IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology
Vol. 2, Issue 12, December 2015

Reusability Estimation Model for Object Oriented Software

Namrata Chopra[1], Namrata Dhanda[2]

Research Scholar, CSE, Goel Institute of Technology and Management, Lucknow, India[1]

Head of Department, CSE, Goel Institute of Technology and Management, Lucknow, India[2]

Abstract: Estimating Reusability near the beginning in the software development life cycle, particularly at design phase, may help the designers to integrate required improvement and corrections for improving overall quality of the developed software. This research paper proposes a framework for reusability estimation that provides step by step solution for reusability estimation of object oriented software. As well as study developed a multivariate regression model 'Reusability Estimation Model for Object-Oriented software' at design phase of software development process. Proposed model has been mathematically validated through proper statistical measures and contextual explanation has been drawn. This study facilitates the software developers, designer and quality controllers, the inclusion of reusability estimation model to access and quantify software reusability for quality product. Reusability estimation in early stage of software development life cycle always supports the designers to improve the reusability of product class diagram and consequently the reusability of software.

Keywords: Reusability, Estimation, Design Properties, Quality Factors, Object Oriented Design.

I. INTRODUCTION

Software engineering paradigms deals with the development of major software projects and related components. As the size of software project improved, new approaches of system development life cycle come to the background. These approaches incorporate the object oriented programming principles, component based programming concept and aspect based programming concept etc. All approaches offer an enhanced view to present and develop a required software system. These approaches are extremely inspired from the real world and provide an organized and fast software development approach [1, 2]. In this proposed research work we are mainly dealing with the notion of software reusability that exists in all type of approaches. At this point the object oriented approach and component based approach are considered as the key concept of reusability.

At high stage software reusability consists of mainly two types of activities: one is the management of software components, including the specification, classification, and retrieval of existing components; the other is component integration that involves the integration of the reused components into an application. Over the past several years, a large number of methods have been developed to deal with these reuse issues. However, the lack of a seamless integration of these techniques imposes significant obstacles to achieving effective reuse.

II. SOFTWARE REUSABILITY

We begin with some basic definitions. Software reuse is the make use of existing software or software information to construct new software. Reusable assets can be either reusable software or software knowledge.

Reusability is a property of a software asset that indicates its probability of reuse [6].

Software reuse purpose is to improve software quality and productivity. Reusability is one of the major software quality factors. Software reuse is of interest because people want to build systems that are bigger and more complex, more reliable, less expensive and that are delivered on time [20, 21]. They have found traditional software engineering methods inadequate, and feel that software reuse can provide a better way of doing software engineering. A key idea in software reuse is domain engineering (product line engineering).

The basic insight is that most software systems are not new. Rather they are variants of systems that have already been built. Most organizations build software systems within a few business lines, called domains, repeatedly building system variants within those domains. This insight can be leveraged to improve the quality and productivity of the soft ware production process [7, 19]. The C++ language was also designed to encourage reuse as described in [31].

Reuse has been confirmed to present many rewards, as soon as we reuse code components and other artifacts:

- Reduce time.
- Reduce the cost of developing the product.
- Improve the productivity of the development teams.
- Improve the predictability of the development process.
- Always boost the quality and reliability of the software product.

Reuse eventually saves us time and money, and will ultimately lead to a more stable and reliable product [22-25].

DOI 10.17148/IARJSET.2015.21231

III. REUSABILITY QUALITY CRITERIA

Table 1: Reusability Quality Criteria Defined by Area Experts

S.No.	Factor	Quality Criteria	Mode
1	REUSABILITY	Structured Augment ability	Criteria of Boehm quality Mode
2		Generality Independence Self- documentation Modularity Software independence	Criteria of McCall quality Mode
3		Complexity, Concision Consistency, Generality Modularity Self-documentation Expandability, Simplicity	Criteria of Ming-Chang Lee Mode

