Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.18576/amis/120116

Using Ordered Probit Model to Study the Effects of Component Quality on Reusability

Simrandeep Singh Thapar[1,*], *Paramjeet Singh*[2] *and Shaveta Rani*[2]

[1] Department of Computer Applications, ACET, Amritsar, Punjab, India
[2] Department of Computer Science & Engineering, GZSCIET Bathinda, Punjab, India

Received: 3 Oct. 2017, Revised: 20 Dec. 2017, Accepted: 23 Dec. 2017
Published online: 1 Jan. 2018

Abstract: Quality of a software component is inversely proportional to the number of inadequacies found in a component. Individual software components must be selected with utmost care as these are crucial to the success of CBSE approach. In this paper, reusability based component quality framework (RQF) is used to identify highly reusable software components. A component quality model as a part of RQF is also developed using statistical approach. It has six essential quality characteristics which are ranked by three stakeholders according to their preferences. Subsequently, effect of characteristics of component quality model on reusability is tested using ordered probit model which showed that reliability is the most influential characteristic which is instrumental in increasing the reuse of software components.

Keywords: Ordered Probit Model, Quality model, Reusability, Software Component, Stakeholder

1 Introduction

CBSE is a systematic, quantifiable and disciplined set of activities that aims at constructing and designing software by using a set of reusable software components [1]. It is reported by Ampatzoglou[2] that in FreeBSD (a well known operating system) about 43% of the classes have been reused from other projects and that open source software reuse rates are extremely high. More specifically, 53% of the projects have performed reuse activities in 30% of their development process and that 49% of projects have reused more than 80% of their code.

The CBSE asserts component reuse that provides software engineers with a number of measurable benefits such as lower defect density (almost 50% less), higher stability between releases which leads to the high quality component in reduced cost and shortened time hence improved productivity [3], [4], [5]. Based on the claims of studies [1], [6], with reuse actual efforts made in different software development phases are 17% to 20% lower for releases than the actual effort without reuse; component assembly leads to a 70 percent reduction in development cycle time; an 84 percent reduction in project cost, and a productivity index of 26.2, compared to an industry norm of 16.9. Intuitively, savings occur with components reuse as these are already tested and are not to be built from scratch. Further, overall product quality improves if quality components are reused [7].

Besides several benefits of reuse, it is however obligatory to deliberate about the questions [8], [9], [10] related to reuse beforehand such as:

–Is the component reusable in many implementations with only minor changes?
–How common is the component's function within the domain?
–Is reuse through modification feasible?
–Will the component fit into the system without damaging or harming the already existing system?
–Has the component been effectively tested by the component vendor before its appearance on the component market?
–Is the component reliable?
–If there are problems later with the component will it be easy to correct the problems and maintain the component with ease?
–Does the component satisfy the requirement of the component purchaser or end user?

These questions crop up since there is no certification standard or quality framework which may assure

* Corresponding author e-mail: simthain@gmail.com

stakeholders that a particular component is of high quality and highly reusable. And, if these questions are not addressed and low quality components are used, it may cause damage to the existing system leading to an avalanche. Therefore, success of CBSE is largely dependent on the quality of each component that comprises the system [9], [10], [11].

Quality is a significant factor during component selection since almost all software components provide the required functionality but a component which exceeds customer expectations is selected. Hence, a quality framework is needed for specifying quality requirements and to evaluate quality [12], [13]. It must also focus on reusability of software components to get more support from industry [3], [4], [11]. Many proposed quality frameworks [12], [13], [14],[15] are general in nature and focus on software quality rather than addressing quality issues of components. Therefore, a separate quality framework is needed because software components have different architecture, smaller in size and lesser in scope.

In this paper, a reusability based quality framework (RQF) is proposed that is useful for stakeholders to identify highly reusable and high quality components for software development. A high quality component can be identified with the involvement of concerned stakeholders in quality framework at various levels such as at initial stage to specify their quality requirements and at the end to validate the software component [11], [13]. Three major stakeholders are user, developer and manager. Their satisfaction level from the quality of component must be given suitable importance at the time of quality evaluation [16]. The quality framework emphasizes use of blackbox components since use of whitebox components increases the cost of project which would defy the goal of reuse [17]. The reusability based approach and flexibility of its application in various domains will provide adequate confidence to the organizations to practice it for the benefits like reduction in development cost, shorter development times and hence development of highly reliable and stable software [6], [18].

In section 2, literature of quality models from year 1970 to 2012 is briefly discussed and their shortcomings are brought out. In Section 3 development of reusability based quality framework is discussed. Section 4 discusses the design of statistical study conducted. Ordered probit model is briefly explained in Section 5. Section 6 elaborates the results obtained of the statistical study. We concluded this in Section 7 paper by outlining the important characteristics for reusability.

2 Literature of quality frameworks

The literature pertaining to quality frameworks is extensively studied and 26 quality models and framework from year 1970 to 2012 are identified [19]. All of these models are introduced with the purpose of defining quality in a new way. These models discuss quality with various dimensions including quality evaluation, quality improvement and stakeholder's view of quality.

As shown in Fig. 1, these models are categorized into basic quality models (1970-2001) and tailored quality models (2002-2012). In the year 1970, the research in quality improvement and software evaluation uplifted due to growing quality expectations of customers from software. During the period (1970-2001), less number of software organizations were existing, CBSE paradigm was not prevalent and nature of software products was less diversified. So, only five quality models are introduced in this period. Most of these models, especially ISO 9126 serves as a basic model for the models proposed after year 2001. During (2002-2012) software industry expanded rapidly with tough competition of providing high quality software products that stimulated research in this discipline. Researchers in this period proposed twenty one quality models which are enhanced forms of basic quality models and are according to the needs of distinct applications, domains and stakeholders.

