UNIVERSITI PUTRA MALAYSIA

A QUALITY MODEL FOR COMPONENT-BASED SOFTWARE

MOHAMED ABDULLAHI ALI

FSKTM 2019 13

A QUALITY MODEL FOR COMPONENT-BASED SOFTWARE

By

MOHAMED ABDULLAHI ALI

Thesis submitted to the School of Graduate Studies
Universiti Putra Malaysia

in fulfilment of the requirements for the

Master of Software Engineering

JANUARY 2019

COPYRIGHT

DEDICATION

This thesis is dedicated to my beloved parents.

Abstract of thesis presented to the Universiti Putra Malaysia in fulfilment of the requirement for the Master of Software Engineering

A QUALITY MODEL FOR COMPONENT-BASED SOFTWARE

By

MOHAHMED ABDULLAHI ALI

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

In Component Based Software Development (CBSD), applications are built from existing components by assembling and replacing interoperable parts. Now day's component based software engineering considers one of the growing approaches for software development, its Reusable components that minimize implementation time, cost. To evaluate design quality of the component is important because it has main impact to the final implementation therefore, the existing component quality models all of them are based on generic attributes of the component so that none of them were discussed attributes that specific at design level for the component that has the main influence to the final product hence designing high quality component needs to get component quality model that specific in design level that based on design attributes for component. This thesis proposed Quality Model for Component-based Software at design level. To evaluate this quality model, it implemented a prototype metrics tool. Finally, this prototype metrics tool will help the developers to detect the design problems and indication the goodness of design early, hence good design leads to ease for maintenance and improve the quality of the final product.

Abstrak tesis yang dikemukakan kepada Universiti Putra Malaysia sebagai memenuhi keperluan untuk ijazah Sarjana Kejuruteraan Perisian

Oleh

MOHAHMED ABDULLAHI ALI

FAKULTI SAINS KOMPUTER DAN TEKNOLOGI MAKLUMAT

Dalam Pembangunan Sistem Berasaskan Komponen (CBSD), aplikasi dibangunkan daripada komponen sedia ada dengan memasang dan menggantikan bahagian yang boleh beroperasi secara rentas. Pada masa sekarang kejuruteraan perisian berasaskan komponen merupakan satu daripada kaedah pembangunan perisian yang semakin berkembang dengan komponen yang boleh diguna pakai semula mengurangkan masa pelaksanaan dan kos. Penilaian kualiti rekabentuk komponen adalah penting kerana ianya memberi kesan utama kepada pelaksanaan akhir, maka model kualiti komponen sedia ada adalah berdasarkan ciri-ciri umum komponen tersebut supaya tiada diantaranya adalah ciri-ciri diperbincangkan yang khusus pada tahap rekabentuk untuk komponen tersebut yang mempunyai kesan utama terhadap produk akhir yang seterusnya merekabentuk komponen berkualiti tinggi memerlukan kepada model kualiti komponen yang khusus untuk tahap rekabentuk berasaskan ciri-ciri rekabentuk bagi komponen. Tesis ini mencadangkan Model Kualiti untuk Perisian Berasaskan Komponen pada tahap rekabentuk. Bagi menilai model kualiti tersebut, satu prototaip peralatan metrik telah dilaksanakan. Akhir sekali, prototaip peralatan metrik tersebut akan membantu pembangun perisian untuk mengesan masalah dalam rekabentuk dan sebagai petunjuk kepada kelebihan rekabentuk awal, seterusnya rekabentuk yang baik menjurus kepada penyelenggaraan yang mudah dan meningkatkan kualiti produk akhir.

ACKNOWLEDGEMENT

In the name of ALLAH, the Beneficent, the Compassionate, thanks and praise to God for giving me strength and patience to complete my duties successfully.

I would like to express my very great appreciation to my supervisor Dr. Ng Keng Yap for his valuable and constructive suggestions during in this research. His willingness to give his time so generously has been much appreciated.

I would like to thanks my family for their support and encouragement. This thesis would not have been done without the foundation created by my beloved mother, Zeynab Abdullahi, and my father, Abdullahi Ali, and their spurring me on further, even when different continent separated us for many years. Thank you!

Finally, to all UPM staff, thanks you for your facilitation. I would like to acknowledge to any individual who are not mentioned here for his/her irreplaceable helps and cooperation.

APPROVAL

Thesis submitted to the Senate of University Putra Malaysia and has been accepted as fulfilment of the requirement for Master of Computer Science (Software Engineering).

