Quantification of Software Quality Parameters using Fuzzy Multi Criteria Approach

Jagat Sesh Challa, Arindam Paul, Yogesh Dada, Venkatesh Nerella, Praveen Ranjan Srivastava

Abstract — **Software quality is the measure of appropriateness of the design of the software and how well it adheres to that design. There are some metrics and measurements to determine the software quality. Software quality measurement is possible only by quantifying the characteristics affecting the software quality. For measuring the quality, the parameters or quality factors are considered that vary over a domain of discourse. The quality factors stated in ISO/IEC 9126 model are used in this paper. Due to the unpredictable nature of these factors or attributes fuzzy approach has been used to estimate the software quality.**

I. Introduction

SOFTWARE Engineering is a framework that encompasses a process, a set of processes and an array of tools for a person who builds software. In the current scenario the importance of Software Engineering has been growing due to the recent changes to the trends of Modern IT Industry. This has also resulted in growing focus and research in the field of Software Quality.

Researchers have developed various models to study and understand the Software Quality. Various Software Quality Models by previous researchers include McCall's Model [1], Boehm's Model [2], FURPS Model, Dromey's Model [3], Sehra's Model, ISO/IEC 9126 Model [4], etc. ISO/IEC 9126 [4] is the most recent model and adheres to the results of almost all other models. The current work considers ISO/IEC 9126 [4] Model as the base model to estimate the software quality.

The remainder of the Paper is as follows. Section 2 describes the background work; Section 3 describes the Software Quality Model; Section 4 describes the procedure adopted to quantify the software quality along with a case study; Section 5 describes some analysis and Section 6 mentions conclusions and future work.

II. Background Work

Researchers have made very good attempts to estimate the software quality parameters. Reference [5] has classified software quality into developer's, user's and project manager's perspectives. Weighted average of the factors affecting these perspectives has been taken to compute final software quality. Reference [6] has subdivided the quality factors into criteria and sub criteria and then the metrics affecting these sub criteria have been quantified. They clearly elucidated their approach by quantifying Portability. Reference [7] has quantified the software quality criteria mentioned in ISO/IEC 9126 [4] for Component based Software Development Model using Analytical Hierarchy Process (AHP). Reference [8] made an attempt to evaluate the cost of software quality. Reference [9] considers the Software Quality in terms of Quality, Effort and Cycle Time and tried to quantify the same. A systemic quality model was developed by [10] for evaluating the software product. They considered various characteristics and sub characteristics influencing the software quality to estimate the software quality. Reference [11] tried to evaluate the code quality using various metrics with the help of Analytical Hierarchy process model. References [12] and [13] tried to rank various software products on the basis of SRS (Software Requirement Specifications) in the order of software quality using fuzzy multi criteria approach.

Current Work attempts to quantify the software quality in various perspectives including Developer, User and Project Manager, on the basis of ISO/IEC 9126 model [4]. Fuzzy has been employed to measure the unpredictable values of the metrics affecting software quality.

III. Software Quality

Software Quality is a measure of how successful is the Software in meeting the needs and demands of users and achieving the goals of developers. Quality model consists of a set of characteristics, sub characteristics and metrics for evaluating software quality.

A. *ISO/IEC 9126 Model*

The latest Software Quality Model proposed by ISO (International Standard Organization) is the ISO/IEC 9126 Model [4]. This model defines software quality in terms of six characteristics. They are Functionality, Efficiency, Maintainability, Portability, Reliability and Usability. Table 1 mentions the characteristics and sub characteristics of this model in brief. For further details on this model please refer to [4].

After considering various additions to ISO/IEC 9126 Model by various researchers, a new model has been designed for evaluating the software quality. Reference [5] has proposed Software Quality model is terms of three perspectives – Developer's Perspective, User's Perspective and Project Manager's Perspective. The characteristics of the ISO/IEC 9126 Model are distributed into these three Perspectives, as shown in Table 1. Reference [7] proposes addition of some sub characteristics to the model. They are -

Manuscript received February 18, 2001. Quantification of Software Quality Parameters using Fuzzy Multi Criteria Approach.