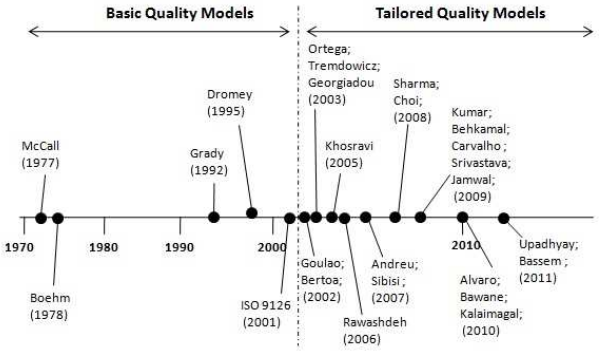

Fig. 1: Quality models proposed till 2012.

As we are concerned with only component quality models, we identified some major shortcomings of existing component models:

1. In [3], [10], [12], [14], the stakeholders and independent parties are not involved in the process of requirement elicitation and quality evaluation/validation of quality model.
2. It is not clarified in [4], [12], [13], [20], [21] that how a quality model can increase the reuse of software components. It is also obscure how to evaluate the software components and how to assess relation and impact of each characteristic, sub-characteristic and attribute.
3. The models [3], [13], [15], [20] are not confined to any domain or application which doubts their applicability. Also, Characteristics are not prioritized depending on the needs of particular domain or stakeholders.

4.No guidelines are suggested related to use or application of given models in different applications and domains [5], [11], [15].

These shortcomings of existing component quality models are resolved to great extent by reusability based quality framework.

3 Reusability based quality framework

The objective of reusability based quality framework (RQF) is to identify highly reusable and high quality software components. Fig. 2 presents the functioning of RQF. It consists of two stages, development of a component quality model and quality evaluation of a component. The implementation of RQF is mentioned in the following steps:

1.Elicit quality goals of stakeholders and map these goals into high level characteristics of software components.
2.These higher level characteristics are decomposed to the lowest level of hierarchy that covers all implied aspects of component quality. From this hierarchy, relationships may be drawn out among characteristics and sub-characteristics.
3.Assign software metrics to the lowest level of quality hierarchy, here, attributes of software components.
4.Requirements for selection or evaluation of appropriate component in particular application or domain are specified by stakeholders.
5.Candidate components are evaluated against the specified requirements with the help of metrics that are associated to component quality model.
6.Results are analyzed in order to validate the component for final selection. If results are satisfactory then component is recommended for integration with the system or may be stored for future use in internal component repository. In case of unsatisfactory results, another component is selected for evaluation.

Involvement of stakeholders in RQF is intuitive since quality can be perceived only with the eyes of users. Three main categories of stakeholders involved in software engineering activities are user, manager and developer. Each of these has different quality goals such as users want ease in using troublefree software, managers are interested in economical use of resources and developers are interested in less rework. So, stakeholders must be satisfied from quality of a component, although, to some extent.

Fig. 3 shows relationship diagram of a quality attribute with other facets of RQF. According to [22], a characteristic is a set of properties by which quality can be described and evaluated. It can be refined into multiple levels of sub-characteristics and a sub-characteristic can be refined into attributes. An attribute is a quality property

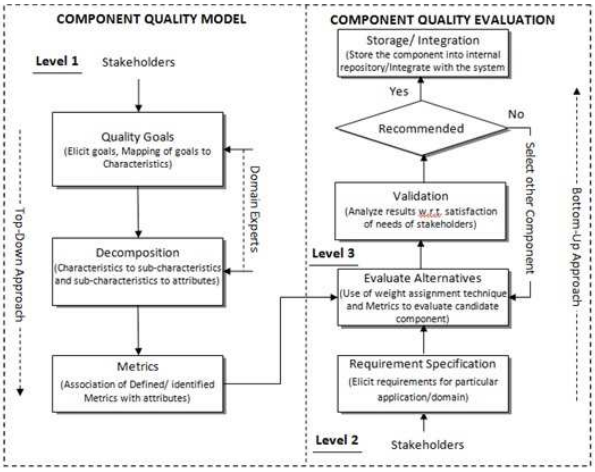

Fig. 2: Reusability based quality framework.

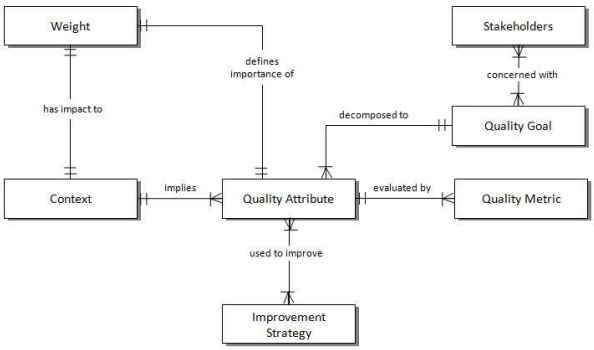

Fig. 3: Attribute relationship diagram of quality model.

to which a metric can be assigned, where a metric is a procedure for examining a component to produce a single number.

3.1 Development of Component Quality Model

In the present paper, the first stage of RQF is implemented, that is development of a quality model for software components. A quality model is the set of characteristics and sub-characteristics, as well as the relationships among them and it largely depends on the kind of target product to be evaluated [23].

