Framework for Evaluating and Ranking the Reusability of COTS Components based upon Analytical Hierarchy Process

Sonu Mittal
Research Scholar,
SGVU, Jaipur, India

Pradeep Kumar Bhatia
Professor, GJUS&T, Hisar, India

Abstract - Component Based Software Systems (CBSS) are usually made up of reusable commercial off the shelf (COTS) software components, which are developed by third party. There are a large amount of COTS components available and the CBSS developer or the integrator of these components has to evaluate and compare the quality of these COTS components according to its requirements. Reusability is also one of the most important quality factors for the selection of these COTS components. It is the measure of the degree up to which that software component can be easily reused for the integration into new environment. There are large numbers of criteria proposed by researchers to evaluate the reusability of COTS components like customizability, interface complexity, understandability etc. This paper reviews all those reusability criteria and proposes a new reusability framework by quantifying six reusability criteria customizability, interface complexity, portability, commonality, observability and documentation level. The proposed framework can also be used in ranking the various chosen COTS components based upon Analytical Hierarch Process (AHP) approach whose flexibility and accuracy makes it better candidate for such type of multi-criteria decision making (MCDM) problem. The applicability of the proposed framework is also illustrated through an example.

Keywords - Component Based Software System (CBSS), Component Based Software Development (CBSD), Commercial of the shelf components (COTS), Reusability and Reusability criteria, Analytical Hierarchy Process (AHP), Multi Criteria Decision Making (MCDM) problem.

I. INTRODUCTION

The idea to construct software in the same way as hardware is constructed i.e. by integrating reusable components is becoming very popular in the last decade. Software reuse reduces development time, effort, cost and increases productivity and quality. Studies in software engineering confirm these benefits [3, 4]. Software reuse in its most common form can be seen in component based software development (CBSD). CBSD is an approach in which systems are built from well -defined, independently produced pieces, known as components. As these components are usually black boxes in nature, their internal structure and coding is not provided with them. The demand for reliable and qualitative software components is continuously rising. In response to meet the rising demand software firms have been producing variety of reusable Commercial Off-The Shelf (COTS) components that are customizable and tailored to meet specific needs of the Integrator/developer of component based software systems (CBSS). Selection of inappropriate COTS component may badly affect the development process and functioning of the CBSS. The task of COTS component evaluation and selection has become more multifaceted due to (i) complexity in accessing applicability of COTS components to the needs of the CBSS due to availability of large number of components in the market, (ii) existence of incompatibilities between various types of software and hardware systems, (iii) lack of metrics and quality measuring tools in this area, and (iv) their black box nature and lack of information provided by the vendors [2]. The task of COTS evaluation and selection is often assigned under schedule pressure and evaluators may not have time or experience to plan selection process in detail. Therefore, they may not use most appropriate method for selection [1]. Thus, evaluating COTS software that meets specific needs of the CBSS is complicated and time consuming decision making process. Selecting one of the low quality components may lead to failure of the entire system. Such two case studies have been already pointed by R.srinivasn et. al. [5]. Therefore, it is very necessary to build a quality framework through which we can evaluate the quality of these third party components for CBSS.

There is several existing quality models used to evaluate software systems, namely: McCall's, Boehm, ISO 9126, FURPS, Dromey, Triangle and Quality Cube etc. There are several quality models proposed by the researchers [18-21] to evaluate the quality of components; primarily based on the ISO 9126 model. As shown in our previous review work [1], none of them agrees on all the characteristics/sub characteristics to be as it is used for evaluating the components. Although most of the researchers [19,21] agree on the reusability feature to be added to the ISO 9126 model to extend it for the component quality evaluation model, yet they differs in their evaluation level and methods. This paper raises this quality issue of reusability evaluation of black box (COTS) components that are to be integrated in the CBSS and proposes an evaluation framework based upon the Analytical Hierarchy Process (AHP) approach.

After this short introduction, section II defines reusability and reviews various reusability factors proposed by the researchers Section III presents the reusability criteria and metrics proposed by us for the proposed reusability evaluation framework for COTS components. Section IV describes decision making framework the AHP approach for ranking the selected COTS components and also explains why to use this method for evaluating the reusability of COTS components for their integration in the CBSS. Section V illustrates the applicability of AHP approach for COTS components reusability evaluation and ranking of the reusable COTS component. The limitations, conclusions and future scope of this work are described in Section VI.

