Testability Estimation Model (TEM^{OOD})

Mohd Nazir and Raees A. Khan

{Mohd Nazir,Raees A. Khan}@Springer.com

Abstract. Testability analysis early in the development process is a criterion of crucial importance to software developers, designers and the managers. Early estimation of testability, exclusively at design phase assists designers to improve their designs before the coding starts. Consequently, it significantly reduces rework during and after implementation, as well as designing effective test plans, better project and resource planning. An effort has been put forth in this paper to identify the major factors contributing in testability estimation at design phase. Further, a model is developed to quantify software testability in design phase. Furthermore, the correlation of Testability with these factors has been tested and justified with the help of statistical measures.

Keywords: Testability, OO Design Constructs, Testability Factors, Testability Estimation.

1 Introduction

Software industry now exists in an environment more turbulent than ever before. It is essential for the companies to develop the processes smart enough to be easily adapted in the fast changing business requirement and then to be able to compete in the highly complex market environments. Software Engineering today needs best practices and tools to support developers to develop software that is as fault free as possible. Many tools and methods exist today, but the question is, if and how they are used and more importantly in which circumstances they are used and why [1]. Testability has been identified as one of the major issues in the field of Software Engineering. It provides insights that are found to be very much valuable during design, coding, testing and quality assurance [2][13].

Testability analysis early in the development process is a criterion of crucial importance to software developers, designers and the managers. However, most of the mechanisms available for testability estimation may be used in the later phases of system development life cycle [5][10]. It is a well understood fact that a decision to change the design at a later stage in order to improve testability proves to be very expensive and error-prone. On the other hand, early estimation of testability, exclusively at design phase assists designers to improve their designs before the coding starts. It may significantly reduce rework during and after implementation, as well as designing effective test plans, better project and resource planning.

The framework can estimate testability of object oriented software early in design phase, and it has been validated with a sound theoretical basis [2]. An effort has been

N. Meghanathan et al. (Eds.): CCSIT 2012, Part III, LNICST 86, pp. 178–187, 2012.

put forth to identify the major factors contributing in testability estimation at design phase. It has been concluded that Understandability and Complexity are the two factors affecting software testability estimation in design phase. Taking into account the significance of their contribution, a model has been developed to quantify software testability in design phase. Furthermore, test is performed to justify the statistical correlation of Testability with Understandability and Complexity.

2 OO Design Constructs

Object-oriented analysis and design have become the popular concepts in software development. These methodologies focus on objects as the primary agents involved in a computation. The process of early analysis and design provides the information needed to assess the quality of a design's classes, structure, and the relationships before they are committed to an implementation. A good object oriented design needs design rules and practices that must be known and used. Their violation will eventually have a strong impact on the quality attributes. Object oriented principles guide the designers what to support and what to avoid.

Several measures have been defined so far to estimate object oriented design. Cohesion, coupling, encapsulation and inheritance are the essential themes of object orientation that are commonly known to be the basis of internal quality of object oriented design and support in the context of measurement. Cohesion is related to the internal consistency of the parts in the design; it can be used to identify the poorly designed classes. Coupling indicates the relationship among the modules. Inheritance is the sharing of attributes and operations among classes based on a hierarchical relationship; it occurs at all levels in the hierarchy. Encapsulation is a mechanism that realizes the concept of abstraction and information hiding; it hides internal specification of an object and shows only the external interface.

3 Testability Factors

Testability is now established to be a distinct software quality characteristic. The notion of software testability has been a subject to a number of different interpretations by the experts. However, testability has always been an elusive concept and its correct measurement is a difficult exercise [10]. Further, there is little consensus among practitioners about 'what aspects are actually related to software testability'. Hence, it is difficult to get a clear view on all the potential factors that can affect testability and the dominant degree of these factors under different testing contexts [14]. Researchers and Practitioners have made significant amount of contributions in the direction of exploring testability factors in general and of object oriented software in particular [12][15][16][17][18][19]. It appears quite conclusive from the available literature that there is a conflict among practitioners in considering the factors while estimating testability in general and exclusively at design level.

