EMPIRICALLY VALIDATED SIMPLICITY EVALUATION MODEL FOR OBJECT ORIENTED SOFTWARE

Dr. Abdullah[1] and Dr. Mahfuzul Huda[2]

[1]Assistant Professor, Adigrat University, Adigrat- Tigray, Ethiopia-Africa.
[2]Assistant Professor, Saudi Electronic University, Riyadh -Saudi Arabia

ABSTRACT

Software program developers need to go from beginning to ending and understand source code of the program and other software attributes. The software complexities and length of the program exceedingly affects many design level quality attributes, specifically Simplicity, Testability and software Maintainability. Incomplete design of any software generally leads to misunderstanding and ambiguities and therefore to gives faulty design and development results. This is mainly seeming and appears owing to the absence of it's an appropriate observation, design and development control. However, high level design and program simplicity are very necessary and one of the vital attributes of the system development cycle.

This research paper highlights the impact and significance of design level software simplicity in common and as a one of the most useful key factor or index of software quality assurance and testing. In this research work principally there are three major efforts are made. As a first contribution, a valuable relationship between software design quality factor simplicity and related object oriented design properties, this has been set up. In the second contribution, using design level corresponding metrics a simplicity evaluation model for object oriented software is developed. Subsequently, the developed simplicity model has been rationally authenticated by means of experimental data try-out.

KEYWORDS

Simplicity, Reliability, Maintainability, Testability, Portability, Flexibility, Object Oriented Software.

1. INTRODUCTION

To timely fulfill the continuing changing demands and the requirements of users and the customer or may be some additional reasons, developed software must want to be proper renovated or altered on or after time to time. This procedure of maintenance or modification is generally worked out by current/existing programmers and developers, which might be not have built that previous application software [5]. They must require proper narrating and proper recognizing source codes and additional associated credentials [21]. Yet for the program creators of the object oriented design, next a break of not many years, it might not be always a simple or easy work for them to understand original source programs and other associated design documents of the program or the software [8, 22]. Incorrect interpretations and explanations can proceed to or direct to misconceptions and finally to not acceptable delivered results [15]. Rundown of an understanding and the capability to eloquent the procedures in use, it is not probable that it can be enhanced [16, 23].

So, the ability to be understood and the design simplicity of software have many direct influences on the criteria that indirectly or directly disturb quality of software. Complicated software design assuredly direct to bad maintainability and testability results, which in go products to un successful maintenance and testing that may outcome to bad significances [9, 24]. It is unspoken truth that defects of software design creation have a robust adverse effect on expected quality attributes. Organizing a good quality complex design endures to be an inefficiently well-defined procedure [10]. Hence, software implementation design and further development must be simple as well as minimal complex, this minimizes the required development efforts for the unexpected coming functional and operational extensions. Moreover, the design is alienated into functionally separated and less interdependent modules of reasonable program size [25]. Finally, we can achieve to build in this method so as to make them simply testable, maintainable, and possibly stable [11, 14].

2. SOFTWARE QUALITY

In this segment, it discusses the contents of the following quality model for software: McCall software quality model, Boehm software quality model, FURPS model, and finally Dromey model. The McCall's software quality method has focus on three (3) main value of software application: Product Transition perspective (that is adaptableness to new platform setting) product Revision perspective (The capability to go through proper changes) product Operations perspective (Its operational characteristics).

McCall's Quality Assurance Model contains 11 software quality factors and twenty-three quality criteria [3]. Where the quality criteria define disparate kinds of software properties and software quality criterions are valuable characteristics to single or supplementary of the software quality factors. Boehm's quality model works to qualitatively evaluation of the quality of software [1, 7]. The high level factors address three classifications; maintainability, portability and general utility into as utility. In the intermediate level criteria, Boehm quality model has seven quality factors like reliability, portability, Usability, efficiency, Human engineering, understandability, flexibility. Dromey's software quality model proposed a complete framework to evaluate the phases of requirement, design and implementation [12]. The high level design properties for the Dromey's implementation software quality model that include: internal, correctness, descriptive and contextual.

