ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com
International Journal of Advanced Research in Management, Architecture, Technology and Engineering
(IJARMATE) Vol. 3, Issue 2, February 2017

Designing Soft Computing framework for prediction of maintainability, reusability & reliability generic system and by use of multi objective decision making technique to estimate single value of quality

Richa Vats(Ph.D Scholar,CSE Deptt,SRM University)

Dr.Arvind Kumar(Assistant Professor, CSE Deptt,SRm University)

Abstract

Software quality is an important aspect for every software manufacturer. There are many models present that are used to describing the quality attributes of the software. For this paper, our base model is ISO-9126, and from this model we use three software quality metrics: maintainability, reusability, and reliability. We find these three quality attributes of software and from these three values, make a system which gives a single value of software quality. In the initial part of the paper, we use fuzzy simulation of matlab to find the individual values of quality, and then use a multi objective decision making technique to find the single value of software quality. For this entire system, we consider a generalized software system.

Introduction

Our objective while making any software is to produce good quality maintainable software in time and within budget. Here quality is very important. People understand quality, appreciate quality but may not be able to clearly express the same. Different people understand different meanings of quality, such as:

- Conformance to requirements
- Fit for the purpose
- Level of satisfaction

This is the layman definition for defining the software quality. Now the main questions is how do we maintain the software quality internally? We take a generalized software system for this paper,i.e. means the system we proposed in this paper is applicable to most software systems.

To find the software quality we consider an ISO/IEC 9126 model that defines software quality parameters, which varies from one project to another project. Because there are many others factors - often external - thatthat directly affect the software quality (Maintainability), maintainability is an important aspect for software quality. Other crucialfactors that ensure software quality are Reusability and Reliability - reusability allows us to reuse the code, or in any type of fault in software, reusability attribute helps us recover the code or troubleshooting when a fault occurs in the code, saving considerable time. By using minimum resources and gain maximum output from the software, software quality is decreased day by day , so main objective is quantified the software quality by using fuzzy logic approach with appropriate modification in it. We used fuzzy logic approach to measure the reliability, reusability and maintainability. Since this measurement is on a single entity of software quality, in last section of this thesis, through the use of AHP (Analytical Hierarchy Process),pairwise comparison matrix of 3 software quality attribute were made.The pairwise matrix is prepared on the same basis of fuzzy logic approach that we implemented in the earlier part of the paper. Using the pairwise comparison matrix for main goal which is three software quality attributes and make separate pairwise comparison matrix for every attribute and then using level of realization to find the effectiveness of each of the sub attribute of main attributes. Level of effectiveness is used to calculate the single value of software quality by using some assumption (in generalized way).

Literature Reviewed

Within the last ten years there have been many models proposed on the software quality model in which the authors take single attribute of software quality like reliability, maintainability and so on. But there is a gap in these proposed models. Unified software quality model is more important for software making companies to make good and quality assured software.

[4]Defines as reliability is non-functional requirement and further divided into three NFR's (Non-Functional Requirements) which are (Availability, Failure Probability and Recoverability). They use fuzzy logic approach to find the reliability of the system with the help of MATLAB simulation.

[5] Measuring reliability of an Aspect Oriented Software, in this paper they use 4 NFR (Maturity, Fault Tolerance, Reliability Compliance and Recoverability) by using Fuzzy Logic Approach taking assumption for making a library software concluded as library software which is developed using Aspect Oriented Approach is having high level fault tolerance, good recoverability and high reliability compliance standard .

[6]S. W. A. Rizvi and R. A. Khan proposed a maintainability estimation model for object oriented software in design phase.

In another study Hayes and Zhao [7], proposed a Maintainability model that categorizes software modules as 'easy to maintain' and 'not easy to maintain'. The model helps the developers to identify the modules those are not easy to maintain, before integrating them.

[8] Avadhesh Kumar, Rajesh Kumar and PS Grover proposed a model to an evaluation of maintainability of Aspect Oriented System.

To make system more reliable [9] Daesung Park, Sungwon Kang and Jihyun Lee proposed a modelDesign Phase Analysis of Software Qualities Using Aspect-Oriented Programming, which is quite useful if we see in terms of software quality.

[10] Avadhesh Kumar, Rajesh Kumar and PS Grover proposed A Fuzzy Logic Approach to Measure Complexity of Generic Aspect-Oriented Systems.