An accurate measure of software quality depends on testability measurement, which in turn depends on the factors that can affect testability. Foregoing description

shows that there is a conflict among practitioners about considering the factors while estimating testability. Therefore, it seems highly desirable and significant to identify the factors that facilitate or hinder testability in order to get the reliable and correct measure of testability. Though, getting a universally accepted set of testability factor is only probable, an effort has been made to collect a commonly accepted set of factors that can affect testability. However, without any loss of generality, it appears reasonable to include the factors namely, complexity, controllability, Observability, understandability, traceability, built-test as testability factors.

4 Mapping between Design Constructs to Testability Factors

In order to establish a contextual impact-relationship between OO software characteristics and testability factors, the influence of OO characteristic on each factor of testability was examined by several researchers. Most of the studies focused their attempt to examine the impact of object oriented characteristics and have successfully established relationships with quality factors. However, researcher examined and assessed their impact on the particular aspect of study i.e. testability and by associatively and congruence perspective, concluded on identifying testability factors affected by object characteristics. It was observed that each of these characteristics, either have positive or negative impact on the factors that affect testability of object oriented software.

After an exhaustive review of available literature on the topic [6] [10] [11] [12] [15] 18], the relation between OO software characteristics and testability factors, shown in Figure 1 has been established. Based on this relationship, models are to be developed for getting the quantitative value of testability factors. Further, the relative significance of individual design properties that influence software testability is weighted proportionally.

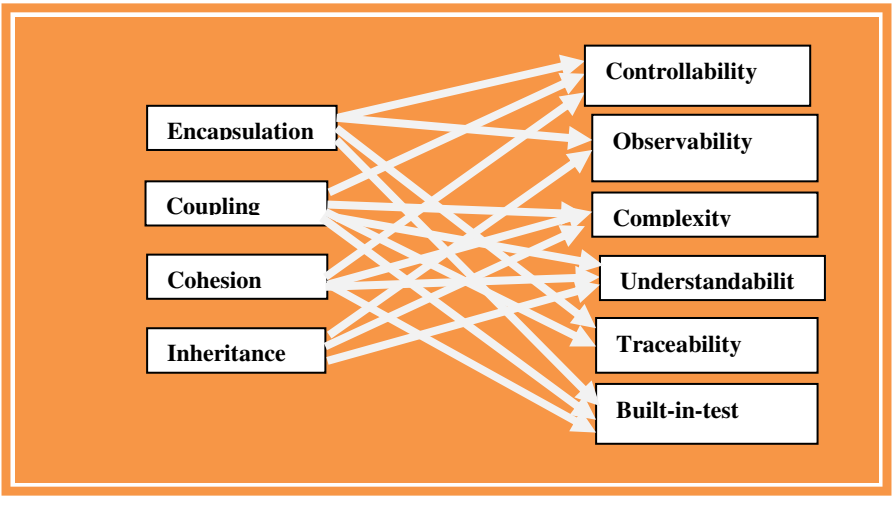

Fig. 1. Mapping Design Constructs with Testability Factors

Using the concept of multiple linear regressions, the relationship between dependent variables and multiple independent variables has been established and coefficients are determined.

5 Major Contributors

From the foregoing discussion it is revealed that the set of factors contributing testability estimation varies with the different school of thoughts. There is no universal set of testability factors to be addressed while computing testability. However, in section 3, it was established that 'a commonly accepted set of six factors of software testability include Complexity, Controllability, Observability, Understandability, Traceability and Built-in-Test'. It is further observed that each of the factors has its own contribution in making software testable. Out of these six factors, some of them have their positive impact in improving testability of object oriented software, while others have negative or negligible impact.

The only reason to establish a set of testability factors is to make the developer understand the factors to be addressed while considering testability. Therefore, it is equally mandatory to address the identified set of testability factors to integrate testability during development. But, it is also very important to know the weightage of the factor to be addressed and its contribution in developing testable software. A factor with low weightage need not be addressed because of its negligible contribution in testability estimation at design phase. Therefore, it is essential to know the contribution of each factor in order to accurately measure testability of software. Understandability and Complexity are the factors affecting software testability estimation in design phase. Therefore, these factors must be addressed well in advance while integrating testability in design phase. Models for these factors are developed in the following section.

