International Journal of Electrical and Computer Engineering (IJECE)
Vol. 9, No. 6, December 2019, pp. 5428~5435
ISSN: 2088-8708, DOI: 10.11591/ijece.v9i6.pp5428-5435

Development metrics measurement level for component reusability evaluation approach (CREA)

Suryani Ismail[1], Fatihah Mohd[2], Masita Abdul Jalil[3], Wan M. N. Wan Kadir[4]
[1,2,3]School of Informatics and Applied Mathematics, Universiti Malaysia Terengganu, Malaysia
[4]Software Engineering Department, Faculty of Computing, Universiti Teknologi Malaysia, Malaysia

Article Info

Article history:

Received Feb 3, 2019
Revised May 18, 2019
Accepted Jun 27, 2019

Keywords:

Component reusability
Component evaluation
Metric measurement
Reusable components
Software development
Software reuse

ABSTRACT

The study of software component reuse is rising in software development field and one of the methods used to reduce the production cost and time. Among the problems faced by software developers in component reuse, is the difficulty to determine which set of components are suitable to use in new software development. Thus, this study was conducted with the purpose; to define the characteristics of software component reusability evaluation approach (CREA) based on experienced software developer's feedback, and to estimate the measurement level for each of the predefined metric. Three characteristics and sub characteristics, namely understandability (documentation level and observality), adaptability (customizability), and portability (external dependency) were identified that have been used to develop the metrics for CREA. The result for all metrics will be used as an input to the fuzzy inference system (FIS) for measuring the reusability level of the component.

Corresponding Author:

Suryani Ismail,
School of Informatics and Applied Mathematics,
Universiti Malaysia Terengganu,
21030 Kuala Terengganu, Terengganu, Malaysia.
Email: sue@umt.edu.my

1. INTRODUCTION

In software engineering, the trend is changing from the traditional software development approach to the extension and integration with existing systems [1-4]. An ideal software component reuse technology would enable software developers to quickly use and adapt components in software development. In addition, this technology could be used in all application domains, to reduce the time and effort required to build and maintain software systems and to enhance the quality of software systems by reusing quality and reusable software components [4-6]. Software component reuse is considered as an important solution to many software engineering problems. It has been claimed to improve the productivity and the quality of software development [7-9]. Many organizations have benefited from using reusable components in reducing the time and cost of software development [10-12].

From an exhaustive study in software component reuse, it can be concluded that software component reuse is one of the important factors in facilitating software reuse in new softwar development [13, 14]. Software become hard to be developed, understood, managed, controlled and maintained without any software component reuse. Software component reuse plays an important role as it always keeps track of the relationships among the artifacts to help developers or system analysts in performing their tasks. It helps ensure that software development time could be cut down and upon a change has been made and all of impacted components have been reused, plug in into the new system and tested effectively. Among the problems faced by software engineers in component reuse is the difficulty to determine which set of components are suitable to use in new software development. Problem of feature and

component selection, if these are given a set of such components, it is hard to determine a subset that it minimizes the risk and maximize the commercial return [15].

Shambhu and Mishra [16] stated that software component reuse helps reducing production cost and time in a new software development. It posits that utilizing components reusability evaluation approach to provide significant support for facilitating component for reuse in software development. There are many characteristics of component reusability such as portability, adaptability/legibility, understandability and confidence that are mentioned by previous researchers [17-19]. The metrics of component reusability were produced based on these characteristics and sub characteristics.

Thus, in this study, a survey was given to eighteen respondents who are considered to be the expert software component users. The respondents consist of eight officers from the Application Development Sections from Information Technology Management Center, and ten computer science lecturers at the School of Informatics and Applied Mathematics. The survey was conducted with the purpose:

- To define the characteristics and sub characteristics for software component reusability/component reusability evaluation approach (CREA) based on feedback from actual and experienced software developer (expertise),
- To estimated measurement level for each of the predefined metric. The estimation of the measurement level is important to set the range of values based on information from actual users.