Table 1, shows all the important quality characteristics which are preferred by existing component quality models. It is clear from the table Functionality, Reliability, Usability, Efficiency, Maintainability and Portability are most preferred characteristics for software components. Functionality as a quality characteristic of a software component is explicitly stated by stakeholders but other characteristics Reliability, Usability, Efficiency,

Table 1: Important characteristics of software components

Characteristics / Quality models	Functionality	Reliability	Usability	Efficiency	Maintainability	Portability	Reusability	Traceability	Security	Interoperability	Testability	Usability-in-use	Safety-in-use	Manageability	Modularityy
Andreou [3]	1	1	1	1	1										
Upadhyay[5]	1	1	1	1	1	1	1	1							
Kalaimagal[9]	1	1	1	1	1	1	1		1	1	1	1	1		
Sharma[10]	1	1	1	1	1	1									
Alvaro[11]	1	1	1	1	1	1									
Bertoa[23]	1	1	1	1	1	1									
Rwashdeh[24]	1	1	1	1	1									1	
Goulao[25]	1	1	1	1	1	1									
Choi [26]	1	1	1	1	1	1	1								1

Maintainability and Portability are desired factors and are decisive at time of component selection. Presence of all these six characteristics increases the prospects of component reuse as these are indispensable for all kind of applications or domains.

Table 2 shows 24 sub-characteristics which are preferred by these models. These sub-characteristics can be appropriately associated to 6 characteristics selected in Table 1 by using factor analysis technique. For this, KMO and Bartlett's test was conducted on collected data, the value of KMO from the results was (0.803) which should be above 0.5 and also the value of Bartlett's test was significant (0.000). Thus, from the perspective of KMO and Bartlett's test, factor analysis test is suitable for the given sample. The extraction communalities estimated that amount of variance in 16 sub-characteristics is above (0.5), these are selected and 8 sub-characteristics with variance below (0.5) are discarded.

Table 3 shows highest factor loadings of sub-characteristics under six components, other factor loadings with low values are omitted intentionally. The sub-characteristics accuracy (.725), suitability (.744) and security (.789) have the highest factor loadings at first component. Hence, these are grouped under the factor named Functionality. Sub-characteristics maturity (.896), recoverability (.775) and fault-tolerance (.659) have the highest factor loadings at second component, so these are grouped under the factor Reliability. Similarly, sub-characteristics understandability (.833), Learnability (.615) and operability (.547) are grouped under Usability; resource efficiency (.687) and time efficiency (.822) under Efficiency; testability (.861) and replaceability (.524) under Maintainability; interoperability (.787), adaptability (.540) and deployability (.596) are grouped under Portability.

Table 4 shows the component quality model with 6 characteristics and 16 sub-characteristics chosen from Table 1 and Table 3. The sub-characteristics are refined into attributes which are specific for black box software components and these can be directly measured by using metrics. It is to be noted that only those characteristics, sub-characteristics, attributes are incorporated in model which are statistically important to increase the reuse and quality of software component.

4 Design of the statistical study

4.1 Theoretical basis

The proposed component quality model is statistically investigated in this section. We investigated higher quality of which characteristics indicate higher Reusability and we are interested in determining positive association between Reusability and characteristics of quality model in regression model. To check this following hypotheses are assumed:

- H01: there is a significant association between characteristics and sub-characteristics of quality model
- H02: there is a significant association between stakeholders (User, Developer and Manager) and characteristics of quality model
- H03: there is a significant association between reusability and component quality model

4.2 Dependent and Independent Variables

Reusability is defined as the capability of a software component to be used in different contexts repeatedly. Reusability may be described in terms of user satisfaction, effectiveness and productivity [22]. These three factors are desired goals of all the stakeholders. Reusability is taken as a dependent variable. The independent variable is component quality model, which consists of six characteristics (Functionality, Reliability, Usability, Efficiency, Maintainability and Portability). These characteristics as independent variables are measured by judgments of respondents to the questionnaire concerning the significance of these characteristics to increase the reuse of software components.

4.3 Pilot survey

The questionnaire used for the survey is a structured and consisted of two major sections. The first section intended to collect the various demographic factors; the second section is intended to collect the responses about the reusability. The pilot survey was conducted on a group of 50 students of master degree. Subsequently, it was

Table 2: Important sub-characteristics of software components

Sub-Characteristics	Andreou[3]	Upadhyay[5]	Kalaimagal[9]	Sharma [10]	Alvaro[11]	Bertoa [23]	Rwashdeh[24]	Goulao [25]	Choi [26]
Security	1	1		1	1	1	1	1	1
Completeness	1	1							1
Suitability		1		1	1		1	1	1
Accuracy				1	1	1	1		
Interoperability	1			1	1	1	1	1	
Compliance				1	1	1	1	1	
Maturity	1	1		1	1	1	1	1	1
Recoverability				1	1	1	1	1	1
Fault tolerance				1	1			1	1
Learnability	1	1		1	1	1	1	1	1
Operability	1	1		1	1	1	1	1	1
Understandability			1	1	1	1	1		1
Configurability			1		1			1	
Time behavior		1		1	1	1	1	1	1
Resource behavior	1	1		1	1	1	1	1	1
Scalability				1	1				1
Customizability	1	1							
Testability	1	1		1	1	1	1	1	1
Changeability	1	1		1	1	1	1	1	1
Component stability			1	1	1				
Deployability		1			1				
Replaceability			1	1	1	1		1	1
Adaptability				1	1			1	1
Installability		1		1					

Table 3: Factor analysis results of sub-characteristics

	Component					
Sub-characteristics	1	2	3	4	5	6
Accuracy	0.725					
Suitability	0.744					
Security	0.789					
Maturity		0.896				
Recoverability		0.775				
Fault-tolerance		0.659				
Understandability			0.833			
Learnability			0.615			
Operability			0.547			
Resource Efficiency				0.687		
Time efficiency				0.822		
Testability					0.861	
Replaceability					0.524	
Interoperability						0.787
Adaptability						0.540
Deployability						0.596

Extraction Method: Principal Component Analysis.
Rotation Method: Varimax with Kaiser Normalization.

conducted on 15 Assistant professors of Information technology. As a result of their feedback, some questions were rephrased to make the language simple to understand.

