Determining Characteristics of the Software Components Reusability for Component Based Software Development

Suryani Ismail[1], Wan M. N. Wan Kadir[2], Noor Maizura Mohamad Noor[1] and Fatihah Mohd[1]
[1]School of Informatics & Applied Mathematics,
Universiti Malaysia Terengganu, 21030 Kuala Nerus, Terengganu, Malaysia.
[2]Software Engineering Department, Faculty of Computing,
Universiti Teknologi Malaysia, 81310 Johor Baharu, Malaysia.
sue@umt.edu.my

***Abstract*—Nowadays, due to the availability of many alternatives of common-off-the-shelf software components, Component-based Software Development (CBSD) is becoming a popular approach to software development. CBSD is the software development with the assembly of existing software components. There are many characteristics and sub characteristics for software component reusability available today. The challenge is how to determine the suitable characteristics and sub characteristics reusable component for CBSD. The aim of this study is to determine the suitability of characteristics and sub characteristics for software component reusability for CBSD. The survey is conducted among of software reuse practitioners at Universiti Malaysia Terengganu. The finding from the empirical study conducted that involves software developers and practitioners as the respondents will be used in development of metrics for reusable component. This metrics can be used to measure the reusable component for CBSD.**

***Index Terms*—Component Based Software Development; Empirical Study; Reusable Component; Software Component.**

I. Introduction

Currently, component based software development (CBSD) is becoming a popular approach to the development of software. It is the new approach of software development, utilizing the assembly of existing software components. CBSD aims to make the most reuse of present software artifacts. Although there has been a significant interest in component reuse since the early 1980s, it was only grown into a recognized practical in past few years and economical technique to software development [1]. Many organizations implementing CBSD as their software development model in order to decrease cost of development, reduce market time and improved the quality of the software reuse [2].

CBSD is a procedure that highlights the design and building systems using design of reusable software components [3]. In the process of software development, developers may use an existing software components with a small or without any modification so that the development times are reduced. Figure 1 shows Lego as an example of a component based approach. It provides a set of building blocks in a variety of shapes and colors. Lego is peddled in boxes that have a number of blocks that can be composed to create up toys such as trains, cars, and airplanes [4]. System development with components mostly focuses on objects that can be simply reusable and relationships among the objects; beginning from the requirements of system and from the ease of access of existing components [5].

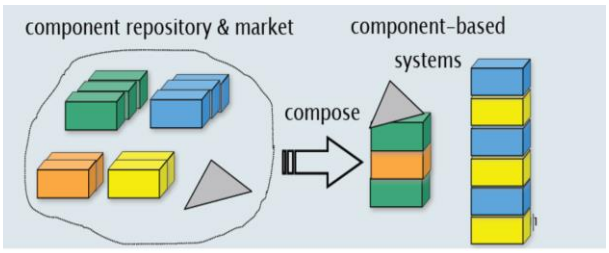

Figure 1: Concept of Component-based software engineering [4]

Software development can be categorized based on their use in the CBSD process; i) adapted components, ii) assembled components, iii) update components [3]. These components have been reused in the software development based on their types of need for CBSD processes.

Currently, there are numerous types of research in CBSD which can be group into seven (7) categories; component modeling and specification, retrieval techniques and specification matching, generative approach to component development, adaptation techniques, coordination and composition languages, verification, testing and certification and configuration management. In CBSD, new software developments always employ software reuse concepts in general and software component reuse in specific.

Shambhu and Mishra [6] stated that software component reuse helps reducing production cost and time in a new software development. CBSD is a techniques used by researchers and practitioners to improve the quality of software systems with lower cost and shorter time to market, where it uses existing reusable components instead of writing from scratch [5]. There are many characteristics of component reusability such as portability, adaptability/legibility, understandability and confidence that provided significant support for facilitating component for reuse in CBSD.

The aim of this study is to determine the suitability characteristics and sub characteristics for software component reusability for CBSD that exist from three models; Reboot model [7], Cardino model [8] and Washizaki model [9]. The quantitative approach was used for this research via a survey. The survey is conducted at Information

Technology Management Center, University Malaysia Terengganu (UMT). The results of the survey indicated that the characteristics are appropriated to be used for measure reusable components in CBSC were; i) portability, ii) adaptability (flexibility), and iii) understandability.