In all the models they take single attribute of quality to predict the quality of software which is incomplete itself because there are other factors which also affect the quality of software.

Software Quality

The study of software quality involves a planned and systematic set of activities to ensure the quality of software. Software quality is the degree conformance to clearly defined and documented or not clearly defined and documented requirements and expectation of end user. According to the IEEE 610.12 standard [11], software quality is a set of attributes of a software system and is defined as:

1. The degree to which a system, component, system or process meets specified requirements or customer needs and expectations.
2. The degree to which a system, component, or process meets customer or user needs or expectations.

The quality of software is measured in terms of its capability to fulfill the needs of the user and also its ability to achieve the developer's goal [5]. When user uses the product and finds the product fit for its purpose, he/she feels that product is of good quality. If the product is meeting user requirements,
the user will feel satisfied with the quality of the product. In simpler words, the users' views of quality must align with the product's ease of installation, operational efficiency, and convenience. In software, the quality is commonly recognized as "lack of bugs" in the program. If a software has too many bugs, then it is not able to fulfil the end user requirement.

In the present paper, the ISO/IEC 9126 Model has been considered as the base model [12].

The software engineering community was in search of a single model to standardize the quality factors since 1980. The advantage of such a universal model is quite obvious: it makes it easier to compare one product with another. The result was ISO 9126 in 1992 a hierarchical model with six major attributes contributing to quality. These attributes are:

- Functionality
- Reliability
- Usability
- Efficiency
- Maintainability
- Portability

These six factors can further divided into sub characteristics.

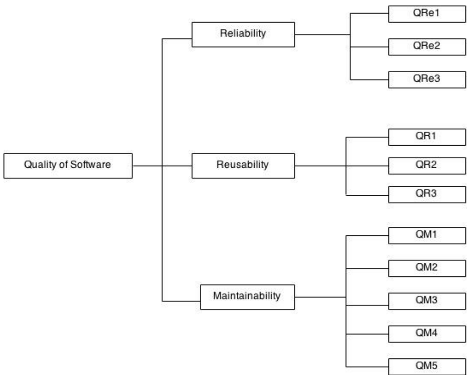


Figure: 1 Quality model for this paper

QRe1: MATURITY: Attributes of software that bear on the frequency of failure by fault in the software.

QRe2: Fault Tolerance: Attributes of software that bear on its ability to maintain a specified level of performance in case of software fault or infringement of its specified nature.

QRe3: Recoverability: Attributes of software that bear on the capability to reestablish its level of performance and recover the data directly affected in case of failure and effort related for it.

QR1: Understandability: Attributes of software that bear on the user's effort for recognizing the logical concept and its applicability.

QR2: Learnability: Attributes of software that bear on the user's effort for learning its application.

QR3: Operability: Attributes of software that bear on the user's effort for operating and operation control.

QM1: Analyzability: Attributes of software that bear on the effort needed for diagnosis of deficiencies or cause of failure, or of path to be identification of parts to be modified.

QM2: Changeability: Attributes of software that bear on the effort needed for modification, fault removal or for environment change.

QM3: Stability: Attributes of software that bear on the risk of unexpected effect of modifications.

QM4: Testability: Attributes of software that bear on the effort needed for validating the modified software.

QM5: Adaptability: Attributes of software that bear on the opportunity for its adaptation to different specified environment without applying other activities or means than provided for this purpose for the software considered.

These sub characteristics are used to define the fuzzy rules and later is used in multi objective decision making criteria in which we used AHP to find the single value of quality.

FUZZY Model

Fuzzy model is the best choice for managing vague, imprecise, doubtful, contradicting and diverging opinions. There are four modules - Rule Base, Fuzzification, Inference Engine, Defuzzification -that transform the crisp inputs into fuzzy values. These values are thenprocessed by inference engine in the fuzzy domain using the rule base created by domain expert. Finally the processed output is transformed from fuzzy domain to crisp domain by defuzzification module [4].