Supervisor,

Dr. Ng Keng Yap

Department of Software Engineering

Faculty of Computer Science and Information Technology

Universiti Putra Malaysia

January/21/2019

DECLARATION

Declaration by Graduate Student

I hereby confirm that:

- this thesis is my original work;
- quotations, illustrations and citations have been duly referenced;
- this thesis has not been submitted previously or concurrently for any other degree at any other institutions;
- intellectual property from the thesis and copyright of thesis are fully-owned by Universiti Putra Malaysia, as according to the Universiti Putra Malaysia (Research) Rules 2012;
- written permission must be obtained from supervisor and the office of Deputy Vice-Chancellor (Research and Innovation) before thesis is published (in the form of written, printed or in electronic form) including books, journals, modules, proceedings, popular writings, seminar papers, manuscripts, posters, reports, lecture notes, learning modules or any other materials as stated in the Universiti Putra Malaysia (Research) Rules 2012;
- There is no plagiarism or data falsification/fabrication in the thesis, and scholarly integrity is upheld as according to the Universiti Putra Malaysia (Graduate Studies) Rules 2003 (Revision 2012-2013) and the Universiti Putra Malaysia (Research) Rules 2012. The thesis has undergone plagiarism detection software.

Signature: ________________________ Date: ___________________

Name and Matric No.: ____

TABLE OF CONTENTS

ACKNOWLEDGEMENT v
APPROVAL vi
DECLARATION vii
LIST OF TABLES xi
LIST OF ABBREVIATIONS xii
CHAPTER 1

1 INTRODUCTION 1
1.1 Research Background 1
1.2 Problem Statement 5
1.3 Research Objectives 8
1.4 Research Scope 8
1.5 Thesis Organization 9
1.6 Chapter Summary 10

2 LITERATURE REVIEW 11
2.1 Introduction 11
2.2 Software Components 11
2.3 History for Software Components 12
2.4 Existing Software Quality Models 13
2.5 Existing Software Component Quality Models 17
2.6 What Is Good Component Design? 18
2.7 Good Component Design according different component models 18
2.7.1 Architecture Description Language (ADL) Component Models 19
2.7.1.1 Compositionality 19
2.7.1.2 Reusability 20
2.7.1.3 Coupling 20
2.7.1.4 Configurability 21
2.7.1.5 Encapsulation 21
2.7.1.6 Interoperability 22
2.7.1.7 Complexity 23
2.7.1.8 Testability 23
2.7.1.9 Cohesion 24
2.7.1.10 Usability 25
2.7.1.11 Slim 25
2.7.1.12 Bloat 26
2.7.1.13 Interface Documentation 27
2.7.2 Object Component Models 28
2.7.2.1 Compositionality 28
2.7.2.2 Reusability 29
2.7.2.3 Coupling 29
2.7.2.4 Configurability 30
2.7.2.5 Encapsulation 31

2.7.2.6 Interoperability 31
2.7.2.7 Complexity 32
2.7.2.8 Testability 32
2.7.2.9 Cohesion 33
2.7.2.10 Usability 34
2.7.2.11 Slim 34
2.7.2.12 Bloat 35
2.7.2.13 Interface Documentation 36
2.7.3 Encapsulated Component Models 37
2.7.3.1 Compositionality 37
2.7.3.2 Reusability 38
2.7.3.3 Coupling 38
2.7.3.4 Configurability 39
2.7.3.5 Encapsulation 39
2.7.3.6 Interoperability 40
2.7.3.7 Complexity 40
2.7.3.8 Testability 41
2.7.3.9 Cohesion 42
2.7.3.10 Usability 42
2.7.3.11 Slim 43
2.7.3.12 Bloat 44
2.7.3.13 Interface Documentation 45
2.8 Comparison design quality attributes for different component models 46
2.9 Guidelines how it assigned rating for quality attributes in comparison table 2.1 48
2.10 Component Measurement 60
2.11 Summary 62

3 RESEARCH METHODOLOGY 63

3.1 Introduction 63
3.2 Research Methodology 63
3.1 Research Questions 63
3.2 Search Strategy 64
3.3 Study Selection 65
3.4 Quality Assessment 65
3.5 Data Synthesis 67
3.6 Result 68