J. S. Challa is pursuing M.E. (Software Systems) at Birla Institute of Technology and Science, Pilani, India. (Phone: +91 9602803764; e-mail: jagatsesh@gmail.com).

A. Paul, Jr., is pursuing M.E. (Software Systems) at Birla Institute of Technology and Science, Pilani, India. (arindampaul.bits@gmail.com).

Y Dada is pursuing M.E. (Software Systems) at Birla Institute of Technology and Science, Pilani, India. (yogeshdada05@gmail.com).

V. Nerella is pursuing M.E. (Software Systems) at Birla Institute of Technology and Science, Pilani, India. (venkatesh.nerella56@gmail.com)

P. R. Srivastava is pursuing Ph.D. at Birla Institute of Technology and Science, Pilani, India. (praveenr@bits-pilani.ac.in)

978-1-61284-764-1/11/$26.00 ©2011 IEEE

i. **Customizability**: It shows the degree to which the software is customizable. (Added to Functionality)
ii. **Scalability**: It shows the degree to which the software is scalable. (Added to Efficiency)
iii. **Track-ability**: It explains the degree to which the Software is Track-able. (Added to Maintainability)
iv. **Reusability**: It gives idea of how reusable the software is. (Added to Usability)

Apart from this [5] proposes three characteristics to be added to the Project Manager's Perspective. They include – Cycle Time, Cost and Schedule Pressure. This is illustrated in Table 1.

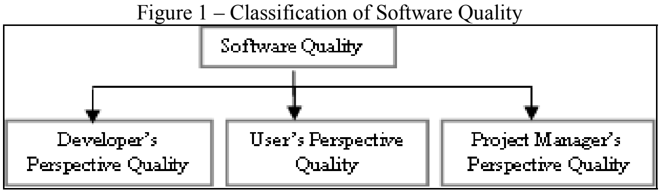
Table 1 (ISO/IEC 9126 Model)

ISO/IEC 9126 Model with new Sub Characteristics Added to it					
DEVELOPER's Perspective				**USER's Perspective**	
Functionality	**Efficiency**	**Maintainability**	**Portability**	**Usability**	**Reliability**
Suitability	Time Behaviour	Analyzability	Replaceability	Understandability	Maturity
Accuracy	Resource Behaviour	Changeability	Adaptability	Learnability	Recoverability
Interoperability	Efficiency Compliance	Testability	Installability	Operability	Fault Tolerance
Security	***Scalability***	Stability	Co – Existence	Attractiveness	Reliability Compliance
Functionality Compliance		Maintainability Compliance	Portability Compliance	Usability Compliance	
Customizability		***Trackability***		***Reusability***	

IV. PROCEDURE WITH THE HELP OF A CASE STUDY

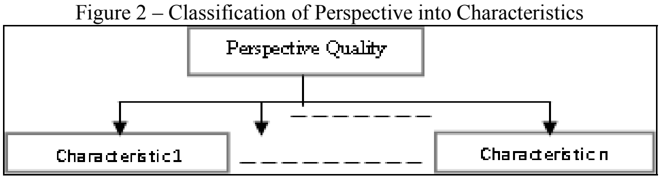
Firstly, the software quality model is classified into three perspectives – Developer's, User's and Project Manager's perspective as shown in Figure 1.

Figure 1 – Classification of Software Quality



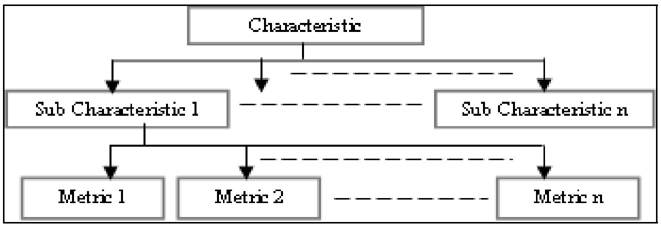
Each perspective is further sub divided into various characteristics as shown in Figure 2.

Figure 2 – Classification of Perspective into Characteristics



Every characteristic is further sub divided into sub characteristics and every sub characteristic is further sub divided into metrics as shown in Figure 3.