II. REUSABILITY AND REUSABILITY FACTORS

As for software reuse, a large number of definitions for software reusability can be found in existing literature, e.g., Kim et. al. [6] has defined software reusability as "a measure for the ease with which the resource can be reused in a new situation". According to gill et. al. [22] "Software reusability is an attribute that refers to the expected reuse potential of a software component." Gill discussed the importance of component characterization for better reusability. He also discussed several benefits of component characterization, which includes improved cataloguing, usage, retrieval and understanding eventually for better reusability.

It is important to distinguish between software reuse and reusability as the former is focused on the practice of reuse itself while the latter tries to make the prospective of artifacts for being reused measureable.

Reusability measures can further be distinguished as white box reusability and black box reusability [7,8]. Coding and Internal structure is available to the integrator in White box reusability while coding and internal structure is not available to the integrator in black box reusability measurement. As COTS components are black box in nature and their coding and internal details (Class structure etc.) are not available so we are concerned here with the black box reusability only. For COTS component reusability we can define it as "The measure of the degree up to which that software component can be easily reused for the integration into new environment."

To our knowledge, the first set of metrics for measuring the reusability of black- box components was proposed by Washizaki et al. [9] in 2003. They proposed to deem three main factors that are expected to affect reusability: understandability, adaptability and portability. Through an empirical analysis of Java Beans components, the authors have established thresholds for each proposed metric. For measuring the overall understandability, the metrics Existence of Meta-Information and Rate of Component Observability are defined. Rate of Component Customizability measures adaptability, while Self- Completeness of Component's Return Value and Self-Completeness of Component's Parameter measure portability.

In 2004, Boxall et al. [10] have proposed that the Understandability of a software component's interface is a major quality factor for determining reusability. To measure this, they have defined a set of metrics, including values such as Interface Size, Identifier Length or Argument Count. The authors have selected 12 components from different software systems in C and C++ to empirically validate their metrics and developed simple tools to automatically calculate them. The derived values have been compared against the expert knowledge of the authors judging the reusability of these components. Consequently, the authors have stated that more empirical research is necessary.

Again one year later, Rotaru et al. [12] have identified adaptability, composability and complexity of software components as determinants for their measure of reusability. The composability of a component is determined by the complexity of its interface. Adaptability is the ability of a component to handle environmental changes. Although a preliminary metric specification is given for all three aspects, the authors have stated that an empirical calibration is necessary to better understand its effects.

In 2009, Sharma et al. [11] have proposed an Artificial Neural Network approach to assess the reusability of software components. The authors have considered determining reusability by four factors: Customizability, Interface Complexity, Portability and Understandability. However, only customizability was quantitatively evaluated so far. The other three factors are only to be assessed qualitatively, i.e. merely ranked on a relative scale by experts.

In 2010, bhardwaj et al.[13] extended the work proposed by Sharma et. al. and applied a fuzzy logic technique. The authors have considered determining reusability by five factors: Customizability, Interface Complexity, Portability, Understandability and Commonality.
Here again portability and understandability were measured qualitatively and not quantitatively. The measurement for interface complexity and commonality were also restricted and no clear description of these metrics makes this framework difficult for practical evaluation of COTS components reusability.

III. QUANTIFYING REUSABILITY CRITERIA FOR THE PROPOSED FRAMEWORK

Based upon the above literature review, we have identified the five reusability criteria: Customizability, Interface Complexity, Portability, Commonality and Understandability as described by bhardwaj et. al. [13]. For Quantification and clear understanding purpose we are dividing understandability into further two sub-criteria Documentation level and Readability. Thus our proposed reusability evaluation framework will consist of six reusability criteria. Now, to evaluate the individual criteria, we need to use the related metric. Various metrics have been proposed by various researchers [7-13] but here units of all the available metrics may not necessarily be same. Like, some may be measured in numbers; others may be in ratio or just presence. As our evaluation framework will be based upon Analytical Hierarchy Process (AHP), we need to normalize these values in the range of 0 to 1, so that all can be fit under a unique scale. Within the normalized range, the highest value for a metric is 1 and is the maximum achievable level. In this section, we are describing these six reusability criteria with their proposed/selected quantifying metrics. The newly proposed metrics by us for interface complexity, Portability, commonality and documentation level are highlighted in bold.
The basis for quantifying them is that the metrics should be proper indicator to measure and rank the overall reusability of COTS component for their selection and integration into CBSS.