This paper is organized into four sections. In Section 2, the related works of study are presented. The component reusability evaluation approach (CREA) characteristics are elaborated in Section 3. The metrics definitions for proposed CREA are presented in Section 4. The finding and discussion are presented in Section 5. Lastly, the conclusion and the future study subjects are drawn in Section 6.

2. RELATED WORKS

In this section, it is discussed the related works of research; components evaluation approaches, reusable components, and characteristics of component evaluation approaches.

2.1. Component evaluation approaches

Component evaluation approaches are essential to evaluate the component with reuse or development for reuse; in order to recognize the component reusability according to their quality, originality, and reusability. According to Land et al. [20], component evaluation has two kinds of method such as; (i) component certification that is done by an independent player to deliver a trustworthy valuation of the component, and (ii) component evaluation that is implemented by a system development association.

From the review of literature related to component evaluation, common approaches used in component evaluation are Product Line Component (PLC) Approach, Original Component (OC) Approach, Quality Component (QC) Approach and Reusable Component (RC) Approach. Based on the review, it was found that evaluation of components primarily focuses on their characteristics, sub characteristics, and metrics to support software component evaluation.

2.2. Reusable component (RC)

The number of components available in the market keeps increasing nowadays. This is becoming very important to have a component evaluation approach that suit with their needs and reusability evaluation is particularly important when reusing components [17]. In this study, the reusable component (RC) approach was proposed to cater for these needs. One of the technique is used by [17] that applied semi-formal technique to define the metrics in RC. While another technique used to define metric is an informal technique [21]. Compared to OC and QC approach that used one level of validation, RC used two levels of validation to validate the metrics which are anecdotal [21] and industrial experiment [17].

Based on Reuse Based Object Oriented Technology (REBOOT) model [22], the RC approach has been proposed for evaluating reusable components. This approach includes four components i.e. understanding, adaptability/ flexibility, portability, and confidence/ probability. Every characteristic has their own 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.

2.3. Characteristics of component evaluation approaches

Referring to the previous studies, the characteristics for component evaluation approaches such as, PLC, OC, QC and RC approaches. There are fifteen characteristics for component evaluation such as functionality, component replacetability, functional commonality, applicability, non-functional commonability, variability richness, tailorability, understandability, reliability, usability, efficiency, maintainability, portability, adaptability (flexibility), and confidence/ probability [21, 23].

3. COMPONENT REUSABILITY EVALUATION APPROACH (CREA)

3.1. Software component evaluation characteristics and sub characteristics

In order to determine the characteristic and sub characteristic of the proposed approach, three models are adopted which is a Reboot model [22], Cardino model [18] and Washizaki model [17]. Reboot model consist of four characteristics and nine sub characteristics, whereas Cardino model consist of four characteristics and six sub characteristics and Washizaki model consist of three characteristics and four sub characteristics. In summary, the proposed approach combines the software reuse process model that has been modified with the characteristic and sub characteristic that adopted from three models as shown in Figure 1 [24].