Figure 3 – Classification of Characteristic into Sub Characteristics & Metrics



At every level from perspective to characteristic to sub characteristic to metric, every parameter is associated with a corresponding rating (r_i) and weight (w_i). So first the fuzzy weighted average of the metrics is taken to evaluate the rating of the sub characteristic. Then the fuzzy weighted average of the sub characteristics is taken to get the rating of the characteristics. Then fuzzy weighted average of the characteristics is taken to get the rating of the perspective. Fuzzy weighted average of different perspectives is taken to get the final software quality in terms of a fuzzy set. Centroid Formula is then employed on this triangular fuzzy set to calculate the final software quality.

The computations performed in the course of this paper quantify the software quality in the range [0 to 1]. The fuzzy ratings obtained after performing the calculations are defuzzified using Centroid formula to get the software quality in the range [0 to 1].

For further evaluation and working of this model, a sample case study has been chosen. The algorithm designed is applied on software called COURSE MANAGEMENT TOOL (CMT) which is used as internal development software at Birla Institute of Technology and Science, Pilani, India (www.bits-pilani.ac.in). The evaluation is the software quality is shown step by step in the following paragraphs.

First the list of all the metrics has been listed in Tables 2a, 2b, 2c. Functionality, Efficiency, Maintainability and Portability belong to Developer's Perspective (Table 2a); Reliability, Usability belong to User's Perspective (Table 2b); Cost, Schedule Pressure and Cycle Time belong to Manager's Perspective (Table 2c).

Table 2a (Metric Inputs of Developer's Perspective)

Characteristic	**Sub Characterisitic**	**Inputs for Metrics calculation**	**D1**	**D2**	**D3**
FUNCTIONALI	Suitability	Total Number of Operations Provided	18		
		Number of Operations not suitable	3		
	Accuracy	Number of Operations Meeting Required Accuracy	10		
		Total no of operations	50		
		Whether Required Precision is satisfied or not	Yes	Yes	No
	Interoperability	Database Used in the Software	Sql server2000		
		Usage Of Multimedia and Graphics	Too low	Too High	Too low
		File System Support	present		
	Security	Number of Access Controllability provided	1		

TY		Number of Access Controllability required provided	1
		Software Enables Restricted User Access or Not	Yes
EFFICIENCY	Time Behavior	Number of Global Variables	30
		Programming Language Used	.Net
		Processing Capability of the Machine	Intel
	Resource Utilization	Percentage CPU usage for the execution of this Component	27
		Supports External Usage of Printers, Scanners, etc	Yes
	Compliance	Whether software adheres to Efficiency Compliance Standards or not	Yes
	Scalability	Whether the software is scalable to include more number of Users or not	Yes
MAINTAINABILITY	Analyzability	Number of Modules	7
		Kilo Lines of Source Code	8k
		Average length of Each of Module	2k
		Programming Language used	.net
		Experience of Manager in Software Firm	2 years
		Experience in Managerial Position	Less than 2 years
		Give KLOC	2
		Give Team Size	4 persons
		Cyclomatic Complexity	15
		Number of Versions Available	1
		CMM Levels	1
		Skills – Technical Skills (Analysis, Database, Programming, Mgmt.)	3 out of them
		Skills – Organizational skills – Industry experience (years)	3
		Skills – Organizational skills – Team skills – Average quality of citizen	good
		Skills – Organizational skills – Team skills – Cooperation among team	excellent
		Skills – Organizational skills – Team skills – Overall performance of team	good
	Changeability	Total Number of Properties	24
		Total Number of Customizable Properties	22
	Testability	Whether Sufficient Number of Test Cases are provided or not	Yes
	Maintainability Compliance	Adherence Maintainability Compliance standards	Yes
	Trackability	Presence of Functional and Behavioural Tracking System	Yes
		Ease of Tracking Older Versions	Very easy
Portability	Adaptability & Installability	Operating Systems Supported	Windows + Linux
		Use of Intrinsic Tools	Yes
		Pre – requisite Packages needed	Packages Available popularly
	Co-existence	Frequency of Deadlocks	Rarely / Frequently / rarely
	Portability Compliance	Adherence to Portability Compliance Standards or not	Yes