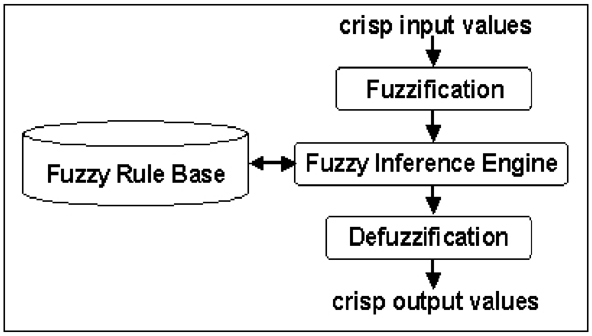


Figure 2: Block Diagram of Fuzzy Model

Working of Fuzzy Model

A fuzzy model performs its operations in the following steps [4]:

Fuzzification of Inputs:

Fuzzification is the first step in the Fuzzy inferencing process. This involves a domain transformation where crisp inputs are transformed into fuzzy inputs.

Apply Fuzzy Operators:

Fuzzy operators are applied to compute the degree of support for each rule for the generation of a single crisp value. This value will then be applied to the output functions. The input to a fuzzy operator is two or more membership values from fuzzified input variables. The output is the single truth-value. Two commonly used fuzzy operators are MIN MAX operators

Apply implication methods:

The shaping of the consequent based on the antecedent is termed as implications. The input for the implications processes is a single number given by the antecedent, and the output is a fuzzy set.

Implication is applied for each rule. Two commonly used implication methods for AND are MIN (i.e. truncation) and PRODUCT (i.e. scaling the height of a fuzzy set). Two commonly used implication methods for OR are MAX (maximum) and probabilistic OR (algebraic sum).

Aggregate all outputs:

Aggregation is a process by which fuzzy sets are combined in desirable ways to produce a single fuzzy set. It is obtained by combining all the fuzzy sets that represent the output of each rule into a single fuzzy set. Aggregation occurs only once for each output variable. For our model the input of the aggregation processes is the list of truncated output functions returned by the implication process for each rule. The output of the aggregation process is one fuzzy set for each output variable.

Defuzzification:

Defuzzification transforms the fuzzy values to crisp values. The input of the defuzzification process is a fuzzy set and the output is a single fuzzy number. Given a fuzzy set that encompasses a range of output values, one number needs to be returned thereby moving from a fuzzy set to crisp.

Fuzzy Model for Reliability

The first step in fuzzy inference step is to fuzzify inputs. The input parameters to fuzzy system are fuzzy sets. For this system use three parameters to define fuzzy system and defuzzification gives the reliability of the system. The input parameters are considered are to be of same weight. Input parameters are:

MATURITY= {Very Low, Low, Medium, High, Very High}

FAULT TOLERANCE= {Very Bad, Bad, Moderate, Good, Very Good}

RECOVERABILITY= {High, Average, Low}

Then these fuzzy sets are represented by a membership function. This can be best achieved by deciding numerical range for the fuzzy input/output variables.

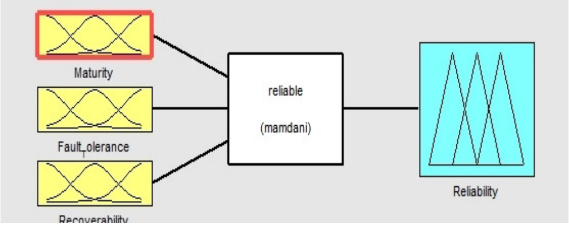


Figure 3 Fuzzy System

Linguistic Variable	Numerical Range
VERY LOW	0-2
LOW	2-4
MEDIUM	4-6
HIGH	6-8.5
VERY HIGH	8-10

Table 1: Fuzzy Variable range for maturity

Linguistic Variable	Numerical range
VERY BAD	0-4
BAD	4-6
MODERATE	6-8.4
GOOD	8.4-10.8
VERY GOOD	9.6-12