4 PROPOSED COMPONENT QUALITY MODEL 70

4.1 Introduction 70
4.2 Proposed Component Quality Model 70
4.3 Attribute Descriptions for The Proposed Component Quality Model 71
4.3.1 Compositionality 71
4.3.2 Reusability 71
4.3.3 Coupling 72
4.3.4 Configurability 72
4.3.5 Encapsulation 72

4.3.6 Interoperability 73
4.3.7 Complexity 73
4.3.8 Testability 73
4.3.9 Cohesion 74
4.3.10 Usability 74
4.3.11 Slim 74
4.3.12 Bloat 75
4.3.13 Interface Documentation 75
4.4 Description for Metrics of the Proposed Component Quality Model 75
4.4.1 I%MCI 75
4.4.2 IMCM 77
4.4.3 CCBC 78
4.4.4 RCC 79
4.4.5 RCO 79
4.4.6 Fan (in/out) 80
4.4.7 CCM 80
4.4.8 COMC 82
4.4.9 AIC 83
4.4.10 PSU 84
4.4.11 RPD 84
5 PROTOTYPE DEVELOPMENT 86
5.1 Introduction 86
5.2 Overview Prototype Metric Tool 86
5.3 Description how prototype metric tool is working 87
5.4 Design and Implementation for Prototype Metric tool 87
5.5 Summary 88
6 DISCUSSION & CONCLUSION 89
6.1 Introduction 89
6.2 Research Discussion & Conclusion 89
6.3 Benefits of the research 90
6.4 Research Limitations and Future Work 90
6.5 Chapter Summary 91
REFERENCES 92

LIST OF TABLES

Table **Page**

Table 2.1: Comparison design quality attributes for different component models 45

Table 2.2: Guideline how it assigned rating for quality attributes in table 2.1 47

Table 3.1: Quality assessment criteria 64

Table 3.2: Quality assessment result 64

Table 3.3: What are main characteristic influencing good component design to derive quality model? 66

Table 3.4: What are metric for good component design? 66

LIST OF ABBREVIATIONS

ADL	ARCHITECTURE DESCRIPTION LANGUAGE
CBSE	COMPONENT BASED SOFTWARE ENGINEERING
CBS	COMPONENT BASE SOFTWARE
CBSD	COMPONENT BASED SOFTWARE DEVELOPMENT
COTS	COMPONENT-OFF-THE-SHELF
CQM	COMPONENT QUALITY MODEL
EUC	END-USER COMPUTING
SQuaRE	SYSTEM AND SOFTWARE QUALITY REQUIREMENTS OF EVALUATION
SCQM	SOFTWARE COMPONENT QUALITY MODEL
Ci	COMPONENT INTERACTION
I%MCI	% AGE METRICS FOR COMPONENT INTEGRATION
IMCM (BB)	INTERFACE METHOD COMPLEXITY METRIC FOR BLACK-BOX
CCBC	COUPLING COMPLEXITY OF BLACKBOX COMPONENT
IIc	INTERFACE INCOMING
OIc	OUTGOING INTERFACES
RCC	RATE OF COMPONENT CUSTOMIZABILITY
RCO	RATE OF COMPONENT OBSERVABILITY

CCM	COMPONENT COMPLEXITY METRIC
PCM	PARAMETER COMPLEXITY METRIC
CCCM	COMPONENT COUPLING COMPLEXITY METRIC
FICM	FAN-IN INTERFACSES
FOCM	FAN-OUT INTERFACES
COMC	COHESION OF METHODS WITH IN COMPONENT
AIC	AVERAGE INTERFACE COMLEXITY
PSU	PROVIDED SERVICE UTILIZATION
RPD	REFERENCE PARAMETER DENSITY

CHAPTER 1

INTRODUCTION

This chapter explores background software engineering, conventional software development challenges and solutions, CBSE, problem statement, research objectives, scope of the research and dissertation organization.

1.1 Research Background

Software engineering is a discipline that concerns the all aspects of software development including methodologies, project management and tools [22]. Traditional software development approaches advocate phase bye phase software process meaning that starting development from scratch so that it results several problems such as budget overrun and late delivery. Moreover, that approaches also lead to low quality and high maintenance software [23].

To tackle the problems of traditional software development, a new approach for component based software engineering (CBSE) was arisen [2]. CBSE is a software development approach that integrates components within appropriate software architecture rather than starting a software development from scratch to save cost and time of development [2]. CBSE solved the problems in traditional software development by composing existing components instead of starting development from scratch [2 and 23]. Finally, CBSE is an approach that change the way of software development and results developing with less time and less effort [11].