Table 2b (Metric Inputs of User's Perspective)

Characteritisc	**Sub Characteristic**	**Inputs for Metrics Calculation**	**U1**	**U2**	**U3**	**U4**	**U5**
Reliability	Maturity	Number of Versions released so far	1				
	Fault Tolerance	Exceptional Handling provided or not	Yes				
		Number of functionalities	10				
		Number of functionalities successfully met	6				
	Recoverability	Availability of Data Backup	Yes				
	Compliance	Adherence to Compliance Standards	Yes				
Usability	Understandability	Documentation	No				
		Help System provided or not	Yes				
		Training of the Software Provided or Not	Not				
		Subjectively Pleasing or Not	No	Yes	No	No	Yes
		Error Handling and Popups present or Not	No				
		Online Help Support Provided or Not	Yes				
		International Language Support	English				
	Operability	Complexity of Functionalities	Easy	Average	Very Easy	Average	Easy
		Type of Interface	GUI				
		Ease of Use and Navigability	Average	Comfortable	Average	Average	Average
	Attractiveness	Usage of Graphics	Average	Attractive	Average	Average	Average
	Compliance	Adherence to Compliance Standards	Yes				
	Reusability	Total Number of customizable Properties	20				
		Total Number of Observable Properties	21				
	Learn-ability	No of observable properties	46				
		Total no of properties	67				
		Type of interface	GUI				

Table 2c (Metric Inputs of Project Manager's Perspective)

Characteristics under Project Manager's Perspective	**Inputs for calculating metrics**	**PM1**
Cycle Time	Cycle Time of the project with relative to the total project size	Low
Cost	Relative Cost of the Project	Medium
Schedule Pressure	Comparative Schedule Pressure	Very High

Every metric, as discussed earlier, is associated with corresponding rating and weight. The rating of the metric is calculated by fuzzifying the value of the metric. For example if we take metric number of global variables, it can be fuzzified in the following manner as shown in Table 3.

Table 3 – Fuzzification of Global Variables

No of Global Variables	**Fuzzy Value**
< 10	VH
10 to 20	H
20 to 30	M
> 30	L

Number of global variables = 30, so it is fuzzified as "Medium (M)". Also the weights assigned to it by three different developers are Very Low, Low and Medium.

Now the triangular fuzzy number is assigned to the rating of the metric on the basis of the Table 4. This table serves as the basis to assign triangular fuzzy number for the ratings of metrics listed in Table 2.

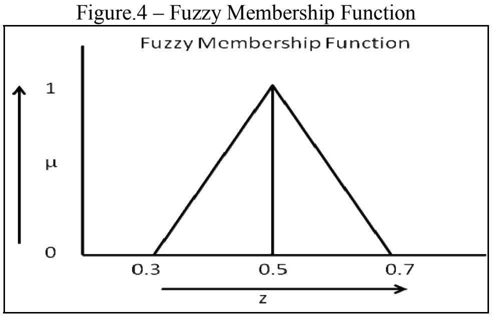
Table 4 – Triangular Fuzzy Sets for fuzzifying Ratings

Importance of Criteria	**Fuzzy Ratings**
Very Low	(0.0,0.1,0.3)
Low	(0.1,0.3,0.5)
Medium	(0.3,0.5,0.7)
High	(0.5,0.7,0.9)
Very High	(0.7,0.9,1.0)

So $r_{\text{global variables}}$ will be (0.3, 0.5, 0.7).

Each triangular number can be represented in the form of Fuzzy Membership function as shown in Fig.4. The membership function is a graphical representation of the degree of participation of inputs describing the system.

Figure.4 – Fuzzy Membership Function



Similarly we assign the triangular fuzzy number to the weights of the metrics as shown in Table 5. This table serves as the basis to assign triangular fuzzy number for the weights of all the metrics listed in Table 2.