Furps quality model [13] formerly presented by quality expert Grady [9], then it is extended by IBM Software (Rational) into FURPS+. Where the '+' indicates such requirements as design constraints, requirements, implementation, physical requirements and interface requirements. There are mainly four characteristics in FURPS quality model. The quality factors and features of the software in the FURPS quality mode clearly indicated under ISO 9000 and stated as "they provide crystal clear implementation guidelines for product quality assurance [17]. ISO 9000 is a process or procedure oriented approach for software quality management [19]. It processes designing, implementing, documenting, monitoring, supporting, improving and controlling. Recently, the ISO/IEC 9126-1: 2001 product quality model of software, which defined mainly six quality factors, has replaced by software product quality model and ISO/IEC 205010:2011 system [20]. The ISO 25010 is the widely used quality standard model now a day. ISO 25010 uses ten main quality factors: operability, Functional, suitability, reliability, security, efficiency, performance, compatibility, maintainability, and portability. The given 28 software quality features are arranged and given under the basis of major six quality criteria [20].

3. SOFTWARE SIMPLICITY

When you genuinely discuss about software design simplicity, means you want to approve and validate of it because it reduced complicated details [18]. Simplicity tends to promote ease of testing and maintenance. Software simplicity is one of the most noteworthy quality factor for any type of system design and development. Designing a simple system is time consuming. It will have significant impact with the architecture level. It is crucial and challenging task that a product design shows reduced complexity and remains as simple design as possible while developer still being able to meet the needs of the requirements and services. Product's systems designs grow and it becomes more challenging and giving complex looks over the time. Eventually starting with a software system's design that is already getting complex means starting with at disadvantages. The more complex the system leads to the more difficult it is to have a comprehensive and an accurate model. An excessively complex design of a system results in a condition where no single user can understand it all at any one time.

The software or program quality factor describes dissimilar kinds of crucial system properties, whether software quality criteria are mainly attributes to single or supplementary of the quality factor. The quality model presented by Boehm try to attempts qualitatively assessment of the excellence of software. The high level software characteristics explain three important classifications; maintainability, portability and general explanation into as utility. Where at the intermediate level properties, the Boehm's design product quality model described seven important features of the quality factors like as efficiency portability, Usability, flexibility, Human engineering, reliability, understandability, [2]. Table 1 is denoted as the software quality factors and quality-criteria of Boehm quality-mode.

4. Simplicity Quality Criteria And Related Factors

Quality criteria are the characteristics which properly define the quality factor. The software quality criteria for the related factors are the characteristics of the software creation process or software product by which the factor can be defined. The relationships amid the simplicity quality factors between the related crucial criteria can be seen in Table 1.

Table1. Criteria of Simplicity Quality-Factors [1, 27, 28, 29, 30, 31]

Criterion	Definition	Related Factors
Simplicity	Those attributes or the factors of the program or the software that gives implementation of functions in the most understandable manner.	• Reliability • Maintainability • Testability • Portability • Flexibility

5. Establishing Relation between Design Properties and Simplicity Factors

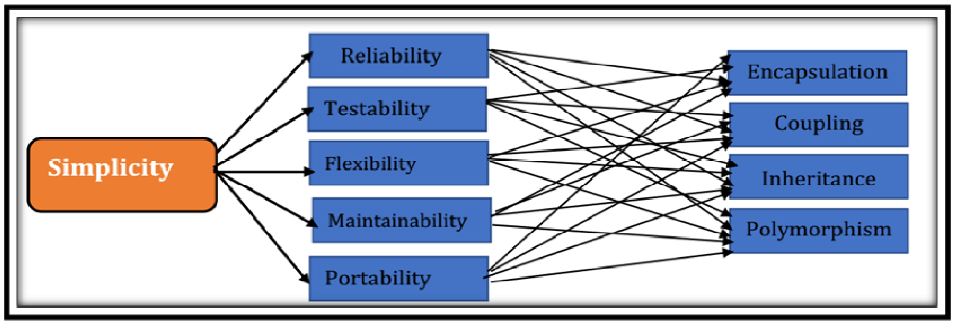

Figure1: Mapping between Design Properties and Simplicity Factors

6. MODEL DEVELOPMENT FOR SIMPLICITY EVALUATION OF OBJECT ORIENTED SOFTWARE

In direction to create Simplicity evaluation model for object oriented software, “Multivariate Linear Regression System” has been applied, that is specified as given below.

