WCRE'00), 2000

[18] M. Genero, M. Piattini, E. Manso, G. Cantone, "Building UML class diagram maintainability prediction models based on early metrics", Proceedings 5th International Workshop on Enterprise Networking and Computing in Healthcare Industry, , IEEE, 2003, pp. 263-275.

[19] Hayes, J. Huffman, Mohamed, N., Gao, T. The Observe-Mine-Adopt Model: An agile way to enhance software maintainability. Journal of Software Maintenance and Evolution: Research and Practice, Volume 15, Issue 5, Pages 297 – 323, October 2003.

[20] G. DiLucca, A. Fasolino, P. Tramontana, and C. Visaggio. Towards the definition of a maintainability model for web applications. In Proceeding of the 8th European Conference on Software Maintenance and Reengineering, pages 279–287. IEEE Computer Society Press, 2004.

[21] Kiewkanya, M., Jindasawat, N., Muenchaisri, P., (2004) "A Methodology for Constructing Maintainability Model of Object-Oriented Design," Proc. 4th International Conference on Quality Software, 8 - 9 Sept., 2004, pp. 206 - 213. IEEE Computer Society.

[22] Hayes J.H. and Zaho L (2005), "Maintainability Prediction a Regression Analysis of Measures of Evolving Systems", Proc.21st IEEE International Conference on Software Maintenance, 26-29 Sept.2005, pp.601-604.

[23] C.V. Koten, A.R. Gray, "An application of Bayesian network for predicting object-oriented software maintainability", Information and Software Technology Journal, vol: 48, no: 1, pp 59- 67, Jan2006.

[24] K.K. Aggarwal, Y. Singh, P. Chandra and M. Puri, " Measurement of Software Maintainability Using a Fuzzy Model", Journal of Computer Sciences, vol. 1, no.4, pp. 538-542, 2005 ISSN 1549-3636 © 2005 Science Publications.

[25] K. K. Aggarwal, Y. Singh, A. Kaur and R. Malhotra, "Application of Artificial Neural Network for Predicting Maintainability using Object-Oriented Metrics, World Academy of Science, pp. 140-144, 2006.

[26] Sub has Chandra Misra, "Modeling Design/Coding Factors That Drive Maintainability of Software

Systems", Software Quality Journal, 13, pages 297-320, 2005.

[27] Y. Zhou and H. Leung, "Predicting object-oriented software maintainability using multivariate adaptive regression splines", Journal of Systems and Software, vol. 80, no. 8, pp. 1349- 1361, 2007

[28] Wang Li-Jin Hu Xin-Xin Ning Zheng-Yuan Ke Wen-Hua ,"Predicting Object-Oriented Software Maintainability Using Projection Pursuit Regression.", Proceedings of the 2005 International Conference on Software Engineering Research and Practice, SERP ,vol.2,pp.942-946.

[29] MO. Elish and KO. Elish, "Application of TreeNet in Predicting Object-Oriented Software Maintainability: A Comparative Study", European Conference on Software Maintenance and Reengineering, pp 1534-5351, March 2009, DOI 10.1109/CSMR.2009.57.

[30] Rizvi S.W.A. and Khan R.A. (2010) "Maintainability Estimation Model for Object-Oriented Software in Design Phase (MEMOOD)", Journal of Computing, Volume 2, Issue 4, April 2010,

[31] Malhotra et.al, Software Maintainability Prediction using Machine Learning Algorithms." Software Engineering: An International Journal (SEIJ), Vol. 2, No. 2, SEPTEMBER 2012

[32] Celia Chen , Alfayez R ,Kamonphop Srisopha and Lin Shi, Why Is It Important to Measure Maintainability and What Are the Best Ways to Do It?, IEEE/ACM 39th

[33] International Conference on Software Engineering Companion (ICSE-C), July 2017.

[34] C Jin , A. L. Jin , "Applications of Support Vector Machine and Unsupervised Learning for Predicting Maintainability using Object- Oriented Metrics", Second International Conference on Multi Media and Information Technology , vol 1 ,no : 1, pp 24-27, April 2010.

[35] Gautam C, kang S.S (2011), "Comparison and Implementation of Compound MEMOOD MODEL and MEMOOD MODEL", International journal of computer science and information technologies, pp 2394-2398.

[36] Malhotra et al. "Software Maintainability Prediction using Machine Learning Algorithms." Software Engineering: An International Journal (SEIJ), Vol. 2, No. 2, SEPTEMBER 2012

[37] Alisara Hincheeranan and Wanchai Rivepiboon," A Maintainability Estimation Model and Tool." International Journal of Computer and Communication Engineering, Vol. 1, No. 2, July 2012.

[38] Dubey et.al."Maintainability Prediction of Object Oriented Software System by Using Artificial Neural Network Approach." International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-2, Issue-2, May 2012.

[39] Laxmi Shanker Maurya et.al," Maintainability assessment of web based application.'', Journal of Global Research in Computer Science, Vol 3, No. 7, July 2012.

[40] Rajendra Kumar and Dhanda N, Maintainability Measurement Model for Object-Oriented Design, International Journal of Advanced Research in Computer and Communication Engineering, Vol. 4, Issue 5, May 2015.

[41] McCall, J.A., Richards, P.K., and Walters, G.F., (1977) "Factors in Software Quality", RADC TR-77-369, Vols I, II, III, US Rome Air Development Center Reports.

[42] Boehm, B. W., Brown, J. R., Kaspar, H., Lipow, M., McLeod, G., and Merritt, M., (1978) Characteristics of Software Quality, North Holland.

[43] ISO 9126-1 Software Engineering - Product Quality - Part 1: Quality Model, 2001.

[44] Grady, Robert, Caswell, Deborah (1987), Software Metrics: Establishing a Company- wide Program. Prentice Hall. pp. p. 159. ISBN 0138218447.

[45] Sneed, H., Mercy, A. (1985), Automated Software Quality Assurance. IEEE Trans. Software Eng., 11Bi, 9: 909-916.

[46] Sommerville, I. (1992). Software Engineering. 4th ed. New York, Addison- Wesley.

[47] Hordijk, Wiebe, and Roel Wieringa. "Surveying the factors that influence maintainability: research design." ACM SIGSOFT Software Engineering Notes. Vol. 30. No. 5. ACM, 2005.

[48] Larrucea X., Santamaria I., O'Connor R., Messnarz R. (eds) Systems, Software and Services Process Improvement. EuroSPI 2018. Communications in Computer and Information Science, Vol 896, pp. 492-503. Springer, Cham.