4.4 Data collection

The study is based on primary data with a sample size of 124 respondents. A total of 250 experienced academicians, software professionals were contacted telephonically and personally and they were explained in

Table 4: Component Quality Model

Characteristics	Sub-characteristics	Attributes
Functionality	Security	Access control
		Encryption
		Auditing
	Accuracy	Correctness
	Suitability	Coverage
		User-Satisfaction
Reliability	Recoverability	Restore Point
	Fault-tolerance	Error-handling
		Availability
	Maturity	Versions Released
Usability	Learnability	Time to use
		Time to configure
		Time to administer
	Operability	Operating effort
		Administer effort
		Customizing effort
	Understandability	Documentation availability
		Documentation readability
		Demonstration/Training
Efficiency	Time efficiency	Throughput
		Response time
	Resource efficiency	Memory utilization
		Processor utilization
		Disk utilization
Maintainability	Testability	Test material
		Proof of test
	Replaceability	Backward-compatibility
Portability	Deployability	Complexity
		Documentation
	Interoperability	Hardware compatibility
		O.S. Compatibility
		Data-format Compatibility
	Adaptability	Adjustability

detail about the survey. These respondents are stakeholders who are directly involved in software engineering activities and are categorized into three categories namely user, developer and manager.

4.5 Measurement Scale

A five point rating scale ranging from not important to most important is used to capture the responses measuring significance of characteristics and sub-characteristics to increase the reuse of software components. Numerical values 1(Not Important) to 5(Most Important) are assigned to the ratings. This is ordinal scale where each category is ordered according to its level, such that 5 being superior to 4.

4.6 Reliability of Measures

Internal consistency of collected data is measured with Cronbach's Alpha. The closer the value of Cronbach's alpha coefficient to 1.0 the greater is internal consistency of the items in the scale [27]. The calculated value of Cronbach's alpha for our data is (0.862), which indicates a high level ($>$0.7) of internal consistency.

5 Analysis Method

A statistical analysis is conducted on six characteristics of quality model with respect to Reusability. At first stage, the correlation coefficients between pairs of characteristics and sub-characteristics were assessed. At second stage, univariate regression for individual characteristics was conducted. Since, our dependent variable Reusability is measured with five point rating scale (1-5) and rating has natural ordering, an appropriate regression model in this situation is an ordered probit regression [28], [29], [30].

An ordered probit model assumes that a dependent variable (Reusability) is a linear combination of the

independent variables (characteristics):

$$y_i^* = \beta' x_i + \varepsilon_i, where\ i = 1,2,\cdots,n$$

Here, y^* is a dependent variable coded as 1,2,...J, where J is the maximum rating given by a user to Reusability(in our case, J's value is 5), i is the number of respondents, β is a vector of coefficients, x_i is a vector of independent variables and ε is a standard error term, normally distributed over N[0,1].

$$y = 1 \quad \text{if } y^* \leq \mu_1$$

$$y = 2 \quad \text{if } \mu_1 \leq y^* \leq \mu_2$$

...

$$y = J \quad \text{if } \mu_{J-1} = y^* = \mu_J$$

The cutpoint or threshold value $\mu_J(\mu_1 \leq \mu_2 \leq \cdots \leq \mu_J)$, is to be estimated with parameter β. The range of dependent variable y^* is partitioned into J mutually exclusive regions. In other words, the ordinal variable y^* indicates the region into which a particular observed response falls. The cumulative probability distribution of y^* can be determined as follows:

$$\begin{aligned} pr[y=1] = pr(y)^* \leq \mu_1) &= pr(\beta' x_i + \varepsilon_i \leq \mu_1) \\ &= pr(-\beta' x_i \leq \varepsilon_i \leq \mu_1 - \beta' x_i) \\ &= \Phi(\mu_1 - \beta' x_i) - \Phi(-\beta' x_i) \end{aligned}$$

$$pr[y=2] = \Phi(\mu_2 - \beta' x_i) - \Phi(\mu_1 - \beta' x_i)$$

...

$$pr[y=J] = 1 - \Phi(\mu_{J-1} - \beta' x_i)$$

Therefore, the probability that y_i selects the jth alternative is given by,

$$pr[y=J] = \Phi(\mu_J - \beta' x_i) - \Phi(\mu_{J-1} - \beta' x_i), j = 0,1,\ldots,J$$

Here, μ_J and μ_{J-1} denote the upper and lower threshold values for category J and Φ denotes a standard normal cumulative distribution function (CDF). .

6 Results

6.1 Descriptive Statistics

For the given study STATA/SE 12.0 is used. Table 5 shows the demographic information of 124 respondents. There are (62.9%) male and (37.1%) female respondents, (52.4%) are 30-40 years of age and (39.5%) are below 30 years of age. Educational qualification of (63.7%) respondents is master degree whereas (31.5%) are graduates. Majority (61.3%) of respondents have more than 3 years experience wherein (21.0%) have 3-5 years, (38.7%) have 5-10 years and (22.6%) have more than 10 years of experience. All the respondents have expertise in various aspects of software engineering such as project management (9.7%), software development (41.9%), quality assurance (14.5%) and other software engineering aspect (33.9%). The academicians and research scholars who are well versed with Software Components and CBSE gave their responses in other category of expertise.

Out of 124 respondents, there are 48 (38.7%) users, 22 (17.7) managers and 54 (43.5) developers. Fig. 4 shows that there are more Users (29.8) with master degree and more Developers (22.5) with graduation degree. It can be seen Managers have more overall experience than Developers. This reflects the fact that employees in software companies with more experience tend to be managers.

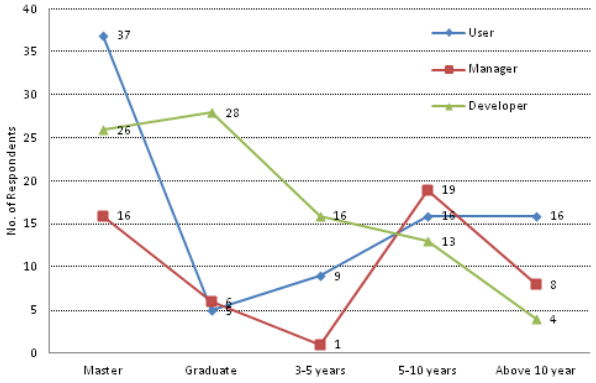

Fig. 4: Demographic information of stakeholders.