Table 5 – Fuzzy Sets for fuzzifying Weights

Importance of Criteria	**Fuzzy Weights**
Very Low	(0.0,0.0,0.25)
Low	(0.0,0.25,0.5)
Medium	(0.25,0.5,0.75)
High	(0.50,0.75,1.0)
Very High	(0.75,1.0,1.0)

So $w_{\text{global variables}}$ is average of (Very Low, Low and Medium). This is average of (0.0,0.0,0.25), (0.0,0.25,0.5) ,(0.25,0.5,0.75) which is equal to (0.08,0.25,0.50)

Similarly we get the ratings and weights of other metrics influencing the sub characteristic *Time Behaviour* as shown below.

Table 6 – Ratings of Metrics under Time Behaviour

Metrics	**Average Rating**	**Average Weight**
Global Variables	(0.30,0.50,0.70)	(0.08,0.25,0.50)
Compiler or Interpreter	(0.70,0.90,1.0)	(0.50,0.75,0.92)
Processing Capability	(0.30,0.50,0.70)	(0.50,0.75,0.92)

Now fuzzy weighted average of the above metrics can be taken to get the fuzzy rating of the sub characteristic ***Time Behaviour***. This is explained below.

$r_{\text{global variables}} * w_{\text{global variables}} = (0.30,0.50,0.70) * (0.08,0.25,0.50) = (0.02,.013,0.35)$

$r_{\text{compiler or interpreter}} * w_{\text{compiler or interpreter}} = (070,0.90,1.0)* (0.50,0.75,.92) = (0.35,0.68,0.92)$

$r_{\text{processing}} * w_{\text{processing}} = (0.30,0.50,0.70) * (0.50,0.75,0.92) = (0.15,0.38,0.64)$

Hence, $r_{\text{Time Behaviour}} = r_{\text{global variables}} * w_{\text{global variables}} + r_{\text{compiler or interpreter}} * w_{\text{compiler or interpreter}} + r_{\text{processing}} * w_{\text{processing}} =$
(0.02, 0.13, 0.35) + (0.35, 0.68, 0.92) + (0.15, 0.38, 0.64) = Max (0.02, 0.35, 0.15), Max (0.13, 0.68, 0.38), Max (0.35, 0.92, 0.64) =

(0.35, 0.68, 0.92)

Similarly we get the ratings and weights of other sub characteristics under Efficiency Characteristic as shown in the Table 7.

Table 7 – Ratings of Different Sub Characteristics under Developer

Sub Characteristics	**Rating**	**Weight**
Time Behavior	(0.35,0.68,0.92)	(0.08,0.25,0.50)
Resource Utilization	(0.35,0.68,0.92)	(0.50,0.75,0.91)
Efficiency Compliance	(0.50,0.70,0.90)	(0.50,0.75,0.91)
Scalability	(0.50,0.70,0.90)	(0.08,0.25,0.50)

Now fuzzy weighted average can be taken to all these sub characteristics to obtain the fuzzy rating of the characteristic Efficiency. It is calculated in the same way to be **(0.25, 0.53, 0.82).**

Similarly we get the ratings of other characteristics under Developer's Perspective as shown in the Table 8.

Table 8 – Ratings of Different Characteristics under Developer

Characteristics	**Average Rating**	**Average Weight**
Functionality	(0.35,0.68,0.91)	(0.58,0.83,1.0)
Efficiency	(0.25,0.53,0.82)	(0.67,0.92,1.0)
Maintainability	(0.35,0.68,0.91)	(0.33,0.58,0.83)
Portability	(0.41,0.75,0.91)	(0.0,0.17,0.42)
Reliability	(0.21,0.45,0.70)	(0.0,0.08,0.33)
Usability	(0.17,0.41,0.70)	(0.0,0.17,0.42)

Now fuzzy weighted average can be taken to all these characteristics to obtain the fuzzy rating of the Developer's perspective as shown below.

Similarly we get the ratings and weights of other perspectives as shown in the Table 9.

Now fuzzy weighted average can be taken to all these perspectives to obtain the fuzzy rating of the overall quality as shown below.