Software component has many definitions. According [24] a component is a "reusable unit deployment and compositions that can access through interface". Szyperski [24] defines a component precisely by looking the characteristic properties of a component: "A software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. A software component can be deployed independently and is subject to composition by third party". D.Souza and Wills [3] described that a component as a reusable part of software, that is independently developed and can be integrated with other components to build larger units. It may be adapted but may not be modified. A component can be, for instance, "compiled code" without a program source (so that it may not be modified) or part of a model or a design.

Although the reusability concept is familiar to us from object oriented technologies, CBSE takes an approach to reusability that is different from conventional software reuse. Aoyama [4] explains this difference as follows: First, components can be composed at run time without the need for compilation. Second, a component separates its interface from its implementation and conceals its implementation details, hence permitting composition without the requiring to know the component implementation details. The interface of a component should be standardized to enable reuse and allow components to interoperate in a predefined architecture. Lastly, all above definitions of software components shows that component is independently developed which enable to compose through standard interface in order to build larger system from pre-existing components.

Component based software (CBS) is an approach that should characterize if they have interface, contracts, framework, and pattern. Interface represents access points to a component. In component specification can be through contracts that enable to make sure a certain conditions holds true during the execution of component with its environment. The framework is a large unit of design which defines relationship between participants of the framework. Patterns define recurring solutions to recurring problems [24]. To summarize, CBS has unique characteristics that can easily distinguish from conventional software's that are: interface, contracts, framework, and pattern.

Many of the challenges were by faced CBSE approach that is the quality of the component which eventually give to the quality of final product [10]. According to IEEE, software quality is defined as "the degree to which system, system component or process meets specified requirements" or "the degree to which system, system component, or process meets customer or user needs or expectations".   CBSE can used for building many domain applications including embedded systems that mostly considers to critical for business success and also many other related domain human safety hence assessing and evaluating is become mandatory in CBSE lifecycle. A risk for choosing a component with unknown attributes is no longer acceptable and when it happens it may cause a huge damage result. However, software component quality become increasingly important activity to bring reliability in reusing of software components. The quality of components has main impact for final system [2]. In conclusion, to evaluate quality of the component is important because it has main impact to the business success, human safety hence evaluating the quality of the components may those reduce risks.

There are several reasons that motivates to focus goodness of the component design such producing quality product, ease maintenance and help to detect design issues early, and architectural differences of the component. Good software components design results in high quality for final product, so that market needs building component that has good quality in order to do that may need to look quality attributes and evaluation (metrics); enables for indication the goodness of design early [16]. Good design leads to ease for maintenance while poor quality derived from poor design because internal structures and methods are exposed that leads for complicated interdependencies hence bad design may responsible for time to market pressure [10]. In CBSE, quality aspect becomes more crucial because of architectural variances hence the quality of the component will be high influence for the quality of the final system [2]. To conclude, there are main reasons that motivate to evaluate the research of the component design including, to detect the design errors early, it helps for ease maintenance if maintenance require, architectural variances of the components and also results to improve the quality of the product because the errors detect at early.

To evaluate software quality, several software quality models were proposed but their limitations were having general attributes software quality so that it's hard to apply specific domains like CBSD therefore Component Quality Model (CQM) were proposed. Software quality models were proposed in order to solve the quality issues and to avoid producing software whose quality is below the standard that may lead [16 and 2]. There is a limitation for existing software quality models because they focus on general quality hence it's very hard to apply to specific domains such as Component-Off-The-Shelf (COTS) and (CBSD) [31]. By referring a set of models [31] and ISO/IEC 9126, CQM was proposed that based on ISO/IEC 9126, it contains marketing attributes and relevant component information [31]. To evaluate software quality, several software quality models were proposed but their limitations were having general attributes software quality so that it's hard to apply specific domains like CBSD therefore Component Quality Model (CQM) were proposed.

1.2 Problem Statement

The existing conventional software quality model is not applicable to software components because internal structure such as source code is not available at CBSE, therefore there is need quality model for CBSE based on black-box [75].