Z=b0+ b1 x1+ b2 x2+ b3 x3……bn xn **Eq. (1)**

Simplicity= b0+ b1 x Encapsulation+ b2 x Coupling+ b3 x Inheritance + b4 x Polymorphism **Eq. (2)**

The data set accustomed for creating simplicity evaluation model for object oriented software is occupied from [26] that have been collected through the measured experimentation. It includes a group of 20 object oriented design class diagrams (symbolized as P1 to P20) and the associated metrics value of respective class diagram. Moreover, the mean value of the specialist’s score of simplicity of these class diagrams is also given and labeled as ‘Known Values’ in this research work. The association amid simplicity factor and program design properties has been recognized as represented in figure one. Using SPSS, respective value of each coefficients is decided and simplicity evaluation model is articulated as given below.

Simplicity= 4.741 -.686 x Encapsulation -1.843 x Coupling + .334 x Inheritance + 2.841 x Polymorphism Eq. (3)

Table 2 shows the coefficients for Simplicity evaluation model. The Unstandardized coefficients component of the table 2 gives us the values that we require in order to develop the regression equation (2). The experimental assessment of Simplicity is very hopeful to get simplicity index of object oriented design for low cost software maintenance.

Table 2: Coefficients for Simplicity Evaluation Model

Model		Unstandardized		Standardized	t	Sig.
		B	Std.	Beta		
1	(Constant)	4.741	5.315		.892	.413
	Encapsulation	-.686	.601	-.319	-1.142	.305
	Coupling	-1.843	.612	-.970	-3.009	.030
	Inheritance	.334	1.436	.075	.233	.825
	Polymorphism	2.841	1.350	.819	2.105	.089
a. Dependent Variable: Simplicity						

6.1. SIMPLICITY MODEL SUMMARY

The model summary table 3 results are very helpful when writing multiple regressions. Capital (R), is the coefficient determinant that tells us how powerfully the all independent-variables are associated to the respective dependent- variable. The value of “R Square” is also very encouraging as it gives us the coefficient determination.

Table 3: Simplicity Evaluation Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	.824[a]	.679	.422	1.13407
a. Predictors: (Constant), Polymorphism, Encapsulation, Inheritance, Coupling				

7. INVESTIGATIONAL TRYOUT

No problem, in what way powerful a hypothetical conclusion may be, it has to be experimentally authenticated if it is functioning to be of any real-world use [6]. This is real in all engineering systems, counting software engineering. Hence, in adding to the hypothetical or theoretical authentication, an investigational and experimental test is similarly significant in directive to make the claim additional adaptable. In assessment of this truth, an investigational authentication of the developed simplicity evaluation model (equation 2) has been performed by the assistance of design level metrics specified in the data set [26]. Summary of the values obtained by the developed model against the 'Known Values' of simplicity are given in Table 4.

Spearman's rank correlation coefficient (rs) used to get the association among values of simplicity nature of a design using developed model and it's given values. The 'rs' was calculated by the method specified as under:
Rank Relation (rs) –

$$r_s = 1 - \frac{6\sum D^2}{n\left(n^2 - 1\right)}$$

'd' = Variance amongst, Computed Rank and Known Rank of Simplicity
'n' = Quantity of all projects used in the experimentation.

Table 4: Simplicity Index Values

Projects Detail	Projects Known Values	Obtained Values Using Developed Simplicity Model
P-11	7.0	5.54
P-12	8.3	3.92
P-13	7.9	4.56
P-14	8.6	7.75
P-15	9.6	7.90
P-16	7.4	4.31
P-17	8.5	6.91
P-18	6.9	2.40
P-19	9.3	6.24
P-20	6.8	2.69

Spearman's Rank Correlation Coefficient (rs = 0.84242) computed for the developed model is more than the threshold value for n=10. This displays that the values of simplicity model calculated using proposed model are extremely associated with the 'Given Values'. Consequently, the association is satisfactory with the high level of confidence, i.e. at the 0.05. The correlation is up to standard with high level of justification i.e. at the 95%. Hence, short of any damage of oversimplification our study can accomplish that simplicity evaluation model (in eq. 2) calculations are consistent and effective in the viewpoint.