This paper is organized into four sections. In Section II, the related works of study is presented. The component evaluation characteristics are elaborated in Section III. The finding and discussion are presented in Section IV. Lastly, the conclusion and the future study subjects are drawn in Section V.

II. RELATED WORKS

The reuse of components in many systems is one of the core contributions of CBSD. The idea of componentizing software had been suggested as a way of tackling the software crisis since 1969 [10]. In this way, a component is developed only once, and could be used in other application which can reduce development effort and indirectly increase the cost-saving for software development process [5].

Biggerstaff and Freeman [11,12] defined software component as a direct reuse of the software. Jacobson [13], proposed that a component is an implementation abstraction, where it is developed and packaged based on the aim of reuse that it differs from code fragments, modules and programs. In practice, Heinman and Council [14] defined a software component as an existing piece of software written with reuse that can be deployed with little or no modification. In general, components include interface, computational, memory, manager, and controller. Components also can be distributed in the form of an object codes and reused in another environment by downloading it online.

One of the popular basic concepts of component based software development is reusability and how often a component is used in new software development [15]. Software reusability is defined as the use of existing software artifacts to build a new software [16]. Quality, productivity and maintainability of new software can be improved using the concepts of component reuse. A reusable component are defined in three categories namely, black box reuse, glass box reuse and white box reuse. In order to recognize the components reusability according to their quality, original, and reusability, the component evaluation approaches are essential to evaluate the components with reuse or development for reuse.

From the review of literature related to component evaluation, common approaches used in component evaluation are product line component (PLC) approach [17], original component (OC) approach [18] quality component (QC) approach [19] and reusable component (RC) approach [20]. It was found that the evaluation of components primarily focuses on their characteristics, sub characteristics, and metrics to support software component evaluation.

The techniques to define the metrics in reusable component are semi-formal technique [9] and informal technique [21]. Compared to OC and QC approach that used one level of validation, RC used two level of validation to validate the metrics which is anecdotal [21] and industrial experiment. Based on Reuse Based Object Oriented Technology (REBOOT) model [7], the RC approach has been proposed for evaluating reusable components. This approach includes four components: understanding, adaptability (flexibility), portability, and confidence (probability). Every characteristic has sub characteristics. The purpose of the RC is to measure the reusability of components in order to realize the reuse of component effectively and to identify the best components in terms of their reusability. Table 1 shows the characteristics for component evaluation approaches described in this section.

Table 1
The Characteristics of Component Evaluation Approaches

Evaluation Approaches	Characteristics
Product line components (PLC)	i) Understandability ii) Component replaceability iii) Functional commonality iv) Applicability v) Nonfunctional commonability vi) Variability richness vii) Tailorability
Original components (OC)	i) Functionality ii) Reliability iii) Usability iv) Efficiency v) Maintainability
Quality components (QC)	i) Functionality ii) Reliability iii) Usability iv) Efficiency v) Maintainability vi) Portability
Reusable components (RC)	i) Portability ii) Adaptability (flexibility) iii) Understandability iv) Confidence/ probability

III. COMPONENT EVALUATION CHARACTERISTICS

Based on previous studies that are simplified in Table 1, common approaches used in component evaluation are PLC OC, QC, and RC approach. From these approaches, there are fifteen (15) characteristics for component evaluation are noted, namely understandability, component replaceability, functional commonality, applicability, nonfunctional commonability, variability richness, tailorability, functionality, reliability, usability, efficiency, maintainability, portability, adaptability (flexibility), understandability and confidence/ probability [8-10, 22-24].

In this study, RC approaches with four (4) characteristics namely portability, adaptability (flexibility), understandability and confidence/ probability are chosen for component evaluation. The evaluation is done by applying experimentation which is a classical scientific technique that can be used to evaluate empirical study [25,26].

Controlled experimentation is defined as: “A replicated experiment is conducted in a smaller artificial environment, but in realistic situations compared to the real projects” [27]. The goal of the controlled experimentation is to evaluate the component reusability evaluation approach in an academic setting. Participants with software engineering background were selected as experimental respondents. Respondents were guided to complete the required experimental tasks. In specific, the experiment results were obtained by highlighting the experimentation context and design procedures, quantitative analysis results.