6 Model Development

The generic quality models, [8][9] have been considered as a basis to develop the Testability Estimation Model for Object Oriented Design (TEM^{OOD}) shown in Figure 2, which involves the following steps:

i. Identification of Factors of Object Oriented Software that influences testability at Design phase.

ii. Identification of Object Oriented Design Characteristics.

iii. A means of linking of them

Based upon the relationship of the factors and testability, the relative significance of individual factors that has major impact on testability at design phase is weighed proportionally. A multiple linear regression technique has been used to get the coefficients. This technique establishes a relationship between dependent variable and multiple independent variables. The MR equation takes the following form:

$$Y = \alpha_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \ldots\ldots\ldots\ldots\ldots\beta_n X_n \quad (1)$$

Where

- Y is dependent variable,
- The X_1, X_2, X_3…….X_n are independent variables related to Y and are expected to explain the variance in Y.
- The β_1, β_2, β_3,.....β_n are the coefficients of the respective independent variables. Regression coefficient is the average amount of dependent increase/decrease when the independents are held constants.
- And α_0 is the intercept.

Using this technique, Understandability Model [3], Complexity Model [4] have been developed that are given below in equation (2) and (3) respectively.

Understandability=1.33515+0.129*CPM+.0463*COM+.03405*INM (2)

Complexity = 90.8488+10.5849*CPM-102.7527*COM+128.0856*INM (3)

It has been extensively reviewed and discussed in section 3.0 and 5.0 that Understandability and Complexity are the major factors affecting software testability estimation in design phase. Therefore, these factors must be addressed well in advance while integrating testability in design phase. Furthermore, the model of Understandability [3] and Complexity [4] forms the strong basis for development of Testability Model. Metrics of the design constructs namely Coupling (CPM), Cohesion (COM) and Inheritance (INM) are used to address the major testability factors namely Understandability and Complexity. These two factors are further used to control testability of object oriented software design. Below is the Figure 2, which gives an overview of the main idea.

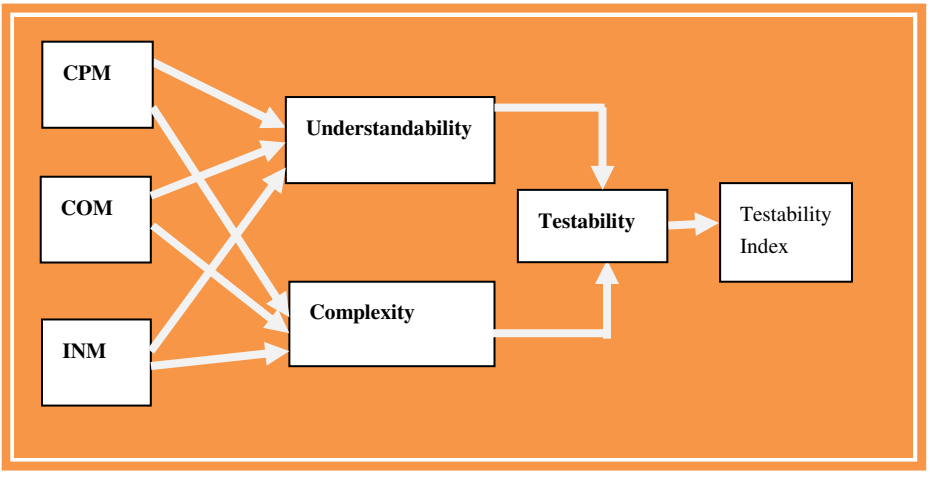

Fig. 2. Mapping Design Constructs with Major Contributors of Testability

In order to establish a model for Testability, a multiple linear regression technique discussed above in equation (1) has been used. Taking into account the impact of Coupling, Cohesion and Inheritance on testability contributors 'Understandability and Complexity', following MR equation has been formulated that can quantify Testability of object oriented design:

$$\text{Testability} = \alpha_0 + \beta_1 * \text{Understandability} + \beta_2 * \text{Complexity} \quad (4)$$

The data used for developing model given in equation (4) has been taken from [20], which consist of six industrial projects with around 10 to 20 classes. The values of 'Coupling Metrics (CPM), Cohesion Metrics (COM) and Inheritance Metrics (INM)' and the values of 'Understandability and Complexity' have been used. Using MATLAB, correlation coefficients are computed and model of Testability estimation is thus formulated as given below in equation (5).

$$\text{Testability} = -483.65 + 300.92 * \text{Understandability} - .86 * \text{Complexity} \quad (5)$$

The applications that are used in validating the multivariate linear regression model for computation of Testability have been taken from [7]. We labeled the applications as: System A, System B, System C and System D. All the systems are commercial software implemented in C++ with the number of classes as shown in table 6.0 (a).