The existing component quality models is too general in terms of the attributes they have identified hence this shows the lack of component quality model that is specific to design level of the component in-order to detect the errors at early stage of design that minimizes the cost, effort, resources for implementation of the component and improves the quality. In [2], a component quality model has been proposed for CBSE that consists of six quality attributes namely functionality, maintainability, usability, efficiency, reliability, fault tolerance, portability hence this components quality model is too generic because it based on attributes for whole component life cycle and it does not focus on component design quality attributes that mainly influences the quality the final system. Several component quality models have been proposed such as [2, 31, 33 and 34] but none of them did not identify the attributes to evaluate the quality of internal design for the components that helps to take decision at early stage of design and detecting design problems early instead of final stage [4]. To conclude, all above discussed component quality models are too general in terms of their quality attributes and also there is no component quality model that is specific at design that can plays important role for the detection of errors at early stage of design meaning that before implementation because it reduces cost, effort and resources and improve the quality of final product.

Several component quality models were proposed but none of them did not identified for both component quality attributes and metrics and also limited research for component measurement so that component metrics is important because component is black-box cannot see the internal structure. Existing component quality model such as: Software Component Quality Model (SCQM) [33], Software component quality characteristics model for CBSE [2], Quality characteristics model for COTS component [34], software component quality model [31] but none of them does not talk together component quality attributes and metrics, in [4] component evaluation it does not only needs to mention what to evaluate (attributes) [4] that is identifying quality attributes but also needs to come up with how to evaluate (metrics) component that is component metrics and [16] it also enable to take decision at early stage of design and detecting design problems early instead of final stage. As on paper [1] stated that measuring component quality attributes is still issue not solved at component design level and needs for further investigation because the component is a black-box that means other developers that needs to integrate the existing component to their work is restricted from internal design of the component [4] hence component design metrics can help developers to make decision at early stage of design and detect problems more quickly [16]. There is limited research for measuring software component quality compared to conventional software quality [2]. To summarize, the above mentioned component quality models all of them they identified the general quality attributes of the component but they don't identify the metrics use to measure those attributes hence this shows needs for proposing a quality model for component specific at design that based on for both attributes and their metrics hence it will lead to develop component design metric tool for component design quality evaluation.

Currently there is lack of tool for evaluating the component design quality hence this may lead difficultness for decision making and also detecting problems at design level so that design quality has main impact to the component quality. As many researchers agree that there is lack of tool for measuring design quality of software components [7, 8, and 9] because this tool will help developers to make decision at early stage of design and detect problems more quickly hence developers can fix their design problems and recheck again their design [16]. On this paper [16] announced that main reason needs to develop this tool is that "design" is the most influential factor for component quality. To sum up, the existing researches shows lack of component metric tool that enables the developers for both early decision making and detecting design errors at early because design plays key role for the component quality.

1.3 Research Objectives

To overcome these problems, the main objectives of this research are:

I. To investigate for main characteristics influencing good component design to derive a quality model.

II. To propose metrics for the quality model.

III. To implement a prototype metrics tool for selected metrics.

1.4 Research Scope

This research limited to propose quality model for component based software at design level in order to evaluate the goodness of the component at earlier stage and detect all design issues before implementation therefore the proposed component quality model will enhance the component quality and leads to implement high quality component at implementation.

1.5 Thesis Organization

CHAPTER 1 discusses about traditional software development and issues. Also this chapter discusses problem statements, research objectives, scope.

CHAPTER 2 discusses general overview about component and history. Also the chapter discusses existing software and component quality models. In addition to that, this unit also covered the.

CHAPTER 3 presents the overall methodology used to conduct this research in order to achieve the main goal of the research.

CHAPTER 4 explain the proposed component quality model in this study. Also discussed in detail the attributes and metrics in the proposed component quality model.

CHAPTER 5 provides the evaluation of the study or research. This evaluation conducted using prototype metric tool.

CHAPTER 6 gives explanation about the conclusion, future work and recommendations. It also provides the achievements of the study such: objectives, research questions and the limitations of the research are covered.

1.6 Chapter Summary

In this chapter, the author described the introduction that guides the entire conduct of this research. Beginning with a background of the study, the chapter continues with a discussion of the problems addressed by this research. Furthermore, specific objectives were discussed. Also, the scope delimiting this study was presented. Finally, readers guide on the organization of this thesis is presented as the closing section of the introductory chapter.

REFERENCES

[1] Anguswamy, R., & Frakes, W. B. (2013). Reuse design principles. In *International Workshop on Designing Reusable Components and Measuring Reusability (DReMeR 2013)*.

[2] Tiwari, A., & Chakraborty, P. S. (2015, February). Software component quality characteristics model for component based software engineering. In *Computational Intelligence & Communication Technology (CICT), 2015 IEEE International Conference on* (pp. 47-51). IEEE.