A. Customizability is defined as the ability to modify a component as per the application requirement. Better customizability will lead to a component with better reusability in applications and thus help in maintaining the component in the later phases.
It may be measured on the basis of writable properties available in the component. The following formula is used to evaluate this criterion by washizaki et. al. [9]:

$$Customizability = \frac{No.\ of\ writable\ properties}{Total\ number\ of\ Properties}$$

By using this metric, one can measure that how much an interface method can be customized. Therefore, it may be used to measure the reusability of the component. Customizability of a component may vary from 0 to 1.

B. Interface complexity Components are black box in nature. The source code of these components is not available. Application may interact with these components only through their well - defined interfaces. Interface acts as a primary source for understanding, use and implementation and finally maintenance for the component. Therefore, the complexity of these interfaces plays a lead role while measuring the overall complexity of the component. Complex interfaces will lead to the high efforts for understanding and customizing the components. Therefore for better reusability, interface complexity should be as low as possible. The following formula is used by us to evaluate this criterion

Interface Complexity = 1 - (Number of interfaces not required / Total number of interfaces provided)

More the unrequired interfaces more will be complexity and hence less will be reusability. The whole value is reciprocal because more complexity will lead to less reusability, thus the reciprocal value of complexity near to 1 will give us the reusability value for the component.

C. Portability It is the ability of a component to be transferred from one environment to another with little modification, if required. It is typically concerned with reuse of component on new platforms. The component should be easily and quickly portable to specified new environments if and when necessary, with minimized porting efforts and schedules. For better reusability, component should be highly portable, means; it should be supported by several platforms. Here, for the proposed work, portability may be defined as:

$$Portability = \frac{No.\ of\ platforms\ the\ component\ can\ support}{}$$

$$\textbf{\textit{Total no. of platforms that may be required by CBSS}}$$

By using this metric, one can measure that how many platforms can be supported by the COTS component. Therefore, it may be used to measure the reusability of the component. Portability of a component may vary from 0 to 1.

D. Commonality It is a factor that measures how well the functionality of COTS components matches to the functionality of CBSS and its standard domain model. If a component provides a superior functionality but does not meet the commonness of the domain requirements, its reusability for the organization will be limited. This factor is common for both functionality and reusability measurement and can be defined as:

$$\textbf{\textit{Commonality}} = \frac{\textbf{\textit{No. of functions common to provided by component and required by domain model}}}{\textbf{\textit{Total no of functions required by the domain model}}}$$

Commonality of a component may vary from 0 to 1.

E. Observability It may defined as the ability to understand component's functional elements as per described in its manual. Functional elements may be referred as the interfaces, operations, or events that a component may support or require from other components to achieve its functionality, i.e., to implement its services. As the source code of the component is not available better observability will lead to a component with better reusability in applications and thus help in using the component in the later phases.

It may be measured on the basis of readable functional elements available in the component. The following formula is used to evaluate this metric by washizaki et. al. [9]:

$$\textit{Observability} = \frac{\textit{No. of readable functional properties}}{\textit{Total number of functional Properties}}$$

By using this metric, one can measure that how many functional elements can be understood. Therefore, it may be used to measure the reusability of the component. Observability of a component may vary from 0 to 1.

F. Documentation Level As the source code of the COTS component is not available to the application developer; documentation is the only source from where he/she can understand the component. Documentation provides the ease with which a user can learn to operate, prepare inputs for, and interpret outputs of a system or component.