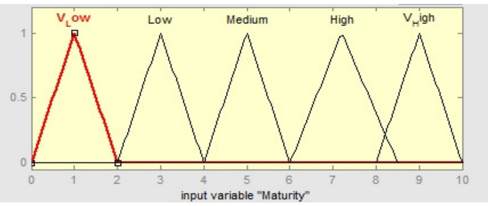


Figure 4: Fuzzification of input variable: Maturity

Linguistic variable	Numerical Range
LOW	-4 – 4
AVERAGE	3-8
HIGH	7-10

Table 3: Fuzzy variable range for Recoverability

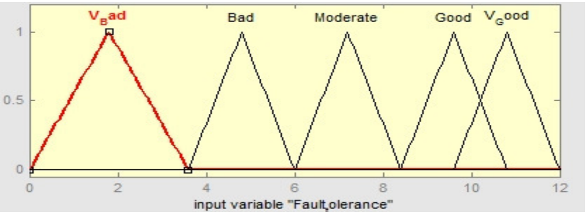


Figure 5:Fuzzificaion of input variable: Fault tolerance

Linguistic Variable	Numerical Range
VERY LOW	0-2
LOW	1.7-3.5
MEDIUM	3-6
HIGH	6-8.5
VERY HIGH	7.5-10

Table 4: Fuzzy variable range for Reliability

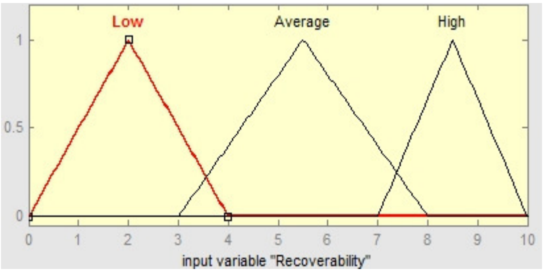
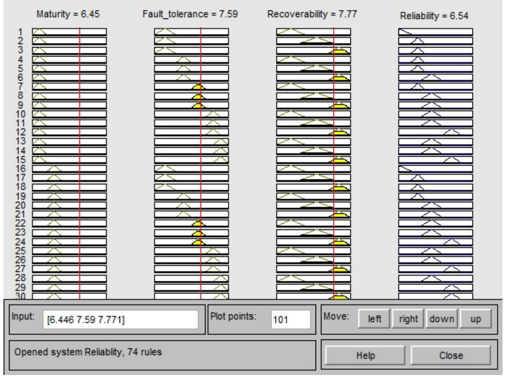


Figure 6: Fuzzificaion of input variable Recoverability

After the fuzzification of input, the operation is carried out in fuzzy domain. Using these three inputs, we define 73 rules in these three parameter to obtain single parameter reliability. These defined rules are the all possible 73 combinations to obtain the single parameter, termed as reliability.

In this model, mamdani fuzzy interference mechanism is used to observe output variable Reliability by Fuzzy Logic Toolbox in MATLAB. Rule viewer helps observe the output reliability level generated i.e. 7.5 corresponding to assumed set of variables as shown below.



Input: [6.446 7.59 7.771]

Plot points: 101

Opened system Reliablity, 74 rules

Figure 7: Rule Viewer for the Reliability model

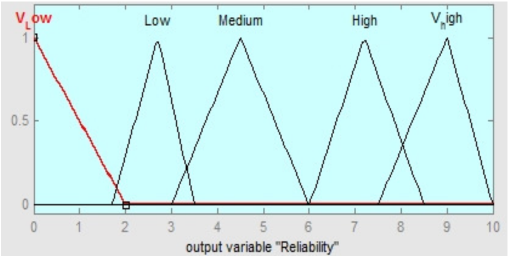


Figure 8: Fuzzification of output variable: Reliability

Defuzzification

The task of defuzzification is to find one single crisp value that summarizes the fuzzy sets that enter it from the inference block.It is a process of mapping from a space of fuzzy control actions defined over an output universe of discourse into a space of crisp (non-fuzzy) control actions. There are various names for this method such as center of- mass, center-of-area, or center-of-gravity method [13].

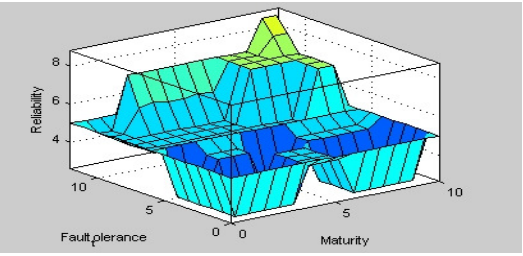
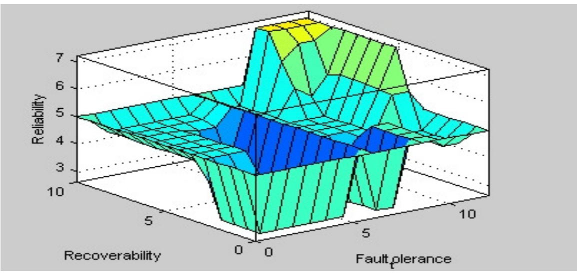


Fig 9: Surface view between Fault Tolerance & Maturity



Linguistic Variable	Numerical Range
VERY EASY	0-3
EASY	2-4
MODERATE	4-6
HARD	6-8
VERY HARD	8-10

Fig 10: Surface view between Fault Tolerance & recov.