[3] Abdellatief, M., Sultan, A. B. M., Ghani, A. A. A., & Jabar, M. A. (2013). A mapping study to investigate component-based software system metrics. *Journal of systems and software*, *86*(3), 587-603.

[4] Kaur, K., & Singh, H. (2009). Evaluating an evolving software component: case of internal design. *ACM SIGSOFT Software Engineering Notes*, *34*(4), 1-4.

[5] Kalia, A., & Sood, S. (2017). A Metrics Based Framework to Improve Maintainability of Reusable Software Components through Versioning. *International Journal of Advanced Research in Computer Science*, *8*(3).

[6] Ismail, S., Kadir, W. M. W., Noor, N. M. M., & Mohd, F. (2017). Determining Characteristics of the Software Components Reusability for Component Based Software Development. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, *9*(3-5), 213-216.

[7] Heineman, G. T., & Councill, W. T. (2001). Component-based software engineering. *Putting the pieces together, addison-westley*, 5.

[8] Carvalho, F., Meira, S. R., Freitas, B., & Eulino, J. (2009, August). Embedded software component quality and certification. In *Software Engineering and Advanced Applications, 2009. SEAA'09. 35th Euromicro Conference on*(pp. 420-427). IEEE.

[9] Alvaro, A., de Almeida, E. S., & de Lemos Meira, S. R. (2005, August). Software component certification: a survey. In *Software Engineering and Advanced Applications, 2005. 31st EUROMICRO Conference on* (pp. 106-113). IEEE.

[10] Kaur, K., & Singh, H. (2008, July). A Metrics Based Approach to Evaluate Design of Software Components. In *18th ECOOP Doctoral Symposium and PhD Student Workshop* (p. 17).

[11] Negi, G. P. (2015). Evaluating Quality of Software Component using Metrics.

[12] MacCormack, A., Rusnak, J., & Baldwin, C. Y. (2007). The impact of component modularity on design evolution: Evidence from the software industry.

[13]Jack.(1998).Configurability.https://www.thwink.org/soft/article/future_app_dev/Configurability.html. accessed 4/8/2018.

[14] Lüer, C., & Van Der Hoek, A. (2002). *Composition environments for deployable software components*. Department of Information and Computer Science, University of California, Irvine.

[15] Lau, K. K., & Wang, Z. (2005). A survey of software component models. In *in Software Engineering and Advanced Applications. 2005. 31 st EUROMICRO Conference: IEEE Computer Society*.

[16] Irwanto, D. (2010, December). Visual Indicator Component Software to Show Component Design Quality and Characteristic. In *Advances in Computing, Control and Telecommunication Technologies (ACT), 2010 Second International Conference on* (pp. 50-54). IEEE.

[17] Stephe W. (2007). Union Design pattern. OpenStax-CNX.

[18] Kitchenham, B., Pretorius, R., Budgen, D., Brereton, O. P., Turner, M., Niazi, M., & Linkman, S. (2010). Systematic literature reviews in software engineering–a tertiary study. *Information and Software Technology*, *52*(8), 792-805.

[19] Sanatnama, H., Ghani, A. A. A., Yap, N. K., & Selamat, M. H. (2008). Mediator Connector for Composition of Loosely Coupled Software Components. *Journal of Applied Sciences*, *8*(18), 3139-3147.

[20] Di Cola, S. Catch Me If You Can: To Use a Component You Need to Find It First.

[21] Lau, K. K., Elizondo, P. V., & Wang, Z. (2005, May). Exogenous connectors for software components. In *International Symposium on Component-Based Software Engineering* (pp. 90-106). Springer, Berlin, Heidelberg.

[22] Sommerville, I. (2010). *Software engineering*. New York: Addison-Wesley.

[23] Simão, R. P., & Belchior, A. D. (2003). Quality characteristics for software components: Hierarchy and quality guides. In *Component-based software quality* (pp. 184-206). Springer, Berlin, Heidelberg.

[24] Crnkovic, I., & Larsson, M. P. H. (2002). *Building reliable component-based software systems*. Artech House.

[25] Ghani, N., Hedges, J., Winschel, V., & Zahn, P. (2016). Compositional game theory. *arXiv preprint arXiv:1603.04641*.

[26] Gill, Nasib S. "Reusability issues in component-based development." *ACM SIGSOFT Software Engineering Notes*28.4 (2003): 4-4.

[27] Sametinger, J. (1997). *Software engineering with reusable components*. Springer Science & Business Media.