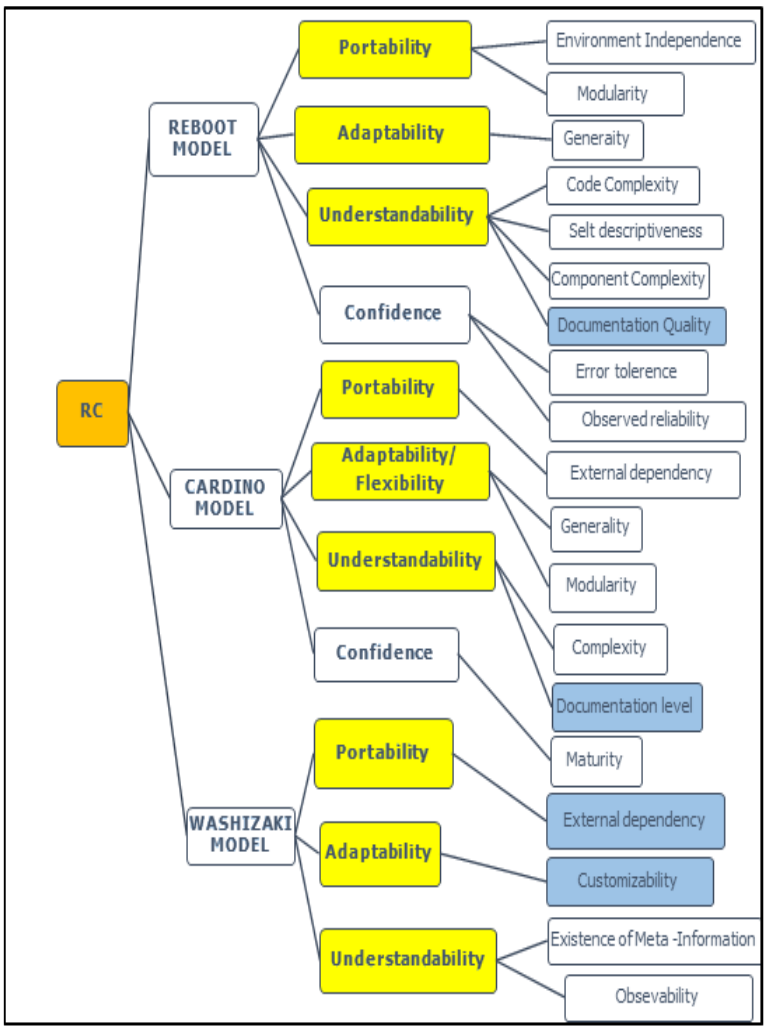

Figure 1. Characteristics and sub characteristic for component reusability

The three characteristics, namely understandability, adaptability, and portability chose for the proposed model in this study. These selected characteristics have also the sub characteristics: understandability (documentation level and observality), adaptability (customizability), and portability (external dependency). From these selected sub characteristics, five metrics for component reusability evaluation were defined, namely the value of Component Documentation Level (VCDL) for measuring the documentation level, Value of Component Observability (VCO) for measuring observability, Value of Component Customizability (VCC) for measuring customizability, and value of Component Fan-in (VCFi), and Value of Component Fan-out (VCFO) for measuring external dependency of the component. The overall characteristics, sub characteristics, and metrics for software component reusability evaluation approach (CREA) based on expertise's feedback are illustrated as Figure 2.

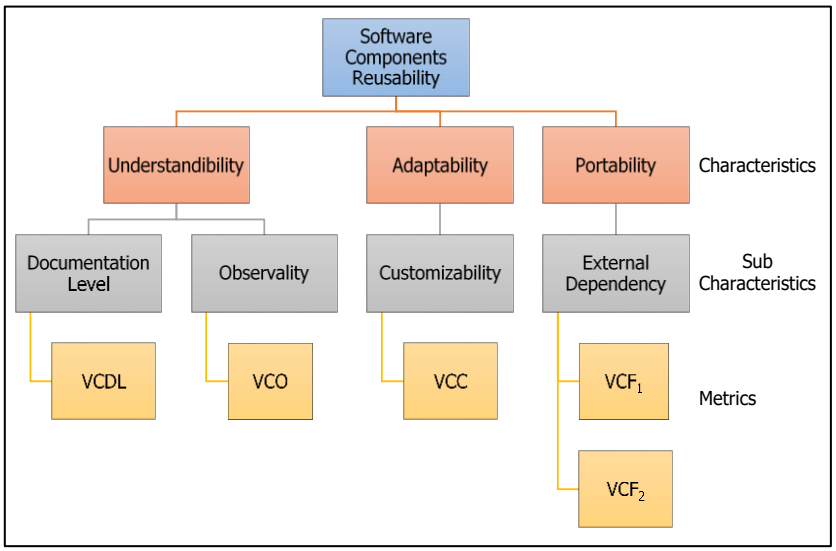

Figure 2. Characteristics, sub-characteristics and metrics for CREA

4. METRICS DEFINITIONS FOR PROPOSED CREA

From the characteristics and sub characteristics that was chosen and simplified as shown in Figure 1, five metrics for component reusability evaluation were defined, namely the value of Component Documentation Level (VCDL) for measuring the documentation level, Value of Component Observability (VCO) for measuring observability, Value of Component Customizability (VCC) for measuring customizability, and value of Component Fan-in (VCFi), and Value of Component Fan-out (VCFo) for measuring external dependency of the component as shown as Figure 2. Measured values of all metrics are always normalized between number 0 and 1 [17, 25-28]. The result for all metrics will be used as an input to the fuzzy inference system (FIS) for measuring the reusability level of the component (overall component reusability).