8. Conclusion and Future Research Direction

This research work shows the importance of simplicity and its correlation with object oriented design properties viz. Encapsulation, Coupling, Inheritance and Polymorphism. Further, research study developed a simplicity evaluation model SEMOOD with the assistance of multiple linear regression method on object oriented design properties. Statistical result confirms that simplicity evaluation model is extremely significant and up to standard. The perfect validation on the simplicity model it is to be completed in future on live industrial projects for improved suitability and usefulness. Software simplicity is dynamic and one of the greatest noteworthy parts of the software development life cycle nowadays. Above described five quality attributes, have numerous highlighted properties in shared, in counting low level coupling, modularity and high level cohesion. Design level Simplicity is directly boosted software Reliability, Flexibility, Testability, Maintainability and Portability. Simplicity of the software when combined with design quality attributes and criteria supports to develop project with less development and further maintenance cost in the minimum time and framework, as well as improve acceptable consistency and better reliability of the final delivered software. Still, the proposed research work wants to be more investigational tryout with a bigger set of data for better level of acceptability and utility.

References

[1] Boehm BW, Brow JR, Lipow M, McLeod G, Merritt M. Characteristics of software quality. North Holland Publishing. Amsterdam, the Netherlands; 1978.

[2] Abdullah, Dr, M. H. Khan, and Reena Srivastava. “Flexibility: A Key Factor To Testability”, International Journal of Software Engineering & Applications (IJSEA), Vol.6, No.1, January 2015. DOI: 10.5121/ijsea.2015.6108.

[3] McCall JA, Richards PK, Walters GF. Factors in software quality, RADC TR-77-369: 1977. (Rome: Rome Air Development Center)

[4] Huda, M., Arya, Y.D.S. and Khan, M.H. (2015) Evaluating Effectiveness Factor of Object Oriented Design: A Testability Perspective. International Journal of Software Engineering & Applications (IJSEA), 6, 41-49. http://dx.doi.org/10.5121/ijsea.2015.6104

[5] Huda, M., Arya, Y.D.S. and Khan, M.H. (2015) Metric Based Testability Estimation Model for Object Oriented Design: Quality Perspective. Journal of Software Engineering and Applications, 8, 234-243. http://dx.doi.org/10.4236/jsea.2015.84024

[6] Abdullah, Dr, Reena Srivastava, and M. H. Khan. "Testability Measurement Framework: Design Phase Perspective”. International Journal of Advanced Research in Computer and Communication Engineering, Vol. 3, Issue 11, Pages 8573- 8576 November 2014.

[7] Boehm BW, Brown JR, Lipow M. Quantitative evaluation of software quality, In Proceeding of the 2nd International Conference on Software engineering. 1976;592605.

[8] Abdullah, Dr, M. H. Khan, and Reena Srivastava. “Testability Measurement Model for Object Oriented Design (TMMOOD)”. International Journal of Computer Science & Information Technology (IJCSIT), Vol. 7, No 1, February 2015, DOI: 10.5121/ijcsit.2015.7115.

[9] Grady, RB. Practical software metrics for project management and process improvement, Prentice Hall; 1992.

[10] Huda, M., Arya, Y.D.S. and Khan, M.H. (2015) Quantifying Reusability of Object Oriented Design: A Testability Perspective. Journal of Software Engineering and Applications, 8, 175-183. http://dx.doi.org/10.4236/jsea.2015.84018

[11] Abdullah, Dr, Reena Srivastava, and M. H. Khan. “Modifiability: A Key Factor To Testability”, International Journal of Advanced Information Science and Technology, Vol. 26, No.26, Pages 62-71 June 2014.

[12] Dromey RG. Concerning the Chimera (software quality). IEEE Software. 1996;1:3343.

[13] Jacobson I, Booch G, Rumbaugh J. The unified software development process, Addison Wesley; 1999.

[14] Drown DJ, Khoshgoftaar TM, Seiya N. Evaluation any sampling and software quality model of high assurance systems, IEEE Transaction on systems, Mean and Cybernetics, Part A: Systems and Human. 2009;39(5):1097-1107.