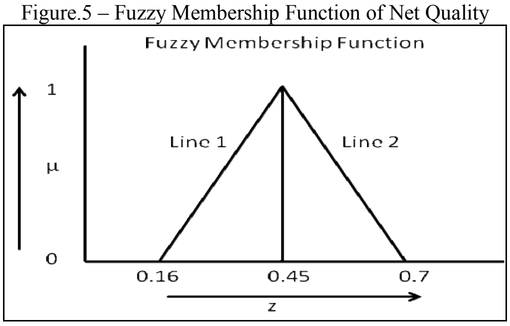
Table 9 – Fuzzy Rating of Net Quality Calculated

Net Quality	**Perspective**	**Net Rating**	**Net Weight**
(0.16,0.45, 0.7)	Developer's Perspective	(0.20,0.56,0.91)	(0.3,0.5,0.7)
	User's Perspective	(0.0,0.07,0.30)	(0.1,0.3,0.5)
	Manager's Perspective	(0.53,0.9,1.0)	(0.3,0.5,0.7)

Now the fuzzy rating of overall quality can be defuzzified using centroid formula to get the final crisp value of the software quality. This value lies in between 0 to 1. This is shown below.

(0.16,0.45,0.7) is the net quality obtained. It can be represented by a membership function as shown in Figure.5.

Figure.5 – Fuzzy Membership Function of Net Quality



Centroid Formula is used for Defuzzification.

Centroid Formula – $z^* = \frac{\int \mu(z) . z . dz}{\int \mu(z) . dz}$

Here z* is the defuzzified crisp value, z is the value on x – axis and μ(z) is the membership function.

Equation of line 1 => $\mu = 3.45z – 0.55$

Equation of line 2 => $\mu = 2.8 – 4z$

Therefore,

$$z^* = \frac{\int (3.45z – 0.55)\, z\, dz\ (z= 0.16 \text{to } 0.45) + \int (2.8 – 4z)\, z\, dz\ (z= 0.45 \text{to } 0.70)}{\int (3.45z – 0.55).\, dz\ (z= 0.16 \text{to } 0.45) + \int (2.8 – 4z).\, dz\ (z= 0.45 \text{to } 0.70)}$$

$$z^* = \frac{0.1175}{0.2705}$$

z* = 0.44 (Final Software quality)

Similarly by defuzzifying the fuzzy ratings of developer's, user's and project manager's perspectives we can get the corresponding perspective quality. The results are shown in Table10.

Table 10 – Final Software Quality Values

Perspective	Quality
Project Manager' Quality	**0.81**
Developer's Perspective Quality	**0.56**
User's Perspective	**0.12**
Total Software quality	**0.44**

V. Analysis

Some contrast has been made with the papers in the related area as described below.

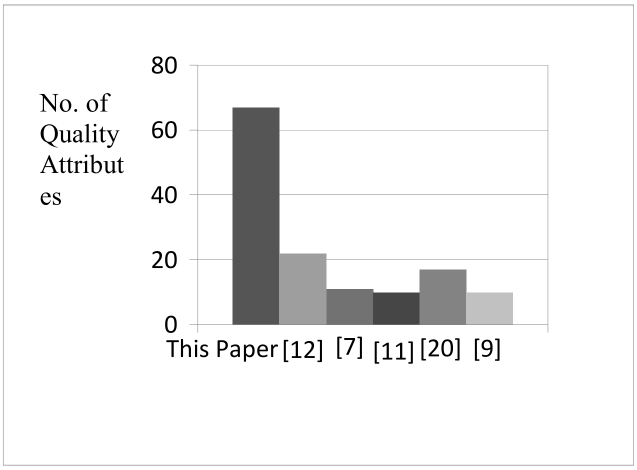
References [5], [14], [7], [15], [16] and [11] tried to estimate the software quality attributes. The criteria that have been chosen to quantify the metrics can be challengeable in these papers, because they have not dealt with the unpredictable nature of the software quality parameters. The current work includes fuzzy logic to deal with that unpredictable nature.

References [12], [13], [17] and [18] has considered fuzzy multi criteria approach in their papers. References [12] and [13] tried to rank the software products, where as [17] and [18] tried to find out the optimal solutions to their respective problems by ranking method. The current work uses Multi Criteria Approach to actually quantify the software quality rather than simply ranking different software products on the basis of their quality.