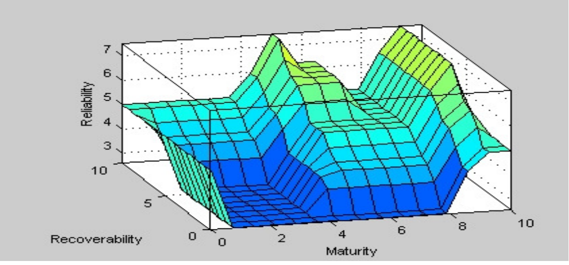


Figure 11: surface view Between Recoverability and Maturity

Result of Reliability

Suppose we have the following crisp values at input:

Maturity (6.45),Fault Tolerance(7.59) and Recoverability (7.77) then output for this system is observed as 6.45 which is High which means that system is reliable if maturity is medium , fault tolerance is either moderate and recoverability is medium system is reliable.

In similar manner, we can find for reusability and maintainability.

Fuzzy Model for Reusability

The first step in fuzzy inference step is to fuzzify inputs. The input parameters to fuzzy system are fuzzy sets. For this system use three parameters to define fuzzy system and defuzzification gives the reliability of the system. The input parameters are considered to be of same weight. Input parameters are:

UNDERSTANDABILITY= {Very Easy, Easy, Moderate, Hard, Very Hard}

USABLITY= {Low, Medium, High}

OPERABILITY = {Very Low, Low, Medium, High, Very High}

Then these fuzzy sets are represented by a membership function. This can be best achieved by deciding numerical range for the fuzzy input/output variables.

Table 5: Fuzzy variable range for Understandability

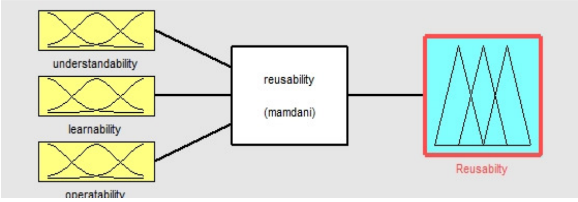


Figure 12: Fuzzy system for reusability

Linguistic variable	Numerical Range
LOW	0-3
MEDIUM	3-6
HIGH	6-10

Table 6: Fuzzy variable range for Learnability

Linguistic Variable	Numerical Range
VERY LOW	0-2
LOW	2-4
MEDIUM	4-6
HIGH	6-8
VERY HIGH	8-10

Table 7: Fuzzy variable range for Operability

Linguistic variable	Numerical Range
LOW	0-4
MEDIUM	4-6
HIGH	7-10

Similarly, the output variable is shown by using linguistic variable as shown below:

Table 8: Fuzzy variable range for output Reusability

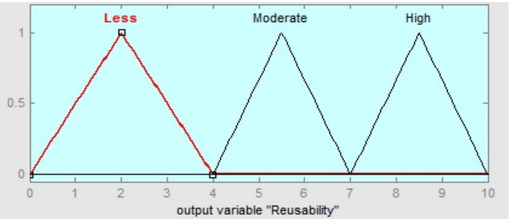


figure 13:Fuzzification of output variable: Reusability

After the fuzzification of input, the set of operation is carried on the fuzzy domains. According to the set of inputs rules were defined between them. Seventy five rules are defined between the 3 inputs of reusability system to obtain output of system which is reusability.

Defuzzification

The task of defuzzification is to find one single crisp value that summarize the fuzzy sets that enter it from the inference block.It is a process of mapping from a space of fuzzy control actions defined over an output universe of discourse into a space of crisp (non-fuzzy) control actions. There are various names for this method such as center of- mass, center-of-area, or center-of-gravity method [13].