[28] Eric M. Dashofy (2002). *Interoperability* [PowerPoint Slides]. Retrieved from https://www.ics.uci.edu/~taylor/ICS221/slides/Interoperability.ppt.

[29] Madiajagan, M., & Vijayakumar, B. (2006). Interoperability in component based software development. *World Academy of Science, Engineering and Technology*, *22*, 68-75.

[30] Gui, G., & Scott, P. D. (2009). Measuring Software Component Reusability by Coupling and Cohesion Metrics. *JCP*, *4*(9), 797-805.

[31] Alvaro, A., De Almeida, E. S., & Meira, S. L. (2006, August). A software component quality model: A preliminary evaluation. In *Software Engineering and Advanced Applications, 2006. SEAA'06. 32nd EUROMICRO Conference on* (pp. 28-37). IEEE.

[32] K.k lua and Simone dicola (2017). An introduction to component based software development. world scientific.

[33] Upadhyay, N., Despande, B. M., & Agrawal, V. P. (2011, January). Towards a software component quality model. In *International Conference on Computer Science and Information Technology* (pp. 398-412). Springer, Berlin, Heidelberg.

[34] Bertoa, M. F., & Vallecillo, A. (2002). Quality attributes for COTS components.

[35] Gill, N. S., de Cesare, S., & Lycett, M. (2002). Measurement of Component-Based Software: Some Important Issues.

[36] Lau, K. K., Elizondo, P. V., & Wang, Z. (2005, May). Exogenous connectors for software components. In *International Symposium on Component-Based Software Engineering* (pp. 90-106). Springer, Berlin, Heidelberg.

[37] Lau, K. K., & Wang, Z. (2007). Software component models. *IEEE Transactions on software engineering*, *33*(10).

[38] Lau, K. K., Ornaghi, M., & Wang, Z. (2005, November). A software component model and its preliminary formalisation. In *International Symposium on Formal Methods for Components and Objects* (pp. 1-21). Springer, Berlin, Heidelberg.

[39] Aoyama, M. (1998, April). New age of software development: How component-based software engineering changes the way of software development. In *1998 International Workshop on CBSE*.

[40] Szyperski, C. (1998). Component Software: beyond object-oriented software. *Reading, MA: ACM/Addison-Wesley*.

[41] Youness, B., Abdelaziz, M., Habib, B., & Hicham, M. (2013). Comparative Study of Software Quality Models. *IJCSI International Journal of Computer Science Issues*, *10*(6), 1694-0814.

[42] Al-Qutaish, R. E. (2010). Quality models in software engineering literature: an analytical and comparative study. *Journal of American Science*, *6*(3), 166-175.

[43] Bertoa, M. F., & Vallecillo, A. (2002). Quality attributes for COTS components.

[44] Alvaro, A., Almeida, E. S., & Meira, S. L. (2005). Quality attributes for a component quality model. *10th WCOP/19th ECCOP, Glasgow, Scotland*, 31-37.

[45] Rawashdeh, A., & Matalkah, B. (2006). A new software quality model for evaluating COTS components. *Journal of Computer Science*, *2*(4), 373-381.

[46] Kharb, L., & Singh, R. (2008). Complexity metrics for component-oriented software systems. *ACM SIGSOFT Software Engineering Notes*, *33*(2), 4.

[47] Lau, K. K., Ling, L., & Wang, Z. (2006, August). Composing components in design phase using exogenous connectors. In *Software Engineering and Advanced Applications, 2006. SEAA'06. 32nd EUROMICRO Conference on* (pp. 12-19). IEEE.

[48] Scheller, T., & Kuhn, E. (2011, August). Measurable concepts for the usability of software components. In *Software Engineering and Advanced Applications (SEAA), 2011 37th EUROMICRO Conference on* (pp. 129-133). IEEE.

[49] Dong, J. (2002). Design Component Contracts: Modeling and Analysis of Pattern-Based Composition [Ph. D. Thesis]. *Waterloo, Ontario, Canada, University of Waterloo*.

[50] Liu, Y., & Cunningham, H. C. (2002, April). Software component specification using design by contract. In *Proceeding of the SouthEast Software Engineering Conference, Tennessee Valley Chapter, National Defense Industry Association* (Vol. 6, p. 2). sn.

[51] Parnas, D. L. (2006). Component Interface Documentation: What do we Need and Why do we Need it?. *FRONTIERS IN ARTIFICIAL INTELLIGENCE AND APPLICATIONS*, *147*, 3.