Here, the term documentation of component refers to component manuals, demos, help system, and marketing information. A good quality document must include functional description, installation details, system administrator's guide, system reference manuals etc. It may also require non -functional details, like performance, security issues, and previous maintenance activities, if any. It will help in understanding the component and reusing and integrating it easily in the CBSS. It will also help in implementing the maintenance activities with less effort. For present work, we categorize documentation quality on a *level scale of 1 to 5 (1 for very low quality documents provided and 5 for the highest quality documents provided).* The metric proposed by us that can be used to measure the documentation from this level scale can be described as follows:-

$$\textbf{\textit{Documentation Level}} = \sum_{i=1}^{n} \textbf{\textit{Level of each documentation } } D_i \textbf{\textit{ provided / n * 5}}$$

Where n is the total number of documents required by the CBSS to clear understanding of the COTS component. Documentation level metric value of a component may vary from 0 to 1.

All the COTS components reusability factors and their related metrics can be summarized as shown by Table 1.

Table 1 COTS component reusability criteria and related metrics

Sr. No.	Reusability Factor	Metric
1	Customizability	*No. of Writable properties/ Total No. of properties*
2	Interface complexity	***1 - (Number of interfaces not required / Total number of interfaces provided)***

3	Portability	***Total no. of platforms component can support/Total no. of platforms required by the system***
4	Commonality	***No. of functions common to provided by component and required by domain model/ Total No. of functions required by the domain model***
5	Observability	*No. of readable functional properties/ Total No. of properties*
6	Documentation level	$\sum_{i=1}^{n}$ ***Level of each documentation*** D_i ***provided / n * 5***

Thus for the four reusability criteria (interface complexity, portability, commonality and documentation level) out of six have been proposed new metrics by us. All the reusability metrics will give value between 0 and 1 with 1 as highest reusability value. These values can be easily used in the proposed AHP framework for evaluating the overall reusability of the COTS component.

IV. THE PROPOSED AHP FRAMEWORK

A. Why AHP approach

The main research question here is how to evaluate the reusable COTS component in facilitating component reuse for new software development. As there are basically, six criteria for the evaluation of overall reusability measurement, this problem can be refereed as Multi criteria decision making (MCDM) problem. MCDM tries to make preference decisions over the available alternatives that are characterized by multiple, usually conflicting, attributes [17]. Goal of the MCDM is: (i) to help decision makers to choose the best alternative, (ii) to sort out the alternatives that seems good among the set of available alternatives, and (iii) to rank the alternatives in decreasing order of their performance [16]. The process of evaluation and selection of the COTS software components reusability involves simultaneous consideration multiple criteria to rank the available alternative COTS components and select the best one. Therefore, evaluation and selection of the COTS component can be considered as multi criteria decision making problem.

The proposed solution for this study is development of COTS component reusability evaluation framework using the Analytical Hierarchy Process (AHP) based approach that suit with the problem. Our literature review reveals that although recent researchers [13] have proposed fuzzy logic based approach of evaluation of reusability that is easy to use and less time consuming than the AHP, yet we recommend the AHP because of its accuracy [15]. In fuzzy based approach decision makers can use linguistic terms to evaluate alternatives that improves decision making procedure by accommodating the vagueness and ambiguity in human decision making. However, it is difficult to compute fuzzy appropriateness index values and ranking values for all alternatives. AHP enables decision makers to structure a decision making problem into a hierarchy, helping them to understand and simplify the problem. Although it is time consuming technique because of the mathematical calculations and number of pair wise comparisons which increases as number of alternatives and criteria increases or changes yet it is preferred MCDM method due to its flexibility and accuracy in the result.

Another famous old method that can be applied to develop such type of framework is weighted scoring method (WSM). Weighted scoring method is easy to use and understand but weights to the attribute are assigned arbitrary and it is difficult task when a number of criteria are high. Another problem with weighted scoring method is that common numerical scaling is required to obtain final score. In AHP approach decision makers can compare each alternative that improves decision making procedure by accommodating the ambiguity in human decision making. AHP is especially suitable for complex decisions that involve the comparison of decision elements which are difficult to quantify. Its application has been reported in numerous fields, such as transportation planning, portfolio selection, corporate planning and marketing.

B. Evaluation using AHP

AHP was proposed by Dr. Thomas Saaty [14] in 1970s and has been applied in wide diversity of applications in different fields. This method allows consideration of both objective and subjective factors in selecting the best alternative. The methodology is based on three principles: decomposition, comparative judgments and synthesis of priorities. Decomposition principle calls for construction of hierarchical network to represent a decision problem

with top level representing overall goal and lower levels representing criteria, sub-criteria (if any) and alternatives as shown in Figure 1.