4.1. Value of component documentation level (VCDL)

The documentation level for the component indicates whether documentation for every method of the classes of the component is documented (available). It can specify existence of the component documentation by checking the documentation of every method that exists on classes of the component. According to Hristov et al. [29], this metric can calculate by the amount, quality, completeness and existence of legal terms and conditions of documentation in the components. Washizaki et al. [17], used existence of meta-information exists or not for the components as a metric to calculate documentation. In this study it used amount of documentation for the components to calculate this metric.

Definition: VCDL is the level of documentation for the component c. It is defined as the number of documentation for the methods that is included in the component c divided by the total number of documentation for the methods that should be in the component. VCDL for a component is computed by (1):

$$VCDL(c) = \frac{Nd(c)}{Ad(c)}, \quad Ad(c) > 0 \qquad (1)$$

$$0, \; Ad(c) <= 0$$

where, c is a component/package, Nd(c) is presented to a number of documentation for methods in c, and Ad(c) is a total number of methods that should be in c.

4.2. Value of component observability (VCO)

The important thing contributing to the observability for the component is the number of read methods provided by a component [17]. Singh and Tomar [30], Koteska and Velinor [31] referred Washizaki metrics in estimation component reusability in their paper. This study used getter method of components for calculating this metric.

Definition: Component Observability is the number of getter method for all classes implemented in the component. Value of component observability is computed by the following expression of (2):

$$VCO(c) = \sum_{i=1}^{n} \frac{Gm(c)}{Am(c)}, \quad Am(c) > 0 \qquad (2)$$

$$0, \ Am(c) \leq 0$$

where, c is a component/package, Gm(c) is a number of getter method in c, Am(c) is a total number of method in c, and n is a number of classes in c

4.3. Value of Component Customizability (VCC)

Programming language, adapter and appropriate methods and interfaces are used to calculate metrics for adaptability [29]. Jatain and Gaur [32] and Washizaki et al. [17], used a number of write/wrote methods provided by the component to calculate customization metrics. This study also used a setter method of component to calculate this metric.

Definition: Component Customizability is the number of setter method for all classes implemented in the component. Value of Component Customizability for a component is calculated by (3):

$$VCC(c) = \sum_{i=1}^{n} \frac{Sm(c)}{Am(c)}, \quad Am(c) > 0 \qquad (3)$$

$$0, \ Am(c) \leq 0$$

where, c is presented to a component/package, Sm(c) is a number of setter method in c, Am(c) is a total number of method in c, and n is a number of classes in c.

4.4. Value of Component fan-in (VCFi)

The fan in of a class is the number of its immediately superordinate i.e. parent classes [33]. Fan in metrics measure the number of classes that depend on a given object [34]. Definition: VCFi is the value of fan-in for all classes implemented in component. Value of the component Fan-in (VCFi) of component c (VCO) is computed by the following calculation as (4):

$$VCF_i(c) = \sum_{j=1}^{n} \frac{F_{ij}(c)}{Af_{ij}(c)}, \quad Af(c) > 0 \qquad (4)$$

$$0, \ Af(c) \leq 0$$

where, c is a component/ package, F_i(c) is a number of fan-in in c, Af(c) is a total number of fan in c, and n is a number of classes in c.