Therefore a survey to determine the characteristics and sub characteristics was done among of software reuse practitioners. A set of questionnaire has been designed based on Linkert scale parameters [28]. The set of questionnaires was distributed among eighteen (18) respondents who are considered to be the expert software component users. The respondents consists of eight (8) officers from the Application Development Sections from Information Technology Management Center, University Malaysia Terengganu (UMT) and ten (10) computer science lecturers at the School of Informatics and Applied Mathematics, UMT. From the survey, the results for suitable characteristics and sub characteristics for RC were determined. The following section elaborated the findings of the survey.

IV. Result And Discussion

Table 2 and Figure 2 show the results of a survey of four main characteristics of RC; i) portability, ii) adaptability (flexibility), iii) understandability and iv) confidence/ probability. The results indicate that the highest mean value for understandability is 3.11 followed by portability, adaptability and confidence; with the corresponding values of 2.94, 2.91 and 2.85 respectively. Since confidence has the lowest results based on the survey, it can be concluded that the appropriate characteristics that are going to be selected for the study are understandability, adaptability and portability.

Table 2
Descriptive Statistics for Characteristics

Characteristics	N	Mean	Std. Deviation
Understandability	18	3.11	0.29
Adaptability	18	2.91	0.49
Portability	18	2.94	0.35
Confidence	18	2.85	0.54
Valid N (list wise)	18		

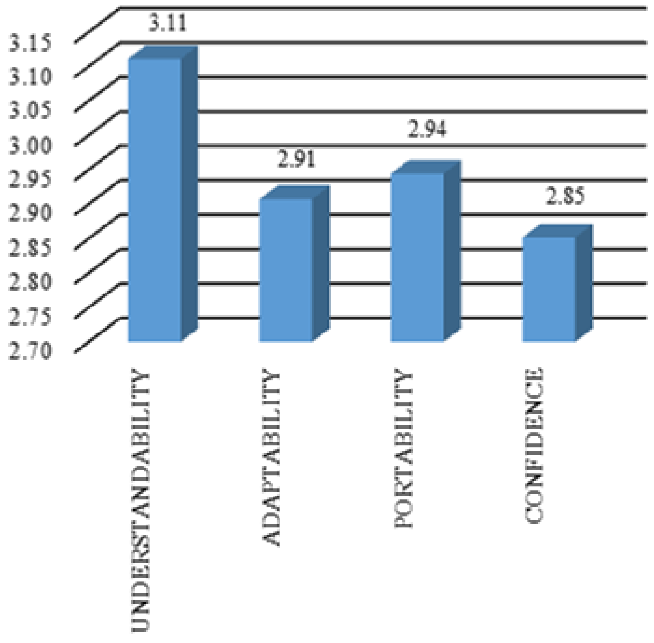

Figure 2: Mean for Each Characteristics

Table 3 shows the results of the survey according to three sub characteristics for understandability. The results show that the highest mean value for the documentation level is 3.42 followed by observability with the mean of 3.22 and complexity, with the value of 2.69. From the results, it can be concluded that the appropriate sub characteristics to be chosen for the study are documentation level and observability.

Table 3
Descriptive Statistics Sub Characteristics for Understandability

Sub Characteristics	N	Mean	Std. Deviation
Documentation level	18	3.42	0.49
Observability	18	3.22	0.35
Complexity	18	2.69	0.77

Table 4 shows the results of the survey on three sub characteristics for adaptability. The results show that customizability has the highest mean value of 3.50 followed by modularity and generality with corresponding mean of 2.72 and 2.5 respectively. From the results it can be concluded that the appropriate sub characteristics to be used for the study is customizability.

Table 4
Descriptive Statistics Sub Characteristics for Adaptability

Sub Characteristics	N	Mean	Std. Deviation
Customizability	18	3.50	0.54
Modularity	18	2.72	0.75
Generality	18	2.50	0.79

Table 5 shows the results of four sub characteristics for portability. Based on the table, compliance will be chosen as the sub characteristic for adaptability since it has the highest mean value of 3.17 followed by external dependencies with a mean value of 3.06. The other two sub characteristics; deployability and replaceability only scored mean values of 2.94 and 2.61 respectively. From the results it can be concluded that the appropriate sub characteristics selected for study is external dependency because based on previous study it is common use for measure reusability.