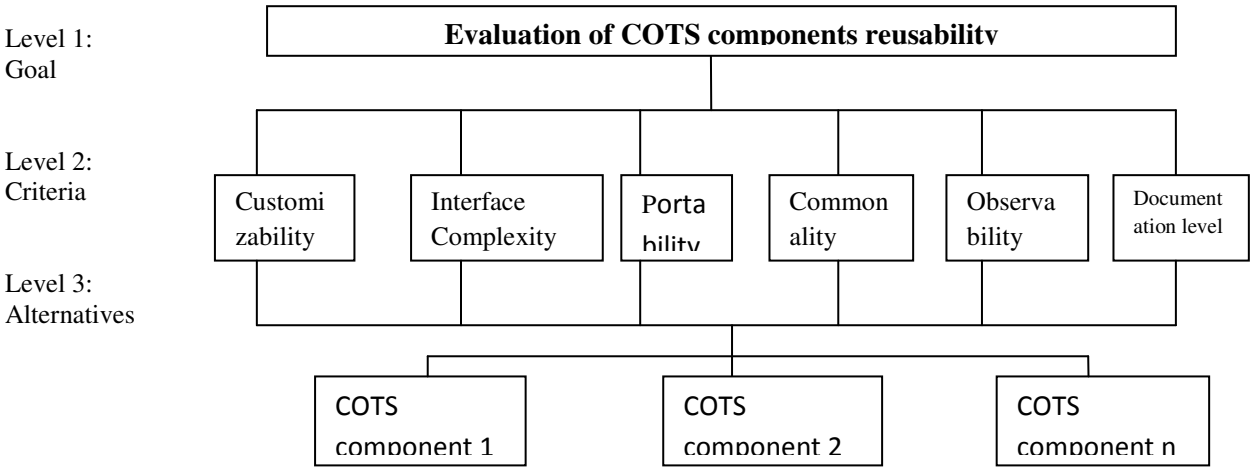

Figure 1: Hierarchical representation of the problem

With the comparative judgments users are required to set up a comparison matrix at each level of hierarchy by comparing pairs of criteria or sub-criteria. In general comparison takes the form: "How important is criteria A_i relative to criteria A_j? Questions of this type are used to establish weights of criteria. The possible judgments used for pair wise comparison and their respective numerical values are described in Table 2. Similar questions are to be answered to access the performance scores of alternatives on the subjective (judgmental) criteria.

Table 2 Pair wise comparison judgment matrix values between elements X and Y.

Comparison Judgment statement	**Matrix Value**
X is equally preferred to Y	1
X is moderately preferred over Y	3
X is strongly preferred over Y	5
X is very strongly preferred over Y	7
X is extremely preferred over Y	9
Intermediate values; when compromise is needed	2,4,6,8
Preference of Y compared to X	1/2, 1/3, 1/4, 1/5, 1/6, 1/7, 1/8, 1/9

The third stage in AHP is identifying preferred alternative by calculating aggregate score of each alternative. Aggregate score is calculated by multiplying normalized score with the weight (importance) of that criterion and sum the results for all criteria. The preferred alternative will have the highest score.

V. ILLUSTRATION BY AN EXAMPLE

The above described method can be illustrated by taking an example. Let us consider an integrator/developer of an e-commerce CBSS have to evaluate the reusability of three candidate COTS components (C1, C2 and C3) and out of these three alternative components he/she has to choose the one which has better reusability. The three candidate components are evaluated for each of the six evaluation criteria are as shown in the Table 3.