Table 1. (a): Projects and Classes

Projects	**Classes**
System A	6
System B	5
System C	6
System D	10

The descriptive statistics and correlations between the design constructs and Understandability are given in Tables 1 (b)-(i).

Table 1. (b): Descriptive Statistics for System A

	Min	Max	Mean
Understandability	1.65	2.39	2.02
Complexity	-823.34	-309.57	-566.45
Testability	-248245	-93643	-170944

Table 1. (c): Correlation Analysis for System A

	Testability	Understandability	Complexity
Testability	1	.71	.99
Understandability		1	.71
Complexity			1

Table 1. (d): Descriptive Statistics for System B

	Min	Max	Mean
Understandability	1.87	2.55	2.21
Complexity	-1726.95	-504.49	-1115.72
Testability	-520158	-152299	-336228.5

Table 1. (e): Correlation Analysis for System B

	Testability	Understandability	Complexity
Testability	1	.94	.99
Understandability		1	.93
Complexity			1

Table 1. (f): Descriptive Statistics for System C

	Min	Max	Mean
Understandability	2.43	5.54	3.98
Complexity	-3862.72	-577.75	-2220.23
Testability	-1162865	-174344	-668604.50

Table 1. (g): Correlation Analysis for System C

	Testability	Understandability	Complexity
Testability	1	.95	.99
Understandability		1	.94
Complexity			1

Table 1. (h): Descriptive Statistics for System D

	Min	Max	Mean
Understandability	3.10	6.83	4.96
Complexity	-5023.94	-2447.64	-3235.79
Testability	2726.41	5894.61	4310.51

Table 1. (i): Correlation Analysis for System D

	Testability	Understandability	Complexity
Testability	1	.81	.84
Understandability		1	.75
Complexity			1

Table 1. (j): Correlation Analysis Summary

	Testability X Understandability	Testability X Complexity
System A	.71	.99
System B	.94	.99
System C	.95	.99
System D	.81	.84

Table 1.(j) summarizes the results of the correlation analysis for Testability model, which shows that for all the Systems, both Understandability and Complexity are highly correlated with Testability.

7 Hypothesis Testing of Coefficient of Correlation

An observed coefficient of correlation of Understandability and Complexity with Testability strongly indicates the higher significance of considering both the factors (Understandability and Complexity) for making a prediction of software testability in design phase. Further to justify the claim, a test to determine the statistical significance of the correlation coefficient observed may be appropriate. For the purpose, Hypothesis testing is performed to test the significance of r (Correlation Coefficient) using the following formula:

$$t_r = \frac{r\sqrt{N-2}}{\sqrt{1-r^2}}$$

With *N-2* degree of freedom, a coefficient of correlation is judged as statistically significant when the *t* value equals or exceeds the *t* critical value in the *t* distribution table.

$H_{0(T\hat{}U)}$: Testability and Understandability are not highly correlated.

Table 2. (a): Correlation Coefficient Test for Testability and Understandability

	System A	System B	System C	System D
Testability ^ Understandability	.71	.94	.95	.81
t_r	2.01	4.77	6.08	3.90
$t_{r\text{-Critical Value}}$	2.447	2.571	2.447	2.228
$t_r > t_{r\text{-Critical Value}}$	x	√	√	√
$H_{0(T\hat{}U)}$	Accept	Reject	Reject	Reject

$H_{0(T\hat{}C)}$: Testability and Complexity are not highly correlated.

Table 2. (b): Correlation Coefficient Test for Testability and Complexity

	System A	System B	System C	System D
Testability ^ Complexity	.99	.99	.99	.84
t_r	14.03	12.15	14.03	4.37
$t_{r\text{-Critical Value}}$	2.447	2.571	2.447	2.228
$t_r > 2.44$	√	√	√	√
$H_{0(T\hat{}C)}$	Reject	Reject	Reject	Reject

Using two-tailed test at the .05 level with different degrees of freedom, it is evident from the tables 2(a) and (b), the null hypothesis is rejected (except for System 'A' of Testability and Understandability). Therefore, the researcher's claim of correlating Testability with Understandability and Complexity at design phase is statistically justified.