[15] Abdullah, Dr, Reena Srivastava, and M. H. Khan. "Testability Estimation of Object Oriented Design: A Revisit". International Journal of Advanced Research in Computer and Communication Engineering, Vol. 2, Issue 8, pages 3086-3090, August 2013.

[16] Tomar AB, Thakare VM. A systematic study of software quality models, International Journal of software engineering & application. 2011;12(4):61-70.

[17] ISO 9001:2005, Quality management system Fundamentals and vocabulary; 2005.

[18] Huda, M., Arya, Y.D.S. and Khan, M.H. (2014) Measuring Testability of Object Oriented Design: A Systematic Review. International Journal of Scientific Engineering and Technology (IJSET), 3, 1313-1319.

[19] ISO 9001:2001, Quality management system Requirements; 2001.

[20] ISO /IEC25010: Software engineering– system and software quality requirement and evaluation (SQuaRE)- system and software quality model; 2011.

[21] Huda, M., Arya, Y.D.S. and Khan, M.H. (2015) Testability Quantification Framework of Object Oriented Software: A New Perspective. International Journal of Advanced Research in Computer and Communication Engineering, 4, 298- 302. http://dx.doi.org/10.17148/IJARCCE.2015.4168

[22] McCable TJ. A complexity measure, IEEE Transaction on Software Engineering, 1976; SE-2(4):308-320.

[23] Kemerer CF. An empirical validation of software code estimation models, Communications of the ACM. 2008;30(5):416-429.

[22] Basili VR, Weiss DM. A methodology for collecting valid software engineering data, IEEE Transactions on Software Engineering. 1984;SE-10:728-738.

[25] Pandian CR. Software metrics – A guide to planning, Analysis, and Application, CRC press Company; 2004.

[26] MoboDexter Software India Pvt. Ltd., Novel Tech Park, Third Floor, #43/4, GB playa, Hosur Road Bangalore

[27] Tomar AB, Thakare VM. A systematic study of software quality models, International Journal of software engineering & application. 2011;12(4):61-70.

[28] Boehm BW, Brow JR, Lipow M, McLeod G, Merritt M. Characteristics of software quality. North Holland Publishing. Amsterdam, the Netherlands; 1978.

[29] Dromey RG. A model for software product quality. IEEE Transaction on Software Engineering. 1995;21:146-162.

[30] ISO /IEC25010: Software engineering– system and software quality requirement and evaluation (SQuaRE)- system and software quality model; 2011.

[31] Lee, Ming-Chang. "Software Quality Factors and Software Quality Metrics to Enhance Software Quality Assurance." British Journal of Applied Science & Technology 4.21 (2014).

AUTHORS

Dr. Abdullah, Assistant Professor, Department of Information Technology, Adigrat University, Ethiopia (Africa). Dr. Abdullah did MCA from UP Technical University & Ph.D. (Computer Applications) from BBD University, Lucknow India. He has more than Twelve Years of Teaching and Research experience as an Associate Professor / Assistant Professor/ Lecturer in reputed Universities of India and Foreign/Abroad.Dr. Abdullah has written Three Books and sixteen study materials for different Indian Government and Private Universities published numerous articles, several research papers in the National and International refereed Journals and conference proceedings. His published research work cited and accepted by lot of academicians and researchers across the world and published research papers got 76 Google Scholar citations. His areas of expertise are Software Quality Assurance, Software Testing, Object Oriented Design and Development. He has around Twelve Years of Teaching and Research experience in India and Abroad. He has contributed a lot in academics & research in the form of delivering lectures, research supervision and curriculum developments. He has also organized Seminars, National & International Conferences. He is a Reviewer and Editorial Board member of various national and International research journals as well as professional bodies.

Dr. Mahfuzul Huda currently working as Assistant Professor in the department of Computer Science & Engineering, Saudi Electronic University, Kingdom of Saudi Arabia. He has more than ten years of core teaching experience. Dr. Huda has credible contribution in research work. His research papers accepted and appreciated by various researchers across the world and his published research papers got 66 Google Scholar citations with h-index: 5 & i10-index: 3, in a very short span of time. Subsequently, he has guided more than 50 students for academic projects and seminars and attended more than 30 seminars and workshops related to academic work.