Table 3 Details of three COTS components based upon evaluation criteria

Reusability Evaluation Criteria	**COTS Component C1 Metric values**	**COTS Component C2 Metric values**	**COTS Component C3 Metric values**	**Integrator Requirements**
Customizability	12/30=0.40	34/50=0.68	12/40=0.30	1

Interface complexity	1-(8/20)=0.60	1-(12/15)=0.20	1-(18/20)=0.10	1
Portability	3/7=0.43	4/7=0.57	4/7=0.57	1
Commonality	12/30=0.40	10/30=0.33	18/30=0.60	1
Observability	21/30=0.70	40/50=0.80	36/40=0.90	1
Documentation level	(3+4+5)/3*5=0.80	(3+3+3)/3*5=0.60	(2+3+4)/3*5=0.60	1
Total Reusability	**3.33**	**3.18**	**3.07**	**6**

From the above table observations it is clear that for customizability criteria C2 is better than C1and C1 is better than C3. While for Interface complexity criteria level C1 is better than C2 and C3. Portability of C2=C3. Both of these are better than C1. Similarly for commonality criteria C3 > C1 > C2. The Observability of C3 > C2 > C1 and documentation level of C1 > C2=C3. Thus here for one criteria component C1 seems better than others while for other criteria component C2 or C3 seems better reusability than other components. So it is very difficult to evaluate the overall reusability of these components and find which component has better reusability. It comes under MCDM problem and can be easily solved by AHP approach.

The flexibility and the accuracy produced by the AHP approach makes it better candidate for solving such type of problems. Here the integrator/developer of CBSS can consider all the six reusability evaluation criteria or may add/delete from these criteria and can assign proper weightage to each criteria according to the type of application which he/she developing, hence provide flexibility to the framework according to its need. Let us consider the order of priority for all the six reusability evaluation criteria for the ecommerce CBSS is given by the integrator as Customizability > Interface Complexity = Portability = commonality > Observability = documentation level.

We have used an open source tool Open Decision Maker (ODM) v 1.0.1, to make sensitivity analysis between the alternative factors and to calculate the overall weights using AHP. The critical consistency ratio is under 0.1 in all the cases. (See appendix A). The results are shown in the Table 4, Table 5 and Table 6.

Table 4: Weight values for each criteria

Main Criteria Weighting:

	Name	Value
1.	Customizability	30.06%
2.	Interface Complexity	17.27%
3.	Portability	17.27%
4.	Commonality	17.27%
5.	Observability	9.07%
6.	Documentation level	9.07%

Table 5:

Alternative-Main Criterion-Matrix:

	Commonality	Customizability	Documentation level	Interface Complexity	Observability	Portability
C1	23.85%	23.85%	60.00%	73.06%	16.34%	14.29%
C2	13.65%	62.50%	20.00%	18.84%	29.70%	42.86%
C3	62.50%	13.65%	20.00%	8.10%	53.96%	42.86%

Table 6: Overall reusability score obtained by each component

Alternatives Ranking:

	Name	Value
1.	C2	36.68%
2.	C1	33.16%
3.	C3	30.16%

The result in table 6 shows that the overall reusability of COTS component C2 is best (36.68 %) under these conditions. Thus this framework not only helps in evaluation of reusability of COTS components but also helps in ranking of these components according to various reusability criteria weights. Table 5 shows the evaluation of each component according to values obtained by each criterion. Each criterion is pair wise compared with each alternative (see appendix A). Although it seems to have better reusability of COTS component C1 (33.3%) among all the components, yet after proper weighting and doing pair wise comparison of each criteria with some sensitivity analysis through AHP the component C2 shows better reusability (36.68 %) among all the components.

VI. LIMITATIONS, CONCLUSION AND FUTURE WORK

Evaluation of COTS components reusability is a puzzling and time consuming task due to multi criteria decision based problem. We can easily evaluate the reusability of COTS components using proposed AHP based framework. We can also rank each alternative component according to its reusability. The Pair wise comparison of each alternative and each criteria makes it supportive for evaluation of quantitative (customizability, commonality, etc.) as well as qualitative (documentation level) criteria and provides more accurate results. The flexibility of choosing and weighting each criterion according to requirements of CBSD also makes this framework strong for evaluating the reusability of COTS components.

Although the flexibility and the accuracy provided by this framework makes it stronger candidate for evaluating the reusability of COTS components yet it has certain limitations. If the numbers of criteria are high then a lot of pair wise comparison work have to be done which is very tedious and time consuming work and makes this framework complex. Although with the help of AHP automation tools (e.g. ODM) the work can be easily done yet if user requirements change then pair wise comparison has to be done again. We have illustrated this approach through an example but still empirical validation of the framework yet to be done.