IV. OBJECT ORIENTED DESIGN PROPERTIES

Quantification Object-oriented programming languages provide another approach to reusability. A good discussion is contained in CACM. The properties of object oriented languages that help reusability include information hiding, property inheritance, and polymorphism. Information hiding is a reusability mechanism, since those parts of a system which cannot see information that must change can be reused to (re)build the system when that information does change. Property inheritance allows new subclasses to be built on top of super classes by inheriting variables and methods of the super class. The process of inheritance encourages reuse of previously defined data attributes and procedures in a more specific manner [3, 5, 8]. Polymorphism means that operations have multiple meanings depending on the types of their arguments. Polymorphism can make reuse more flexible. Huda et al. have developed a programming environment for object-oriented programming which supports reuse of classes through the use of an expert system [22]. Object-oriented programming languages provide Reusability in using reusable objects [31]. However, it is sometimes difficult to combine operations defined by different reusable objects. Even in an object oriented environment, a major problem is that it is still difficult for users, especially those who were not involved in the development of the existing software resources, to know whether there are reusable software resources to match their needs [9, 20]. Moreover, organizations will continue to use traditional software development approaches for reasons of inertia and efficiency as well as because of the large installed.

Table 2: Object Oriented Design Properties and Their Benefits

Design Attributes	Description	Achievements
Design Size	A measure of the number of classes used in a design.	• Representative of design quality • characteristics of object oriented development
Abstraction	A measure of the generalization, specialization view of the design.	• Effectiveness • Functionality
Encapsulation	Hides the implementation details.	• Reduces complexity • Reusability • Easier testing and maintenance
Coupling	Defines the interdependency of an object on other objects in a design.	• Low Coupling Provides • Reusability • Good understandability
Cohesion	Assesses the relatedness of methods and attributes in a class.	• Reusability • Understandability
Inheritance	Allows child classes inherit the characteristics of existing parent	• Reusability • Eliminates redundant code
Polymorphism	Ability to take more than one form.	• Provide abstraction • Eliminates redundant code
Messaging	A count of the number of public methods that is available as services to other classes. This is a measure of the services that a class provides.	• Functionality • Effectiveness

DOI 10.17148/IARJSET.2015.21231

Object oriented technology have turn into the most accepted and recognizable concept in software industry. Object oriented notion is now broadly used by software industry [32]. Despite the truth that technology is not grown-up enough from testing point of view [10-12], almost everybody speak about it, approximately everyone state to be doing it and nearly everyone says that it is superior than conventional function oriented design [13]. For the reason that most of the center of the object oriented approach to software development has been on analysis and design phase, only a small research studies have been faithful to explore the concept of Reusability in object oriented system. Object oriented ideology direct the designers what to carry and what to stay away from. Numerous measures have been defined so far to estimate object oriented design [14-17]. There are various important themes of object orientation that are identified to be the foundation of internal quality of object oriented design and support in the perspective of estimation. These themes significantly take account of cohesion, coupling, inheritance, and encapsulation [26, 27].

V. DATA COLLECTION

Data used during the study has taken from Bansiya et al. [18]. This dataset has used in regression analysis for establishing the Reusability model, while Reusability as dependent variable.

VI. RELATIONSHIP OF REUSABILITY WITH DESIGN PROPERTIES

After an in depth assessment of existing literature on the topic [28-30], we established a relationship amongst object oriented design properties and reusability as shown in Figure 1

$$\boxed{Y = \mu + \beta_1 * X_1 + \beta_2 * X_2 + \;\ldots\ldots + \beta_n * X_n + \varepsilon \qquad (1)}$$

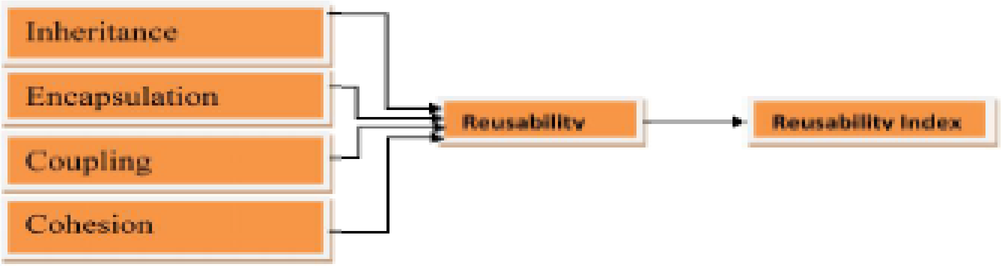

FIG.1. CORRELATION ESTABLISHMENT BETWEEN REUSABILITY AND DESIGN PROPERTIES

MODEL DEVELOPMENT

Estimation of class diagram's Extendibility and Reusability is precondition for the accurate Reusability estimation model.