[52] McGrenere, J., & Moore, G. (2000, May). Are we all in the same" bloat"?. In *Graphics interface* (Vol. 2000, pp. 187-196).

[53] Efi Papatheocharous. (2012). Component-based software engineering. Retrieved from https://www.cs.ucy.ac.cy/~cs00pe/epl603/lectures/Lect11-12.pdf

[54] Morasca, S. (2001). Software measurement. In *Handbook of Software Engineering and Knowledge Engineering: Volume I: Fundamentals* (pp. 239-276).

[55] Rana, P., & Singh, R. (2014). A Study of Component Based Complexity Metrics. *International Journal of Emerging Research in Management & Technology*, *3*(11), 159-16.

[56] Fenton, N., & Bieman, J. (2014). *Software metrics: a rigorous and practical approach*. CRC press.

[57] Kaur, N., & Singh, A. (2013). A complexity metric for black box components. *International Journal of soft computing and engineering*, *3*(2).

[58] Kaur, N., & Singh, A. (2013). A Metric for Accessing Black Box Component Reusability. *International Journal of Scientific & Engineering Research*, *4*(7), 1114-1121.

[59] Rotaru, O. P., & Dobre, M. (2005, January). Reusability metrics for software components. In *aiccsa* (pp. 24-I). IEEE.

[60] Kumar, S., Tomar, P., Nagar, R., & Yadav, S. (2014). Coupling metric to measure the complexity of component based software through interfaces. *International Journal*, *4*(4).

[61] Agarwal, J., Dubey, S. K., & Tiwari, R. (2017). A Roadmap to Identify Complexity Metrics for Measuring Usability of Component-Based Software System. In *Advances in Computer and Computational Sciences* (pp. 33-41). Springer, Singapore.

[62] Kumari, U., & Upadhyaya, S. (2011). An interface complexity measure for component-based software systems. *International Journal of Computer Applications*, *36*(1), 46-52.

[63] Li, Shimin, and Ladan Tahvildari. "A service-oriented componentization framework for java software systems." In *Reverse Engineering, 2006. WCRE'06. 13th Working Conference on*, pp. 115-124. IEEE, 2006.

[64] Tonella, P., Antoniol, G., Fiutem, R., & Merlo, E. (1997, March). Points to analysis for program understanding. In *Program Comprehension, 1997. IWPC'97. Proceedings., Fifth Iternational Workshop on* (pp. 90-99). IEEE.

[65] Boxall, M. A., & Araban, S. (2004). Interface metrics for reusability analysis of components. In *Software Engineering Conference, 2004. Proceedings. 2004 Australian* (pp. 40-51). IEEE.

[66] Washizaki, H., Yamamoto, H., & Fukazawa, Y. (2003, September). A metrics suite for measuring reusability of software components. In *Software Metrics Symposium, 2003. Proceedings. Ninth International* (pp. 211-223). IEEE.

[67] Yadav, K., & Tomar, P. (2014). Design of Metrics for Component-Based Software System at Design Level. *International Journal of Engineering and Technical Research*, *2*(4), 285-289.

[68] Latika, M. (2011). Software component complexity measurement through proposed integration metrics. *Journal of Global Research in Computer Science*, *2*(6), 13-15.

[69] KOSGEY, J. K. K. (2017). *An Evaluation Model for Determining Quality in Academic Websites* (Doctoral dissertation, COHES-JKUAT).

[70] Reussner, R., Poernomo, I., & Schmidt, H. (2003). Contracts and quality attributes for software components.

[71] Reussner, R. H., Firus, V., & Becker, S. (2004, June). Parametric performance contracts for software components and their compositionality. In *Proceedings of the 9th International Workshop on Component-Oriented Programming (WCOP 04)* (pp. 40-49). June.

[72] e Abreu, F. B. (2005, August). Composition assessment metrics for CBSE. In *null* (pp. 96-105). IEEE.

[73] Aloysius, A., & Maheswaran, K. (2015). A review on component based software metrics. *Int. J. Fuzzy Math. Arch*, *7*(2), 185-194.

[74] Rana, P., & Singh, R. (2016). A Design of Cohesion and Coupling Metrics for Component based Software Systems. *International Journal of Computer Applications*, *146*(4).

[75] Goulão, M. (2011). An overview of metrics-based approaches to support software components reusability assessment. *arXiv preprint arXiv:1109.6*