4.5. Value of Component fan-out (VCFo)

The fan out of a class is the number of its immediately subordinate classes [33]. Fan out metrics measure the number of classes on which a given class depends [34]. Definition: VCFO is the value of fan-out for all classes implemented in component c. Value of the component fan-out (VCFO) of component is calculated by the following expression as (5):

$$VCF_o(c) = \sum_{j=1}^{n} \frac{F_{oj}(c)}{Af_{oj}(c)}, \quad Af(c) < 0 \qquad (5)$$

$$0, \ Af(c) \leq 0$$

where, c is a component/ package, Fo(c) is a number of fan-out in c, Af(c) is a total number of fan in c, and n is a number of classes in c. These metrics were used to measure overall component reusability (level of component reusability) using fuzzy logic technique.

5. RESULTS AND DISCUSSION

5.1. Metrics measurement

A survey is conducted to estimated measurement level for each of the predefined metric, i.e. VCDL, VCC, VCO, VCFi and VCFo. The estimation of the measurement level is important to set the range of values based on information about the actual users [35]. The value of the metric is from 0 to 1 [17]. This study proposed approach and implement the idea of using low, medium and high to calculate reusability level for the reuse component.

Table 1 shows the measurement levels for five metrics, VCLD, VCO, VCC, VCFi, and VCFo. The measurement level can be divided into three levels which is low, medium and high.

- For VCLD, the measurement level for low, medium and high is 0 to 0.37, 0.35 to 0.71 and 0.69 to 1.0 respectively.
- As for VCO, the measurement level for low is 0 to 0.37, for medium is 0.34 to 0.70 and for high is 0.68 to 1.0.
- While, for VCC, the measurement level for low, medium and high is 0 to 0.35, 0.36 to 0.71 and 0.68 to 1.0 respectively.
- For VCF_i, the measurement level for low is 0.67 to 1.0, followed by medium 0.34 to 0.68 and high 0 to 0.35 respectively.
- Lastly for VCFo, the measurement level for low is 0.69 to 1.0, for medium is 0.36 to 0.71 and for high is 0 to 0.69 correspondingly.

Table 1. Metrics Measurement Level

Metrics	Low	Med	High
VCDL	0 <= L<= 0.37	0.35 <= M<= 0.71	0.69 <= H<= 1.0
VCO	0 <= L<= 0.37	0.34 <= M<= 0.70	0.68 <= H<= 1.0
VCC	0 <= L<= 0.35	0.36 <= M<= 0.71	0.68 <= H<= 1.0
VCFi	0.67 <= L<= 1.0	0.34 <= M<= 0.68	0 <= H<= 0.35
VCFo	0.69 <= L<= 1.0	0.36 <= M<= 0.71	0 <= H<= 0.37

The result presented in Table 1, shown that the measurement levels for low, medium, and high contain an uncertainty value. Thus, in this situation, fuzzy logic is very suitable to be applied in order to determine the reusability level for the component in Table 1. Reason of choosing fuzzy logic to calculate the uncertainty will be explained in the future work.

6. CONCLUSION

This study describes the proposed approach, CREA. It also describes the calculation of the proposed reusability metrics and the reusability level of components. Even though the characteristics used for this study are the same with others evaluation model [17, 36], but for understandability; different sub characteristic for documentation level of the component was used. The metrics used for this evaluation are also different from the other models. These enhanced metrics were suited to the component features (setter method, getter method, fan in, fan out, and documentation) that were evaluated in this study. The metrics results of the components are used as input to the FIS, while the output is the reusability level for the component, automatically calculated by CREA tool which is a future work for this study.

ACKNOWLEDGEMENTS

This study is fully supported by Fundamental Research Grant Scheme (FRGS), vot number 59504. The authors fully acknowledged Ministry of Higher Education (MOHE) and Universiti Malaysia Terengganu for the approved fund which makes this important research viable and effective.