Therefore before developing Reusability estimation model, the study has developed two models for Extendibility and Reusability. In order to set up all the two models following multivariate linear model (1) has selected.

VII. REUSABILITY ESTIMATION MODEL

In order to create a multivariate model for Reusability of class diagram, metrics listed in [33], will play the role of independent variables whereas Reusability will be in use as dependent variable. The data used for developing Reusability model is taken from [33]. The correlation in the middle of Reusability factors and object oriented characteristics has been established as depicted in Figure 1. Using SPSS, values of coefficient are calculated and Reusability model is formulated as given below:

Table 3: Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	-247.677	264.553		-.936	.521
	Inheritance	58.294	138.764	.228	.420	.747
	Encapsulation	71.660	348.396	.155	.206	.871
	Coupling	26.704	22.778	2.249	1.172	.450
	Cohesion	468.141	685.180	1.485	.683	.618

a. Dependent Variable: Reusability

DOI 10.17148/IARJSET.2015.21231

Reusability= -247.677 + 58.294* Inheritance +71.660* Encapsulation+26.704* Coupling +468.141* Cohesion

Table 4: Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	.945[a]	.893	.463	28.39374

VIII. EMPIRICAL VALIDATION

Empirical validation is a very important stage of proposed research work. Taking view of this exactness, realistic validation of the reusability evaluation model has been performed using sample tryouts. In order to authenticate Reusability model the data has been taken from [19].

Speraman's Coefficient of Correlation $\mathbf{r_s}$ was used to check the significance of correlation among calculated values of Reusability using model and it's 'Known Values'.

The 'r_s' was estimated using the method given as under: Speraman's Coefficient of Correlation

$$r_s = 1 - \frac{6\sum d^2}{n(n^2-1)} \qquad -1.0 \le r_s \le +1.0$$

'd' = difference between 'Calculated ranking' and 'Known ranking' of Reusability. n = number of projects (n=10) used in the experiment

- $\mathbf{r_s}$ **above±.5636** means significant results.

Table5: Computed Ranking, Actual Ranking and their Relation

Projects ↓	Reusability Ranking		d^2
	Computed Rank using Proposed Model	Known Rank given by Experts	
P1	4	2	4
P2	5	1	16
P3	8	10	4
P4	2	4	4
P5	10	9	1
P6	9	8	1
P7	3	5	4
P8	6	6	0
P9	1	3	4
P10	7	7	0

Table 6: Empirical Results

$\sum d^2$	38
r_s	0.769697
r_s> tabulated	±.0.5636

IX. CONCLUSION

The correlation values between reusability for object oriented software through developed Reusability Model and known values are shown in above table. Pairs of these values with calculated correlation value $\mathbf{r_s}$ are greater than tabulated rs value i.e. ±.5636. The correlations are up to standard with high degree of confidence, i.e. up to 95%. Therefore we can conclude without any loss of generality that Reusability Estimation model values are really reliable, significant and applicable.

REFERENCES

[1] Basili, V.R. and Rombach, H.D. (2012) Towards a Comprehensive Framework for Reuse: A Reuse-Enabling Software Evolution Environment. Technical Report CS-TR-2158, University of Maryland, Maryland.

[2] Tracz, W. (2010) Confessions of a Used Program Salesman: Institutionalizing Software Reuse. Addison-Wesley.

[3] Natasha Sharygina , James C. Browne, and Robert P. Kurshan, "A Formal Object-Oriented Analysis for Software: Design for Verification", 2011, pp:1-15

[4] 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.