In [5], [12] and [13], the input to quantify the software quality has been taken from Manager, Developers and Users irrespective of the relevance of the attribute. The noticeable flaw in these papers is that the Developer gives judgment for User's Characteristics and vice versa. This could lead to imprecise and inaccurate results. In the current work, User's Perspective attributes are taken from 5 different users, Developer's Perspective attributes are taken from 3 different Developers and Project Manager's perspective attributes are taken from Project Manager only. So this leads to the calculation of software quality separately for User's, Developer's and Project Manager's Perspectives and removes the ambiguity while collecting the inputs.

This paper considers total 67 metrics for quantifying the software quality. Figure 6 contrasts the number of parameters considered by different researchers.

Figure.6 – Contrast of number of metrics used by various researchers



Many researchers have considered few characteristics of software quality in their evaluation. References [19], [20], [21], [10], [14]tried to quantify one or two characteristics. This paper considers all the characteristics of ISO/IEC 9126 Model.

Various researchers have attempted to quantify the software quality for specific environments like Object Oriented Environment [11], Aspect Oriented Environment, component based development systems [22] and [15], Commercial Off the Shelf Systems [23] and [24], etc. This paper presents a method to quantify software quality for generic applications.

Reference [25] used Fuzzy AHP to estimate the software quality. This approach is similar to that adopted in the current work. But they also did comparative analysis and ranking rather than computing exact software quality. The current work computes the exact software quality.

VI. Conclusion and Future Work

This paper attempts to precisely give an algorithm to estimate the Software Quality criteria using Fuzzy Multi Criteria approach.

Depending upon the value calculated for the software quality following inferences about the quality of the software have been inferred as shown in Table 11.

Table 11 – Inference on Software Quality

Overall Software Quality Calculated	Inference on Software Quality
More than 0.65	Very Good
Between 0.5 and 0.65	Good
Between 0.35 and 0.5	Average
Between 0.25 and 0.35	Poor
Less than 0.25	Very Poor

This work can be extended by considering some more factors in the model to quantify the software quality and also by using Fuzzy AHP, Chouquet Integral, Neural Fuzzy, etc.

REFERENCES

[1] J. A. McCall, P. K. Richards, and G. F. Walters, "*Factors in Software Quality*", 1977, Vol. I, II, and III, US Rome Air Development Center Reports - NTIS AD/A-049 014, NTIS AD/A-049 015 and NTIS AD/A-049 016, U. S. Department of Commerce.

[2] B. W. Boehm, J. R. Brown and M. L. Lipow, "*Quantitative Evaluation of Software Quality,*" *Proceedings of the 2nd International Conference on Software Engineering*, San Francisco, CA, USA, October, 1976, pp.592-605.

[3] R. G. Dromey, "*A model for software product quality*," *IEEE Transactions on Software Engineering*, Vol.21, No. 2, February, 1995, pp.146-162.

[4] *ISO/IEC 9126-1:2001, Software Engineering-Product Quality—Part 1: Quality Model,* Int'l Organization for Standardization, 2001, Available at www.iso.org.

[5] P. R. Srivastava and K. Kumar, "*An Approach towards Software Quality Assessment*," Communications in Computer and Information Systems Series (CCIS Springer Verlag), Vol. 31, No. 6, 2009, pp -345-346.

[6] O. Lamouchi, A.R. Cherif, and N. Lévy, "*A framework based measurements for evaluating an IS quality,*" Proceedings of the fifth on Asia-Pacific conference on conceptual modelling, Wollongong, NSW, Australia, January, 2008, pp.39-47.

[7] 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.

[8] S.A. Slaughter, D. E. Harter, & M. S. Krishnan, "*Evaluating the Cost of Software Quality,*" Communications of the ACM, Vol. 41, No. 8, August, 1998, pp. 67-73.

[9] M. Agarwal, & K. Chari, "*Software Effort, Quality, and Cycle Time: A Study of CMM Level 5 Projects*," IEEE Transactions on Software Engineering, Vol.33, No.3, March, 2007, pp. 145-156.

[10] O. Maryoly, M.A. Perez and T. Rojas, "*Construction of a Systemic Quality Model For Evaluating Software Product,*" Software Quality Journal, Vol. 11, No. 3, July, 2003, pp.219-242.