REFERENCES

[1] A. Alvaro, et al., "Quality Attributes for a Component Quality Model," *10th International Workshop on Component-Oriented Programming (WCOP) in Conjunction with ECOOP*, 2005.
[2] J. W. Hooper and R. O. Chester, "Software Reuse: Guidelines and Methods," R. A. D., Ed. New York, Plenum Press, 1991.
[3] M. Dabhade, et al., "A Systematic Review of Software Reuse using Domain Engineering Paradigms," *2016 Online International Conference on Green Engineering and Technologies (IC-GET)*, pp. 1-6, 2016.
[4] T. Vale, et al., "Twenty-Eight Years of Component-Based Software Engineering," *Journal of Systems and Software*, vol. 111, pp. 128-148, 2016.
[5] H. M. Manoj and A. N. Nandakumar, "A Novel Optimization towards Higher Reliability in Predictive Modelling towards Code Reusability," *International Journal of Electrical and Computer Engineering*, vol. 7, pp. 2088-8708, 2017.
[6] A. Verma and P. Kaur, "Design an Algorithm for Software Development in Cbse Environment using Feed Forward Neural Network," *IAES International Journal of Artificial Intelligence*, vol. 4, pp. 53-61, 2015.
[7] J. S. Poulin, et al., "The Business Case for Software Reuse," *IBM Systems Journal*, vol. 32, pp. 567-594, 1993.
[8] R. P. Diaz, "Implementing Faceted Classification for Software Reuse," *Communications of the ACM*, vol. 34, 1991.

[9] W. Schäfer, et al., "Software Reusability," in R. P. Diaz, et al., Ed. Ellis Horwood, Hemel Hempstead, 1994.
[10] W. Tracz, "Software Reuse: Motivators and Inhibitors," in W. Tracz, Ed., "Tutorial: Software Reuse-Emerging Technology," Washington, IEEE Computer Society, pp. 62-67, 1988.
[11] C. W. Kruger, "Software Reuse," *ACM Computing Surveys*, vol. 24, pp. 132-138, 1992.
[12] C. H. Lung, "An Analogy-Based Domain Analysis Mehodology," *Ph.D Thesis*, Arizona State University, pp. 252, 1994.
[13] A. Aloysius and K. Maheswaran, "A Review on Component Based Software Metrics," *International Journal Fuzzy Mathematic Arch*., vol. 7, pp. 185-194, 2015.
[14] B. Deniz and S. Bilgen, "An Empirical Study of Software Reuse and Quality in an Industrial Setting," *International Conference on Computational Science and Its Applications*, pp. 508-523, 2014.
[15] M. Harman, et al., "Search-Based Approaches to the Component Selection and Prioritization Problem," *GECCO '06: ACM Press*, 2006.
[16] K. J. Shambhu and R. K. Mishra, "Accessing Software Quality for Component-Based Software Through Trustworthiness and Dependenbability Analysis," *Internal Journal of Development Research*, vol. 5, pp. 4259-4261, 2015.
[17] F. B. Abreu, "Comparing Software Quality Models," *INES Technical Report*, 2001.
[18] H. Washizaki, et al., "A Metrics Suite for Measuring Reusability of Software Components," *The Ninth International Software Metrics Symposium (METRICS'03)*, 2003.
[19] J. M. Morel, 2003. Available: http://www.idi.ntnu.no/grupper/su/ courses/dif8901/presentations2003/r01.pdf.
[20] R. Land, et al., "Towards Efficient Software Component Evaluation," *Component Selection and Certification Swedish Foundation for Strategic Research*, vol. 56, 2009.
[21] A. Parvatiyar, et al., "Smart Advisor and Search Optimizer: Web-Based Applications of Fuzzy Rules, Intelligence Systems and Hierarchical Clustering for Relational Decisions," *The Sixth Research Conference on Relational Marketing and CRM*, pp. 6-12, 2002.
[22] A. Jedlitschka, et al., "Reporting Experiments in Software Engineering," *Guide to Advanced Empirical Software Engineering*, F. Shull, et al., Ed. London: Springer, pp. 201-228, 2008.
[23] M. Goulao, "Software Components Evaluation: An Overview," *ACM Computing Surveys*, 2004.
[24] S. Ismail, et al., "Determining Characteristics of the Software Components Reusability for Component Based Software Development," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 9, pp. 213-216, 2017.
[25] A. S. Andreou and M. Tziakouris, "A Quality Framework For Developing And Evaluating Original Software Components," *Information and Software Technology Journal*, vol. 49, pp. 122-141, 2006.
[26] Y. Singh, et al., "Software Reusability Assessment Using Soft Computing Techniques," *ACM SIGSOFT Software Engineering Notes*, vol. 36, pp. 1-7, 2011.
[27] P. S. Sandhu and H. Singh, "A Reusability Evaluation Model for OO-Based Software Components," *International Journal of Computer Science*, vol. 1, pp. 259-264, 2006.
[28] N. W. Nerurkar, et al., "Assessment Of Reusability In Aspect-Oriented Systems Using Fuzzy Logic," *ACM SIGSOFT Software Engineering Notes*, vol. 35, pp. 1-5, 2010.
[29] D. Hristov, et al., "Structuring Software Reusability Metrics for Component-Based Software Development," *The Seventh International Conference on Software Engineering Advances (ICSEA)*, pp. 421-429, 2012.
[30] A. P. Singh and P. Tomar, "Estimation of Component Reusability Through Reusability Metrics," *International Journal of Computer, Electrical, Electrical, Automation, Control and Information Engineering*, vol. 8, pp. 1729-1736, 2014.
[31] B. Koteska and G. Velinov, "Component-Based Development: A Unified Model of Reusability Metrics," *ICT Innovations 2012*, pp. 335-344, 2013.
[32] A. Jatain and D. Gaur, "Estimation of Component Reusability by Identifying Quality Attributes of Component: A Fuzzy Approach," *The Second International Conference on Computational Science, Engineering and Information Technology, ACM*, pp. 738-742, 2002.
[33] C. Borysowich. Available: http://it.toolbox .com/blog/enterprise –solution/design-principle-fanin-vsfanout-16088.
[34] M. Schroeder, "A Practical Duide to Object Oriented Metrics," *IT Pro*, 1999.
[35] T. Wahyuningrum and K. Mustofa, "A Systematic Mapping Review of Software Quality Measurement: Research Trends, Model, and Method," *International Journal of Electrical & Computer Engineering*, vol. 7, pp. 2088-8708, 2017.
[36] C. Singh, et al., "An Estimation of Software Reusability Using Fuzzy Logic Technique," *International Conference on Signal Propagation and Computer Technology (ICSPCT2014)*, pp. 250-256, 2014.