[5] Krueger, C.W. (2009) Software Reuse. ACM Computing Surveys, **24**, 131-183.

[6] 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

[7] IEEE Press (1990) IEEE Standard Glossary of Software Engineering Technology. ANSI/IEEE Standard 610.12-1990.

[8] Nikolaos Tsantalis, Alexander Chatzigeorgiou, "Predicting the Probability of Change in Object-Oriented Systems", IEEE Transactions on Software Engineering, VOL. 31, NO. 7, July 2005, pp: 601-614.

[9] 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.

[10] Freeman, P. (1983) Reusable Software Engineering Concepts and Research Directions. In Tutorial: Software Reusability, 10-23.

[11] ISO (2001) ISO/IEC 9126-1: Software Engineering—Product Quality—Part-1: Quality Model. Geneva.

[12] Stephanie Gaudan , Gilles Motet and Guillaume Auriol , "A new structural complexity metrics applied to Object Oriented design assessment", http://www.lesia.insa-toulouse.fr/~motet/papers /2007 _ISSRE_GMA.pdf.

[13] Everald E. Mills, "Software Metrics", SEI Curriculum Module SEI-CM-12-1.1, Software Engineering Institute, Dec 1988, pp: 1-43.

[14] Haifeng Li Minyan Lu Qiuying Li , "Software Metrics Selecting Method Based on Analytic Hierarchy Process", Sixth International Conference on Quality Software, 2006. QSIC 2006, 27-28 Oct. 2006, pp: 337 – 346, ISSN: 1550-6002, ISBN: 0-7695-2718-3.

[15] 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

[16] Cooper, J. (1994) Reuse-the Business Implications. In Marciniak, 1071-1077.

[17] Offutt, R. and R. Alexander, (2001): A fault Model for Subtype Inheritance and Polymorphism. In 12th International Symposium, Software Reliability Engineering, Nov 27-30, 2001, IEEE, pp. 84-93.

[18] Jagdish Bansiya, "A Hierarchical Model for Object Oriented Design Quality Assessment", IEEE Transaction of Software Engineering, Volume 28, No. 1, January 2002, and pp: 4-17

[19] Binder, R.V. (1994) Design for Testability in Object-Oriented Systems. Communications of the ACM, **37**, 87-101. http://dx.doi.org/10.1145/182987.184077.

[20] 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.

[21] Yong Cao Qingxin Zhu. Improved metrics for encapsulation based on information hiding. DOI: 10.1109/ICYCS.2008.76, The 9th International Conference for Young Computer Scientists, IEEE computer society 2008, p: 742-724.

[22] 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

[23] Sch aril N., Black Andrew P., Ducasse S. Object oriented Encapsulation for Dynamically Typed Languages. OOPSLA 2004, ACM, pp: 130–139.

[24] 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.

[25] Usha Chhillar, Shuchita Bhasin , " A New Weighted Composite Complexity Measure for Object-Oriented Systems", International Journal of Information and Communication Technology Research Volume 1 No. 3, July 2011,pp: 101-108, ISSN-2223-4985.

[26] 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

[27] Dromey, R.G.: A Model for Software Product Quality. IEEE Transaction on Software Engineering 21(2), 146–162 (1995).

[28] 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

[29] Fiondella, L.; Gokhale, S.S., "Software quality model with bathtub-shaped fault detection rate" Reliability and Maintainability Symposium (RAMS), 2011 Proceedings - Annual , 24-27 Jan. 2011,pp: 1 – 6, ISBN: 978-1-4244-8857-5.

[30] 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

[31] Mohan, K.K.; Verma, A.K.; Srividya, A., "Software effectiveness estimation through black box and white box testing at prototype level ", 2nd International Conference on Reliability, Safety and Hazard (ICRESH), 14-16 Dec. 2010, pp: 517 - 522, ISBN: 978-1-4244-8344-0.

[32] Dromey, R.G. (1996) Concerning the Chimera (Software Quality). IEEE Software, **13**, 33-43. http://dx.doi.org/10.1109/52.476284