Table 5
Descriptive Statistics Sub Characteristics for Portability

Sub Characteristics	N	Mean	Std. Deviation
External dependency	18	3.06	0.54
Compliance	18	3.17	0.38
Deployability	18	2.94	0.54
Replaceability	18	2.61	0.63

Table 6 shows the results of three sub characteristics for confidence. Sub characteristic maturity has the highest mean value of 3.11 followed by error tolerance with a mean value of 2.69 and observed reliability of 2.75.

This study elaborates that mean value belong to confident was not suitable to be selected for RC characteristic since it showed the lowest value from the survey. Furthermore, the previous study stated confidence is more suitable to be chosen for the evaluation of the creation of new software reuse framework [8].

Table 6
Descriptive Statistics Sub Characteristics for Confidence

Sub Characteristics	N	Mean	Std. Deviation
Maturity	18	3.11	0.58
Error tolerance	18	2.69	0.69
Observed reliability	18	2.75	0.65

From the analysis, three of four characteristics are selected, such as understandability, adaptability, and portability that have been employed in proposed model for RC (see Figure 3). In this proposed model there are two terms being employed; i) the reusability characteristics, and ii) sub characteristics that are organized in an evaluation of component reusability level.

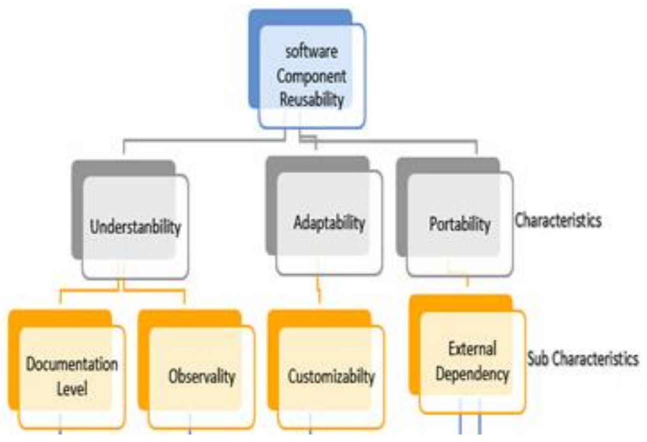

Figure 3: Proposed Model for RC

V. Conclusion

The survey indicates an existing evaluation [22,23] that among four characteristics, only three are suitable to be used as a measurement of RC components. These characteristics are understandability, adaptability, and portability has been employed in proposed component reusability evaluation approach for CBSD. Confidence was not suitable to be selected for RC characteristic since it showed the lowest value from the survey and the previous study stated confidence is more suitable to be chosen for the evaluation of the creation of new software reuse framework.

In future, this proposed approach can be used to develop metrics suite for measure the reusable components and level of reusability for RC.

Acknowledgment

The authors would like to express their deepest gratitude to Research Management Center (RMC), Universiti Teknologi Malaysia (UTM), and Ministry of Higher Education Malaysia (MOHE) for their financial support under Research University Grant Scheme (Vot number Q.J130000.2516.11H71).