VII. REFERENCES

[1] Mittal S., Bhatia P.K., "Software component quality models from ISO 9126 perspective: A review" IJMRS's International Journal of Engineering Sciences, Vol. 2, Issue 2, pp- 6-13, ISSN: 2277-9698, June, 2013..

[2] Kontio, J., Caldiera, G., Basili, V.R., "Defining factors goals and criteria for reusable component evaluation", In: CASCON'96 conference, Toronto, Canada, November 12–14 (1996).

[3] Krueger, C. W.: Software reuse, ACM Comput. Surv., 24, 131-83 (1992).

[4] Mohagheghi, P., and Conradi, R."Quality, productivity and economic benefits of software reuse: a review of industrial studies", Empirical software Engg., 12, 471-516 (2007).

[5] S. Kalaimangal and R. Srinivasan, "A Retrospective on Software Component Quality Models," ACM SIGSOFT Software Engineering Notes, Vol.33, No.5, (November, 2008), pp.1-9

[6] Y. Kim and E.A. Stohr.," Software reuse: survey and research directions", Journal of Management Information Systems - vol.14, March 1998, pp.113-147.

[7] J. Poulin, "Measuring software reusability", Proceedings of the International Conference on Software Reuse: Advances in Software Reusability, 1-4 Nov. 1994, pp.126-138.

[8] Miguel Goulão, Fernando Brito e Abreu, "An overview of metrics-based approaches to support software components reusability assessment" In:CoRR, abs/1109.6802 (2011).

[9] H. Washizaki, H. Yamamoto, and Y. Fukazawa, "A Metrics Suite for Measuring Reusability of Software Components", Proceedings of the 9th International Symposium on Software Metrics (METRICS '03), IEEE Computer Society, Washington, DC, USA, 2003, pp.211-223.

[10] M.A.S. Boxall and S. Araban, "Interface Metrics for Reusability Analysis of Components", Australian Software Engineering Conference (ACWEC'04), Melbourne, Australia, 2004, pp.40-50.

[11] A. Sharma, P.S. Grover, and R. Kumar, "Reusability assessment for software components", SIGSOFT Software Engineering Notes, vol.34, No.2, February 2009, pp.1-6.

[12] O.P. Rotaru and M. Dobre, "Reusability Metrics for Software Componenents", ACS/IEEE International Conference on Computer Systems and Applications (AICCSA'05), Washington DC, USA, 2005, pp.24-31.

[13] V. bhardwaj , "Estimating reusability of software components using Fuzzy logic", M.tech. thesis, 2010.

[14] TL Saaty," Multicriteria decision making: The analytic hierarchy process", McGraw-Hill, 1980.

[15] Perini, A., Ricca, F., Susi, A., "Tool-supported requirements prioritization: comparing the AHP and CBRank methods." Information and Software Technology 51, 1021–1032 (2009).

[16] Mollaghasemi, M., Pet-Edwards, J., "Technical briefing: making multiple objective decisions", IEEE Computer Society Press, Los Alamitos, CA 2007.

[17] Yoon, K., Hwang, C.," Multiple Attribute Decision-Making: An Introduction" Sage Publisher. 1995.

[18] M. Bertoa, A. Vallecillo, "Quality Attributes for COTS Components", In the Proceedings of the 6th International ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE), Spain, (2002).

[19] S.D. Kim and J.D Park, "C-QM: A Practical Quality Model for Evaluating COTS Components", Proceedings of the 21st IASTED International Conference on applied informatics, Innsbruck, Austria (February,2003).

[20] Choi, Y., Lee, S., Song, H., Park, J., Kim, S., "Practical S/W Component Quality Evaluation Model", In the 10th IEEE International Conference on Advanced Communication Technology (ICACT), Korea (2008).

[21] A. Sharma, R. Kumar and P.S. Grover, "Estimation of Quality for Software Components - an Empirical Approach," ACM SIGSOFT Software Engineering Notes, Vol.33, No.5,(November, 2008), pp.1-10

[22] Gill, N.S., "Importance of Software Component Characteristics for Better software Reusability", ACM SIGSOFT SEN, 2006. 31(1): p. 1-3.