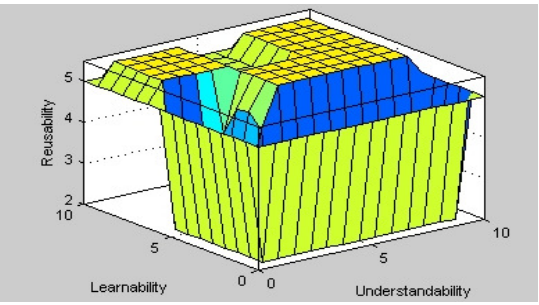


Figure 14: Surface view between learnability

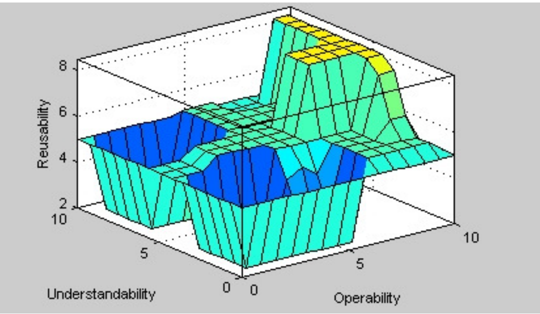


Figure 15: Surface view between operability and understandability

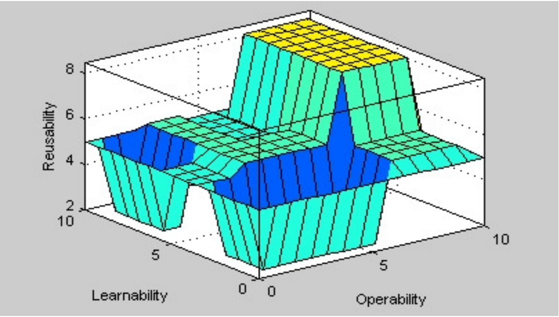


Figure 16: Surface view between operability and learnability

Result of Reusability

Suppose we have the following crisp values at input:

Understandability (6.57), Operability (5.96) and learnability (8.13) then output for this system is observed as 8.5 which is High which means that reusability of code is above average if understandability is high , learnability is medium

ISSN (ONLINE): 2454-9762
ISSN (PRINT): 2454-9762
Available online at www.ijarmate.com
International Journal of Advanced Research in Management, Architecture, Technology and Engineering
(IJARMATE) Vol. 3, Issue 2, February 2017

and operability is very high then we can predict that reusability of code have bright chance.

Fuzzy Model for Maintainability

The first step in fuzzy inference step is to fuzzify inputs. The input parameters to fuzzy system are fuzzy sets. For this system use three parameters to define fuzzy system and defuzzification gives the reliability of the system. The input parameters are considered are to be of same weight. Input parameters are:

ANALYZABILITY= {Low, Medium, High}

CHANGEABILITY= {Low, Medium, High}

STABILITY = {Low, Medium, High}

TESTABILITY= {Very Low, Low, Medium, High, Very High}

ADAPTABILITY= {Low, Average, High}

Then these fuzzy sets are represented by a membership function. This can be best achieved by deciding numerical range for the fuzzy input/output variables.

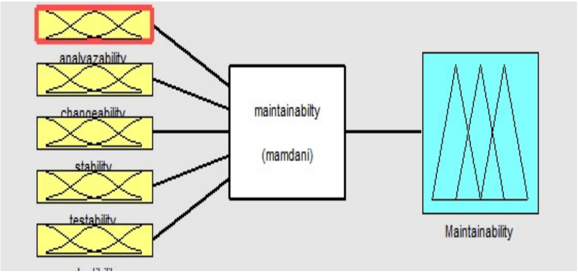


Figure 17: Fuzzy system for maintainability

Linguistic variable	Numerical Range
LOW	0-4
MEDIUM	3-7
HIGH	6-10

Table 9: Fuzzy variable range for input variable for Analyzability

Linguistic variable	Numerical Range
LOW	0-3
MEDIUM	3-7
HIGH	7-10

Table 10: Fuzzy variable range for input variable Changeability

Linguistic variable	Numerical Range
LOW	0-5
MEDIUM	4-8
HIGH	8-10

Table 11: Fuzzy variable range for input variable Stability

Linguistic Variable	Numerical Range
VERY LOW	0-2
LOW	2-4
MEDIUM	4-6
HIGH	6-8
VERY HIGH	7.5-10