BIOGRAPHIES OF AUTHORS

Dr. Suryani Ismail is a senior lecturer at the School of Informatics and Applied Mathematics, UMT. Her research interests include software engineering (software reuse), computer science and information system. She has worked as a researcher in several national funded R&D projects.

Dr. Fatihah Mohd is a Postdoctoral at the School of Informatics and Applied Mathematics of the Universiti Malaysia Terengganu (UMT) in Malaysia. She received a PhD degree in computer science from UMT in 2016. Her research expertise focuses on data mining, artificial intelligence software engineering, and decision support system (clinical informatics). She has published many research papers in numerous refereed journal. She is a member of the MySIG and PECAMP. She has worked as a researcher in several national funded R&D projects.

Dr. Masita Masila Abdul Jalil is a senior lecturer at the School of Informatics and Applied Mathematics, UMT. Her research interests include software reuse, computer science education and information system. She is a member of MySIG, IEEE and PECAMP. She has worked as a researcher in several national funded R&D projects.

Ts. Dr. Wan M.N. Wan Kadir is an Associate Professor in Software Engineering at School of Computing, Faculty of Engineering, Universiti Teknologi Malaysia (UTM). He received his BSc. from Universiti Teknologi Malaysia, MSc. from UMIST, and Ph.D. from The University of Manchester. He has been an academic staff at School of Computing for more than twenty years (since 1997). He is an active researcher with a number of past and current projects. His research interest covers various SE knowledge areas based on the motivation to reduce the cost of development and maintenance as well as to improve the quality of large and complex software systems. His work has been supported by numerous research grants totaling in excess of RM 1 million.