8 Conclusion

Software testability factors are identified and their impact on testability has been analyzed in this paper. Two of the major factors affecting software testability in design phase have been taken into account. Considering both the major factors, 'Understandability and Complexity' a model to quantify testability of object oriented design has been formulated (TEM^{OOD}) and the statistical inferences are validated for better acceptability. Hypothesis testing is performed to test the significance of r (Correlation Coefficient). The proposed model to quantify testability of object oriented software design is highly significant and correlated with software design constructs. Though the model has been validated using try-out data, but its utility may be analyzed for larger set of data. An empirical validation of the model TEM^{OOD} will be carried out as a future work.

References

1. Richard, T.: Towards Automated Software Testing: Techniques, Classification and Frameworks, Ph.D. Thesis, Blekinge Institute of Technology, Sweden (2006)
2. Nazir, M., Khan, R.A., Mustafa, K.: Testability Estimation Framework. Internaional Journal Of Computer Applications (June 2010) (accepted for publication)
3. Nazir, M., Khan, R.A., Mustafa, K.: A Metrics Based Model for Understandability Quantification. International Journal of Computing 2(4) (April 2010)
4. Nazir, M., Khan, R.A., Mustafa, K.: Complexity Quantification Model: A Metric-Based Approach. In: 4th IEEE International Conference on Advanced Computing & Communication Technologies, Panipat (October 30, 2010)
5. Nazir, M., Khan, R.A.: Testability Estimation of Object Oriented Software: A Critical Review. In: The Proceeding of International Conference on Information and Communication Technologies, Dehradun, pp. 960–962 (2007)
6. Jungmayr, S.: Identifying Test-Critical Dependencies. In: The Proceedings IEEE International Conference on Software Maintenance, pp. 404–413 (2002)
7. Genero, M., Olivas, J., Piattini, M., Romero, F.: A Controlled Experiment for Corroborating the Usefulness of Class Diagram Metrics at the Early Phases of Object Oriented Developments. In: Proceedings of ADIS 2001, Workshop on Decision Support in Software Engineering (2001)
8. Khan, R.A., Mustafa, K.: A Model for Object Oriented Design Quality Assessment. In: Process Technology Symposium, Kusadasi, Izmir, Turkey, June 28-July 2 (2004)
9. Dromey, R.G.: A Model for Software Product Quality. IEEE Transaction on Software Engineering 21(2), 146–162 (1995)
10. Mouchawrab, S., Briand, L.C., Labiche, Y.: A Measurement Framework for Object-Oriented Software Testability. Info. and Software Technology 47(15), 979–997 (2005)

11. Bruntink, M., Deursen, A.V.: Predicting class Estability using Object-Oriented Metrics. In: Proc. IEEE International Workshop on Source Code Analysis and Manipulation, pp. 136–145 (2004)
12. Lo, B.W.N., Shi, H.: A Preliminary Testability Model for Object-Oriented Software. In: Proc. International Conf. on Software Engineering, Education, Practice, pp. 330–337. IEEE (1998)
13. Voas, Miller: Improving the Software Development Process using Testability Research. IEEE Software, 114–121 (1992)
14. Zhao, L.: A New Approach for Software Testability Analysis. In: Proceeding of the 28th International Conference on Software Engineering, Shanghai, pp. 985–988 (2006)
15. Binder, R.V.: Design for Testability in Object-Oriented Systems. Communications of the ACM 37(9), 87–101 (1994)
16. Gao, J., Shih, M.-C.: A Component Testability Model for Verification and Measurement. In: Proc. of the 29th Annual International Computer Software and Applications Conference, pp. 211–218. IEEE Comp. Society (2005)
17. James, B.: Heuristics of software Testability (1999)
18. Jungmayr, S.: Testability during Design, Software Technik-Trends. In: Proceedings of the GI Working Group Test, Analysis and Verification of Software, Potsdam, pp. 10–11 (2002)
19. Mulo, E.: Design for Testability in Software Systems, Master's Thesis (2007), `http://swerl.tudelft.nl/twiki/pub/Main/ResearchAssignment/RA-Emmanuel-Mulo.pdf`
20. Khan, R.A., Mustafa, K.: Metric based Testability Model for Object Oriented Design (MTMOOD). SIGSOFT Software Engineering Notes 34(2) (March 2009)