6.2 Significant Characteristics and Sub-characteristics

Table 6 shows the weighted mean and standard deviation for the responses given by different stakeholders to the questions on the characteristics of quality model. Functionality is essential characteristic for all kind of software components; its weighted mean (4.82) is higher than any other characteristic for all types of stakeholders (4.81, 4.81 and 4.86). The mean value shows users prefer Usability (4.58) and Reliability (4.56), Developers prefer Portability (4.78) and Maintainability (4.56) and Manager prefer Efficiency (4.55) the most. Last column (total) in the table gives the combined judgment of all the respondents related to characteristics.

Table 7 shows the mean, standard deviation and correlation coefficients of sub-characteristics which determine their importance and association with characteristics. The more closer the value of mean to 5,

Table 5: Demographic information

Gender	Male	78	62.90%
	Female	46	37.10%
Age	Below 30 years	49	39.50%
	Between 30-40 years	65	52.40%
	Between 40-50 years	9	7.30%
	Above 50 years	1	0.80%
Education qualification	Master	79	63.70%
	Graduate	39	31.50%
	Any other	6	4.80%
Total experience	0-1 year	5	4.00%
	1-3 years	17	13.70%
	3-5 years	26	21.00%
	5-10 years	48	38.70%
	Above 10 years	28	22.60%
Expertise	Project Management	12	9.70%
	Software Development	52	41.90%
	Quality Assurance/Testing	18	14.50%
	Other	42	33.90%
Quality certification of organization	ISO	20	16.10%
	Six Sigma	2	1.60%
	CMM/CMMI	57	46.00%
	Any other	45	36.30%

Table 6: Descriptive statistics of characteristics

	User		Developer		Manager		Total	
Characteristics	Mean	S.D.	Mean	S.D.	Mean	S.D.	Mean	S.D.
Functionality	4.81	0.673	4.81	0.392	4.86	0.351	4.82	0.511
Reliability	4.56	0.649	4.52	0.637	4.41	0.666	4.48	0.643
Usability	4.58	0.71	4.37	0.681	4.45	0.671	4.45	0.667
Efficiency	4.33	0.808	4.53	0.664	4.55	0.596	4.45	0.725
Maintainability	4.46	0.651	4.56	0.604	4.41	0.59	4.53	0.617
Portability	4.5	0.684	4.78	0.462	4.14	0.774	4.59	0.663

the more it is important to corresponding stakeholder and characteristic. Association of a sub-characteristic with corresponding characteristic for a stakeholder is calculated by Pearson correlation method (a factor is highly positively associated to its corresponding factor if its Pearson correlation value is closer to 1.0). For example, Accuracy's mean value is greater than all sub-characteristics so it is considered most important sub-characteristics of Functionality by all stakeholders. It is found highly correlated (P. Corr. 0.505 and Sig. 0.000 at 99% level) to Functionality.

6.3 Marginal Analysis

Table 8 shows ordered probit estimates of characteristics for the Reusability. The overall model is statistically signif-icant (p=0.0000), therefore our hypothesis has been accepted which states that there is a significant impact of characte-ristics of quality model on reusability. Most of characteristics (except portability) are significant ($p<0.05$) for Reusability. There are two cuts and three marginal effects for this model since respondents rated only three categories (5-Most important, 4-Very important and 3-Important) of the dependent variable (Reusability).

A high value of the coefficients as shown in Table 8 implies a greater impact on Reusability. The relative significance of marginal changes in the characteristics is calculated from the partial derivatives of Reusability with respect to individual characteristics in the model. The marginal value provides information on changes to the Reusability probabilities brought about by a unit change in the value of a characteristic from its mean value assuming mean values for the all other characteristics [31]. Table 8 shows Reliability (1.2475) is the most important characteristic which impacts Reusability and Portability (0.2372) is the least important characteristic. Statistically, a one unit increase in the Reliability brings about (1.2475) increase in the importance of Reusability. Also, z value of Reliability (5.36) is higher than other characteristics which indicate its significant influence on the independent variable to the dependent variable.