[11] Y. Kanellopoulos, P. Antonellis, D. Antoniou, C. Makris, E. Theodoridis, C. Tjortjis and N. Tsirakis, "*Code Quality Evaluation Methodology Using The Iso/Iec 9126 Standard,*" International Journal of Software Engineering & Applications (IJSEA), Vol.1, No.3, July, 2010, pp. 17 to 36.

[12] P. R. Srivastava , A. P. Singh, K.V. Vageesh, "*Assesment of Software Quality: A Fuzzy Multi – Criteria Approach*," Evolution of Cumputation and Optimization Algorithms in Software Engineering: Applications and Techniques, IGI Global USA, 2010, chapter – 11, pp.200-219.

[13] P. R. Srivastava, P. Jain, A. P. Singh, G. Raghurama, "*Software quality factor evaluation using Fuzzy multi-criteria approach*," Proceedings of the 4th Indian International Conference on Artificial Intelligence (IICAI 2009), Tumkur, Karnataka, India, December, 2009, pp. 1012-1029.

[14] M. Bertoa and A. Vallecillo, "*Usability metrics for software components*," Proceedings of Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE), Oslo, April, 2006, pp – 136 to 143.

[15] 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.

[16] V. Salvatore, A. Cucchiarelli and M. Panti, "*Computer Based Assessment Systems Evaluation via the ISO9126 Quality Model,*" Journal of Information Technology Education, Vol. 1, No. 3, 2002, pp. 157-175.

[17] S. A. Pratap, "*An Integrated Fuzzy Approach to Assess Water Resources' Potential in a watershed*", ICFAI Journal of Computational Fluid Mathematics, Vol. 1, No. 1, 2008, pp. 7–23.

[18] S.A. Pratap and A. K. Vidyarthi, "*Optimal allocation of landfill disposal site: A fuzzy multi criteria approach,*" Iranian Journal of Environmental Health Science & Engineering, Vol.5, No.1, 2008, pp. 25–34.

[19] I. Heitlager, T. Kuipers, J. Visser, "*A Practical Model for Measuring Maintainability – a preliminary report*," 6th International Conference on Quality of Information and Communications Technology (QUATIC), September, 2007, pp- 30-39.

[20] R. Fitzpatrick and C. Higgins, "*Usable Software and its Attributes: A synthesis of Software Quality European Community Law and Human-Computer Interaction*", Proceedings of the HCI'98 Conference, Springer, London, United Kingdom. 1998, pp – 1 to 19.

[21] J. R. Brown and M. Lipow, "*Testing for Software Reliability*", Proceedings of the international conference on Reliable software, Los Angeles, CA, USA, June, 1975, pp – 518 to 527.

[22] J.A. Borretzen, "*The Impact of Component Based Development on Software Quality Attributes,*" available at http://www.idi.ntnu.no/emner/dt8100/Essay2005/Boerretzen.pdf.

[23] M.R. Vigder, & A.W. Kark, "*Maintaining COTS-Based Systems: Start with the Design,*" Fifth International Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems, Orlando, Florida, USA, February, 2006, pp. 8-13.

[24] R. Adnan, and B. Matalkah, "*A New Software Quality Model for Evaluating COTS Components,*" Journal of Computer Science, Vol. 2, No. 4, 2006, pp. 373-381.

[25] C. W. Chang, C. R. Wu & H. L. Lin, "*Integrating fuzzy theory and hierarchy concepts to evaluate software quality*," Software Quality Journal, Vol. 16, No. 2, 2008, pp. 263–276.

[26] IEEE Standard Glossary of Software Engineering terminology, IEEE Std 610.12-1990.

[27] E.B. Swanson, and E. Dans, "*System Life Expectancy and the Maintenance Effort: Exploring Their Equilibrium*," *MIS Quarterly*, Vol. 24, 2000, pp. 277-297.

[28] T.J. Ross, "*Fuzzy Logic with Engineering Applications*", 2nd Ed, Wiley India Pvt. Ltd, New Delhi, India, 2004.

[29] Pressman, "*Software Engineering: A Practitioner's Approach*", 6th Ed., Tata McGraw-Hill, New Delhi, India, 2005.