Table 12:Fuzzy variable range for Testability

Linguistic variable	Numerical Range
LOW	0-4
MEDIUM	4-7
HIGH	7-10

Table 13: Fuzzy variable range for Adaptability

Similarly, the output variable is shown by using linguistic variable is shown below:

Linguistic Variable	Numerical Range
VERY LOW	0-3

LOW	2.5-4.5
MEDIUM	4-6
HIGH	6-8
VERY HIGH	7.5-10

Table 14: Fuzzy variable range for output variable Maintainability

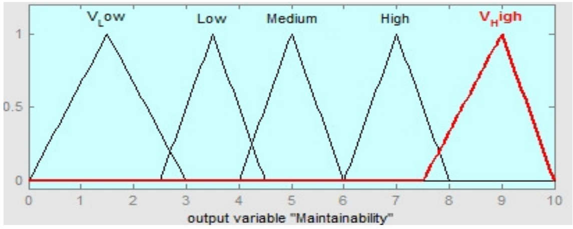


Figure 18: Fuzzification for output variable range for Maintainability

After the fuzzification of input variable, we have to set the no of operation in fuzzy domain to get the output maintainability. 395 rules were defined on the basis of assumption consider as system is genric, and how maintainability is persistent in the system related to these different subcharacterstics or fuzzy input variable.

Defuzzification

The task of defuzzification is to find one single crisp value that summarize the fuzzy sets that enter it from the inference block.It is a process of mapping from a space of fuzzy control actions defined over an output universe of discourse into a space of crisp (non-fuzzy) control actions. There are various names for this method such as center of- mass, center-of-area, or center-of-gravity method [13].