Table 7: Descriptive statistics of sub-characteristics

		User				Developer				Manager				Total			
Characteristics	Sub-Characteristics	Mean	S.D.	P. corr.	Sig.	Mean	S.D.	P. corr.	Sig.	Mean	S.D.	P. corr.	Sig.	Mean	S.D.	P. corr.	Sig.
Functionality	Security	4.27	1.005	0.297**	0.041	4.54	0.693	0.165	0.234	4.36	0.581	0.021	0.925	4.4	0.816	0.232**	0.01
	Accuracy	**4.81**	0.532	0.672**	**0**	**4.78**	0.42	0.319**	**0.019**	**4.82**	0.395	0.156	0.488	**4.8**	0.459	0.505**	**0**
	Suitability	4.35	0.699	0.234	0.109	4.59	0.533	0.445**	0.001	4.45	0.596	0.31	0.16	4.48	0.618	0.295**	0.001
Reliability	Recoverability	4.52	0.799	0.104	0.482	4.59	0.533	0.19	0.17	4.59	0.503	-0.187	0.404	4.56	0.641	0.093	0.307
	Fault-tolerance	4.65	0.758	-0.095	0.52	4.57	0.602	0.144	0.299	4.55	0.51	0.153	0.497	4.6	0.649	0.035	0.701
	Maturity	4.4	0.736	-0.12	0.416	4.44	0.664	0.114	0.411	4.36	0.727	-0.027	0.906	4.41	0.699	-0.005	0.958
Usability	Learnability	4.58	0.739	0.337*	0.019	4.5	0.795	0.139	0.315	4.45	0.671	0.184	0.412	4.52	0.749	0.222*	0.013
	Operability	4.38	0.733	0.085	0.556	4.48	0.574	0.066	0.635	4.36	0.727	0.07	0.757	4.42	0.664	0.065	0.476
	Understandability	4.81	0.673	0.300*	0.038	4.76	0.473	0.106	0.445	4.73	0.456	0.574**	0.005	4.77	0.553	0.256**	0.004
Efficiency	Time efficiency	4.69	0.719	0.366*	0.01	4.33	0.752	0.113	0.414	4.59	0.666	-0.097	0.668	4.52	0.738	0.153	0.089
	Resource efficiency	4.69	0.624	0.464**	0.001	4.28	0.811	0.444**	0.001	4.5	0.512	0	1	4.48	0.715	0.335**	0
Maintainability	Testability	4.67	0.694	0.283	0.051	4.69	0.469	0.23	0.095	4.23	0.685	0.465*	0.029	4.6	0.624	0.309**	0
	Replaceability	4.56	0.649	0.192	0.19	4.56	0.604	0.379**	0.005	4.45	0.51	0.302	0.172	4.54	0.604	0.291**	0.001
Portability	Deployability	4.08	1.069	0.103	0.487	4.28	0.656	-0.104	0.456	4.23	0.685	-0.151	0.503	4.19	0.843	0.013	0.889
	Interoperability	4.58	0.679	0.25	0.086	4.57	0.602	0.128	0.356	4.23	0.813	0.554**	0.007	4.52	0.681	0.330**	0
	Adaptability	4.46	0.683	0.139	0.346	4.44	0.744	-0.037	0.793	4.27	0.631	-0.372	0.088	4.42	0.7	0.007	0.94

***.Correlation is significant at the 0.01 level(2-tailed)*
**.Correlation is significant at the 0.05 level (2-tailed)*

Ordered Probit model estimates Reusability as a linear function of the characteristics as follows:

Reusability = 0.8268 x Functionality
+ 1.2475 x Reliability
+ 0.6772 x Usability
+ 0.9241 x Efficiency
+ 0.8304 x Maintainability
+ 0.2372 x Portability

Predicted probabilities of Reusability as shown in Fig. 5 are estimated as

$$pr(3 = \text{"important"}) = pr(\varepsilon_i \leq 18.54058 - Reusability)$$

$$pr(4 = \text{"very important"}) = pr(18.540508 - Reusability \leq \varepsilon_i \leq 21.6776 - Reusability)$$

$$pr(5 = \text{"most important"}) = pr(21.6776 - Reusability \leq \varepsilon_i)$$

As an example, the marginal values of changes in Functionality on the five probabilities ($pr1, \cdots$,pr5) of the Reusability are obtained by derivatives as follows [21]:

$$\frac{\partial pr(1 = \text{"not important"})}{\partial(Functionality)} = 0$$

$$\frac{\partial pr(2 = \text{"average"})}{\partial(Functionality)} = 0$$

$$\frac{\partial pr(3 = \text{" important"})}{\partial Functionality)} = -\Phi(-\beta' = x_i)\beta_1$$

$$\frac{\partial pr(4 = \text{"very important"})}{\partial(Functionality)} = -[\Phi(18.54 - \beta' = x_i) - \Phi(-\beta' = x_i)]\beta_1$$

$$\frac{\partial pr(5 = \text{"most important"})}{\partial Functionality)} = -[\Phi(21.67 - \beta' = x_i) - \Phi(18.54 - \beta' = x_i)]\beta_1$$

As shown in Fig. 5, among respondents, reusability is observed to be important (pr4) with highest probability (54.83%). However, (45.04%) respondents fall in the most important (pr5) category. As majority of the respondents are falling in the important category, the

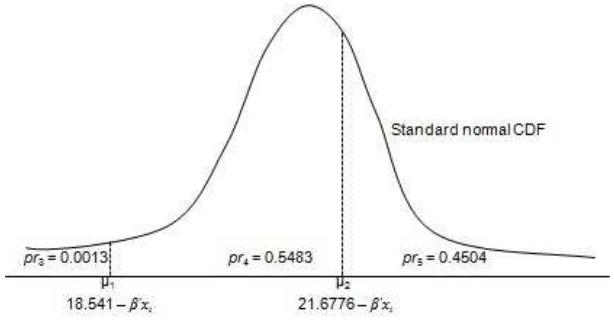

Fig. 5: Estimated probabilities of ordered probit regression.

improvement in their preference to most important category is required. To check the importance of each characteristic, the marginal effects of ordered probit model have been used as shown in Table 9, to explain the variation in most important category (pr5).

Table 9 shows that signs of all characteristics are positive it means Reusability will increase with the per unit increase in these variables. It is observed that all the characteristics (Functionality, Reliability, Usability, Efficiency, Maintainability and Portability) are positively and significantly affecting the probability of category 5. The most important characteristic affecting probability of category 5 is Reliability with p-value 0.000. Thus, an increase in the Reliability by one unit per its mean will enlarge the probability of category 5 by (0.4938%).

Table 10 shows the marginal satisfaction of each characteristic. Note that the marginal effects sum to zero because this follows from the requirement that the probabilities add to 1. It can be interpreted for example of functionality, increasing its value by one unit does not change the probability of a value of 1 for Reusability; the probability of a value of 2 also does not bring any change for reusability; the probability of a value of 3 for reusability decreases it by (0.0035); the probability of a value of 4 also decreases it by (0.3237) whereas the probability of a value of 5 increases it by (0.3273).