References

[1] I. Sommerville, *Software Engineering*. United States of America: Addison Wesly, 2001.
[2] C. Szyperski, D. Gruntz, and S. Murer, *Component Software Beyond Object Oriented Programming*. New York: Adision Wesley, 2002.
[3] R. S. Pressman, *Software Engineering: A practitioner's Approach*. New York: Mc Graw Hill International Edition, 2001.
[4] Basic Concepts of Component-based software. Available at http://www.idt.mdh.se/kurser/cdt501/2008/lectures/book%20Basic%20Concepts%20of%20CBSE.pdf.
[5] A. I. Khan, Noor-ul-Qayyum, and U. A. Khan, "An improved model for component based software development," *Scientific & Academic Publishing, Software Engineering*, vol. 2, pp. 138-146, 2012.
[6] K. J. Shambhu and R. K. Mishra, "Accessing software quality for component-based software through trustworthiness and dependability analysis," *Internal Journal of Development Research*, vol. 5, no. 4, pp. 4259-4261, Apr. 2015.
[7] A. Jedlitschka, M. Ciolkowski, and D. Pfahl, "Reporting experiments in software engineering," in *Guide to Advanced Empirical Software Engineering*, F. Shull, J. Singer, and D. I. K. Sjoberg, Eds. London: Springer, 2008, pp. 201-228.
[8] G. Cardino, F. Baruchelli, and A. Valerio, "The evaluation of framework reusability," *ACM SIGAPP Applied Computing Review - Special Issue on Frameworks and Patterns in Software Reuse*, vol. 5, no. 2, pp. 21-27, 1997.
[9] H. Washizaki, H. Yamamoto, and Y. Fukazawa, "A metrics suite for measuring reusability of software components," in *Proc. Software Metrics Symposium*, 2003, pp. 211-223.
[10] M. McIlory, *Mass Produced Software Components, in NATO Conference Software Engineering*. 1969, Petrocelli/Charter: New York, 1969.
[11] T. Biggerstaff and C. Richter, "Reusability framework, assessment, and directions," *IEEE Software*, vol. 4, no. 2, pp. 41-49, 1989.
[12] P. Freeman, "Reusable software engineering: Concepts and research directions," in *Proc. ITT Proceedings of the Workshop on Reusability in Programming*, 1983, pp. 137.
[13] I. Jacobson, *Object-Oriented Software Engineering: A Use Case Driven Approach*. Redwood City CA, USA: Addison-Wesley. 2004.
[14] G. Kotonya, I. Sommerville, and S. Hall, "Towards a classification model for component-based software engineering research," in *2003 Proceedings 29th Euromicro Conference*, 2003, pp. 43-52.
[15] K. Tyagi and A. Sharma, "Significant factors for reliability estimation of component based software systems," *Journal of Software Engineering and Applications*, vol. 7, no. 11, pp. 934, 2014.
[16] B. Jalender, A. Govardhan and P. Premchand. "Designing code level reusable software components," *Int. Journal of Software Engineering & Applications*, vol. 3, no. 1, pp. 219-229, 2012
[17] P. Clements, *Software Product Lines*. USA: Addison-Wesley, 2002.
[18] A. S. Andreou and M. Tziakouris, "A quality framework for developing and evaluating original software components," *Information and Software Technology Journal*, vol. 49, no. 2, pp.122-141, 2006.
[19] M. Bertoa and A. Vallecillo. "Quality attributes for COTS components," in *6th International Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE'2002)*, 2002, pp. 128-144.
[20] H. Washizaki, H. Yamamoto, and Y. Fukazawa. "A metrics suite for measuring reusability of software components," in *Proceedings of the Ninth International Software Metrics Symposium (METRICS'03)*. 2003, pp. 211-223.
[21] R. Dumke and A. Schmietendorf, "Possibilities of the description and evaluation of software components," *Metrics News*, vol. 5, no. 1, pp. 13-26, 2000.
[22] M. Goulao and F. B. Abreu, "Towards a component quality model," in *Proc. Work in Progress Session of the 28th IEEE Euromicro Conference*, 2002.
[23] J. S. Her, J. H. Kim, S. H. Oh, S. Y. Rhew, and S. D. Kim, "A framework for evaluating reusability of core asset in product line engineering," *Information and Software Technology*, vol. 49, no. 7, pp. 740-760, 2007.
[24] A. Alvaro, E. S. De Almeida, and S. L. Meira, "A software component quality model: A preliminary evaluation," in *32nd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO'06)*, 2006, pp. 28-37.
[25] D. I. K Sjoeberg, J. E Hannay, O. Hansen, V. B. Kampenes, A. Karahasanovic, N. K. Liborg, and A. C. Rekdal, "A survey of controlled experiments in software engineering," *IEEE Transactions on Software Engineering*, vol. 31, no. 9, pp.733-753, Sept. 2005.
[26] C. Wohlin, P. Runeson, M. Host, M. C. Ohlsson, B. Regnell, B. and A. Wesslén, *Experimentation in Software Engineering*. Berlin Heidelberg: Springer, 2012.
[27] M. V. Zelkowitz, and D. R. Wallace, "Experimental Models for Validating Technology," *Computer*, vol. 31, no. 5, pp. 23-31, May 1998.
[28] I. E. Allen, and C. A. Seaman. "Likert scales and data analyses." *Quality progress*, vol.40, no. 7, pp.64, 2007.