Table 8: Ordered probit model results

					[95% Conf. Interval]	
Reusability	Coef.	Std. Err.	z	P>|z |	Lower bound	Upper bound
Functionality	0.8268871	0.3578513	2.31	0.021	0.1255114	1.528263
Reliability	1.2475632	0.2328388	5.36	0	0.7912076	1.703919
Usability	0.6772183	0.2155222	3.14	0.002	0.2548026	1.099634
Efficiency	0.9241408	0.2153377	4.29	0	0.5020866	1.346195
Maintainability	0.8304527	0.2489749	3.34	0.001	0.3424708	1.318435
Portability	0.2372831	0.2224814	1.07	0.286	-0.1987725	0.6733387
cut1	18.54058	2.729175	(Ancillary parameters)			
cut2	21.6776	3.02884				

Table 9: Marginal effects on probability of reusability's category (5-most important)

Characteristics	dy/dx	Std. Err.	z	P>| z |	[95% Conf.Interval]		X
Functionality	0.3273238	0.14039	2.33	0.020	0.052174	0.602474	4.82258
Reliability	0.4938488	0.09195	5.37	0.000	0.313636	0.674061	4.47581
Usability	0.2680773	0.08546	3.14	0.002	0.100569	0.435585	4.45161
Efficiency	0.3658218	0.08458	4.32	0.000	0.200041	0.531602	4.45161
Maintainability	0.3287353	0.09885	3.33	0.001	0.13499	0.522481	4.53226
Portability	0.0939287	0.08793	1.07	0.285	-0.078415	0.266273	4.58871

Table 10: Marginal satisfaction value

Reusability probability	Functionality (4)	Reliability (1)	Usability (5)	Efficiency(2)	Maintainability(3)	Portability(6)
1 - Not Important	0.000	0.000	0.000	0.000	0.000	0.000
2 - Average	0.000	0.000	0.000	0.000	0.000	0.000
3 - Important	-0.0035	-0.0053	-0.0028	-0.0039	-0.0035	-0.001
4 - Very Important	-0.3237	-0.4885	-0.2651	-0.3618	-0.3251	-0.0929
5 - Most Important	0.3273	0.4938	0.268	0.3658	0.3287	0.0939

7 Discussion & conclusion

The role of quality model is to standardize the quality evaluation process and it helps stakeholders in selecting software according to their requirements and application area. A quality model always depends on the kind of target system to be evaluated. Because of this, a separate quality model is emphasized for software components since these are smaller in size, different in architecture and limited in functionality. Development approach and process is also different in case of software components. Therefore, in component quality model, only those aspects or characteristics of quality are needed to be considered which are essential for components. Ideally, it is expected that software system must possess the highest measure of quality but in practice, everybody involved with the system, from developers to managers, has to compromise and focus on the most important quality characteristics. Three major stakeholders namely user, manager and developers are identified which are involved at appropriate levels of software engineering activities. Accordingly, a quality model is proposed which best represents the quality properties expected by various stakeholders that are essential to increase the reuse of software components. This model is a general model which can be applied to any application by modifying or removing some of its characteristics or sub-characteristics. The purpose of this study is to determine the effect of characteristics of components quality model on the reusability. From the results it is determined that there is a significant association between component quality model and reusability. Further, it is also determined that stakeholders are interested in specific characteristics. The main concern of managers is high productivity and economy. Hence, they are interested in efficiency as it helps in increasing productivity and in cost cutting by optimal use of resources in reduced time. Users are ultimate end-users of the product and expect fault free product which leads them to incline towards usability and reliability characteristics. Developers are most technical people who develop the software products by integrating various software components. Results of survey indicated they considered portability and maintainability as most important characteristics. The number given under the characteristics in Table 10 denotes the importance of a characteristic to reusability. It shows the most important characteristic for reusability is Reliability which indicates that a component seems most reliable to stakeholders when it is capable to withstand or recover from failures and has a number of versions available in the market. The

second most important characteristic is efficiency which is the degree to which a component provides adequate performance at nominal use of resources. The third important characteristic is maintainability which is the capability of a component to be verified and replaced easily. Fourth important characteristic is functionality which is the capability of a component to provide stated services to the stakeholders. Functionality is considered most important characteristic by all the stakeholders as shown in Table 6. Usability is fifth important characteristic for reusability which is the capability of a component to easily learned, operate and understood. Portability is the capability of a component to be integrated and deployed under multiple conditions. Although reusability is least effected by portability but it cannot be discarded as it is inherent characteristic of components.

References

[1] R.S. Pressman, Software Engineering: A Practitioner's Approach, 5th ed., New York, USA, McGraw-Hill, 2001.

[2] A. Ampatzoglou, A. Kritikos, G. Kakarontzas and I. Stamelos, An Empirical Investigation on the Reusability of Design Patterns and Software Packages, Journal of Systems and Software **84(12)**, 2265-2283 (2011).

[3] A.S. Andreou and M. Tziakouris, A Quality Framework for Developing and Evaluating Original Software Components, Information and Software Technology **49(2)**, 122-141 (2007).

[4] S. Kalaimagal and R. Srinivasan, A Retrospective on Software Component Quality Models, SIGSOFT Softw Eng Notes **33(6)**, 1-9 (2008).

[5] N. Upadhyay, B.M. Despande and V.P. Agrawal, Towards a Software Component Quality Model, in Proc. 1st Intl. Conf. Computer Science and Information Technology, Bangalore **131(3)**, 398-412 (2011). DOI: 10.1007/978-3-642-17857-3_40

[6] P. Mohagheghi, The Impact of Software Reuse and Incremental Development on the Quality of Large Systems, Ph.D. Thesis, Deptt. of Comp. and Info. Science, Norwegian Uni. Sci. & Tech., Norway, Jul. (2004). DOI: 10.1.1.218.5267

[7] H. Mili, F. Mili, A. Mili, Reusing Software: Issues and Research Directions, IEEE Trans. Softw. Eng. **21(6)**, 528-562 (1995).

[8] D. Lucrédio, K.S. Brito, A. Alvaro, V. Garcia, E.S. Almeida, R.P.M. Fortes and S.L. Meira, Software Reuse: The Brazilian Industry Scenario, Journal of Systems and Software **81(6)**, 996-1013 (2008).

[9] S. Kalaimagal and R. Srinivasan, The Need for Transforming the COTS Component Quality Evaluation Standard Mirage to Reality, SIGSOFT Softw Eng Notes **34(5)**, 1-4 (2009).

[10] A. Sharma, R. Kumar and P.S. Grover, Estimation of Quality for Software Components: An Empirical Approach, SIGSOFT Softw Eng Notes **33(6)**, 1-10 (2008).

[11] A. Alvaro, E.S. de Almeida and S.R. de Lemos Meira, A Software Component Quality Framework, SIGSOFT Softw Eng Notes **35(1)**, 1-18 (2010).

[12] B. Behkamal, M. Kahani and M.K. Akbari, Customizing ISO 9126 Quality Model for Evaluation of B2B Applications, Information and Software Technology **51(3)**, 599-609 (2009).

[13] P.R. Srivastava and K. Kumar, An Approach Towards Software Quality Assessment, in Proc.3rd Intl. Conf. Information Systems, Technology and Management, Ghaziabad **31(6)**, 150-160 (2009). DOI:10.1007/978-3-642-00405-6_19

[14] M. Côté, W. Suryn and E. Georgiadou, In Search for a Widely Applicable and Accepted Software Quality Model for Software Quality Engineering, Software Quality Journal **15(4)**, 401-416 (2007).

[15] N. Bawane and C.V. Srikrishna, A Novel Method for Quantitative Assessment of Software Quality, International Journal of Computer Science and Security **3(6)**, 508-517 (2010).

[16] H.W. Jung, Validating the External Quality Subcharacteristics of Software Products According to ISO/IEC 9126, Journal of Computer Standards & Interfaces **29**, 653-661 (2007).

[17] B. Boehm, Managing Software Productivity and Reuse, Computer **32(9)**, 111-113 (1999).

[18] D.L. Nazareth and M.A. Rothenberger, Assessing the Cost-Effectiveness of Software Reuse: A model for planned reuse, Journal of Systems and Software **73(2)**, 245-255 (2004).

[19] S.S. Thapar, P. Singh and S. Rani, Challenges to the Development of Standard Software Quality Model, International Journal of Computer Applications **49(10)**, 1-6 (2012).

[20] M. Sibisi and C.C. van Waveren , A Process Framework for Customizing Software Quality Models, in Proc. 7th African Conf., Windhoek, 1-8 (2007). DOI: 10.1109/AFRCON.2007.4401495

[21] F. Carvalho and S.R.L. Meira, Towards an Embedded Software Component Quality Verification Framework, in Proc. 14th IEEE Intl. Conf. Engineering of Complex Computer Systems, Potsdam, 248-257 (2009). DOI: 10.1109/ICECCS.2009.26

[22] ISO/IEC 2001a, Software Engineering-Software Product Quality-Part 1: Quality model, ISO, Geneva, Switzerland, ISO/IEC 9126-1, (2001).

[23] M. Bertoa and A. Vallecillo, Quality Attributes for COTS Components, in Proc. 6th ECOOP Workshop on Quantitative Approaches in Object Oriented Software Engineering (QAOOSE), Spain, 2002. DOI: 10.1.1.16.7555

[24] A. Rawashdeh and B. Matalkh , A New Software Quality Model for Evaluating COTS Components, Journal of computer science **2(4)**, 373-381 (2006).

[25] M. Goulo and F.B. Abreu, Towards a Components Quality Model, 28th EUROMICRO Conference, Dortmund, Germany, Sep. 2002. DOI: 10.1.1.14.6351

[26] Y. Choi, S. Lee, H. Song, J. Park and S. Kim, Practical S/W Component Quality Evaluation Model, 10th Intl. Conf. Advanced Communication Technology, Gangwon-Do, **1**, 259-264 (2008). 10.1109/ICACT.2008.4493757

[27] W. Revelle and R. Zinbarg, Coefficients Alpha, Beta, Omega, and the glb: Comments on Sijtsma, Psychometrika **74(1)**, 145-154 (2009).

[28] W. E. Becker and P. E. Kennedy, A Graphical Exposition of the Ordered Probit, Econometric Theory **8(1)**, 127-131(1992).

[29] M.A. Abdel-Aty, Using Ordered Probit Modeling to Study the Effect of Atis on Transit Ridership, Transportation Research Part C Emerging Technologies **9(4)**, 265-277 (2001).

[30] R. D. McKelvey and W. Zavoina, A statistical model for the analysis of ordinal level dependent variables, The Journal of Mathematical Sociology **4(1)**, 103-120 (1975).
[31] A.C. Cameron and P. K. Trivedi, Microeconometrics Using Stata, 1st ed., Stata Press, 2009.

Simrandeep Singh Thapar is an Associate Professor at Amritsar College of Engineering & Technology, Amritsar, India. He received his Ph.D. degree in Software Engineering from IKGPTU, Kapurthala India in the year 2017 and Master degree in Computer Applications from Guru Nanak Dev University, Amritsar, India in 2004. He has 13 years of teaching experience. His research interests include reusability and software quality of software components.

Paramjeet Singh is a Professor at GZSCIET Bathinda, India. He received his Ph.D. degree in Optical Communication in the field of Computer Networks from BITS Pilani, India in the year 2009. He has 18 years of teaching and research experience. His interests include Computer Networks and Software Engineering.

Shaveta Rani is a Professor at GZSCIET Bathinda India. She received her Ph.D. degree in Optical Communication in the field of Computer Networks from BITS Pilani, India in year 2010. She has 17 years of teaching and research experience. Her interests include Computer Networks and Software Engineering.