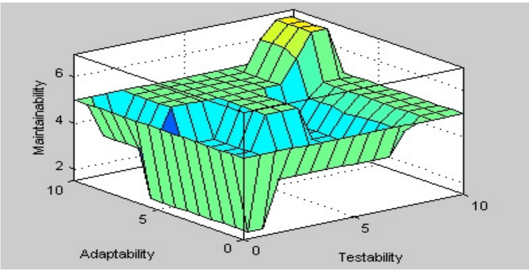


Figure 18: Surface view between Testability and Adaptability

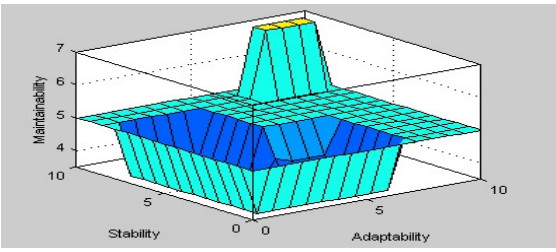


Figure 19: Surface view between Adaptability and Stability

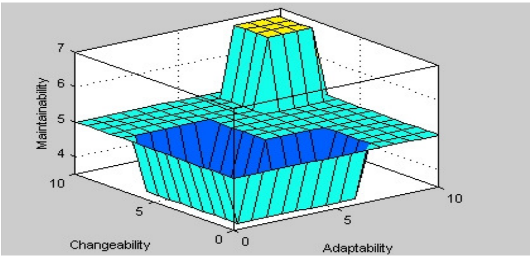


Figure 20: Surface view between Adaptability and Changeability

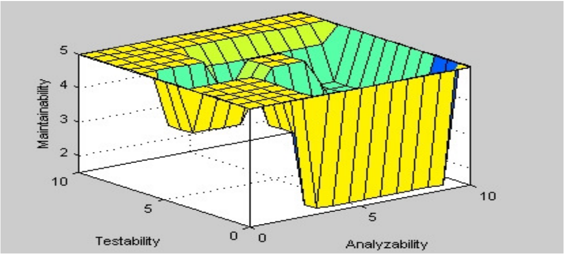


Figure 21: Surface view between Analyzability and Testability

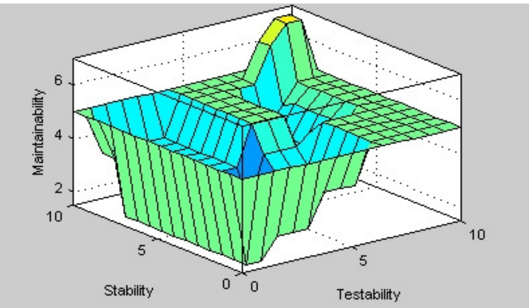


Figure 22: Surface view between Testability and Stability

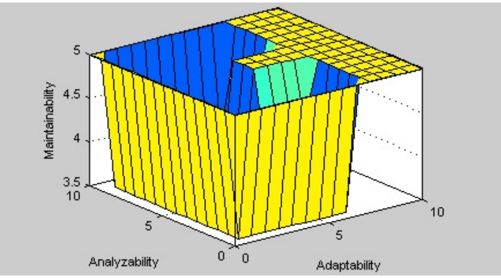


Figure 23: Surface view between Adaptability and Analyzability

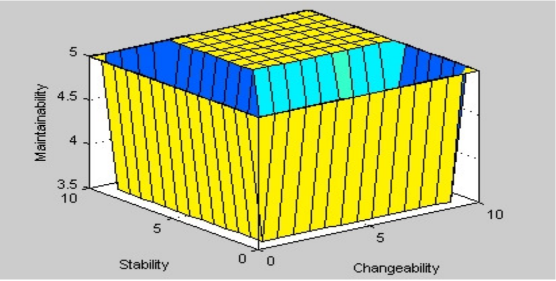


Figure 24: Surface view between Changeability and stability

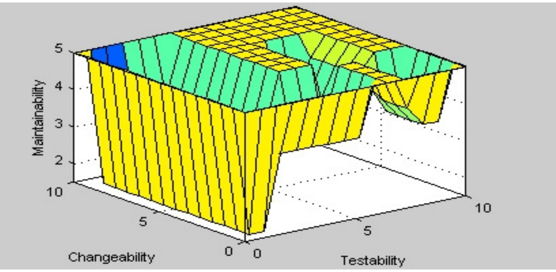


Figure 25: Surface view between Testability and Changeability

Result of Maintainability

Suppose we have the following crisp values at input:

Analyzability (6.09), Changeability (6.94), Stability (7.36), Testability (7.36), Adaptability (7.87) then output for this system is observed as 5.00 which is Medium.This means that maintainability in generic system is medium; in other words we can describe it as if Analyzability of system is either medium or high, Changeability is medium, Stability is medium, Testability is high and Adaptability is high then we conclude that system is subject to maintain.

Multi Objective Decision Making to find Single value of quality

In the above fuzzy sets, we get the individual value of software quality attributes, but we need a system in which multiple attributes/objectives are given as input and the objectives are as we defined in earlier sections of this paper. Now these objectives are fed into the system to find a single value of quality.

For this we use AHP (Analytical Hierarchy Process) which is multiple criteria decision making technique developed by T .Saaty. AHP is a multi-criteria decision making (MCDM) method that helpsdecision-maker facing a complex problem with multiple conflicting and subjective criteria (e.g. location or investment selection, projects ranking, etc). Several MCDM methods have been developed (e.g. ELECTRE, MacBeth, SMART, PROMETHEE, and UTA see (Barthélemy, 2003; Valerie Belton & Stewart, 2002)) and all are based on four steps: problem modelling, weights valuation, weights aggregation and sensitivity analysis [14].

The input can be obtained from actual measurement such as price, weight etc., or from subjective opinion such as satisfaction feelings and preference. AHP allow some small inconsistency in judgment because human is not always consistent. The ratio scales are derived from the principal Eigen vectors and the consistency index is derived from the principal Eigen value.

The hierarchical structure in AHP may vary according to the complexity of the problem and the number of elements and alternatives it includes. Another stage in the AHP technique is the formation of pairwise comparison matrixes according to the constituted hierarchical model. Pairwise comparison matrixes are square matrixes in n x n size.
The comparison of elements of matrixes with one another is made according to 1-9 scale of Saaty.

AHP pairwise comparison matrix is obtained by taking diagonal value 1, and other than diagonal elements, $a_{ij} = 1/a_{ij}$

Comparative Judgment	Intensity of Importance
a_i and a_j are equally important.	1
a_i is weakly important than a_j	3
a_i is more strongly important than a_j	5
a_i is demonstrably or very strongly more important than a_j	7
a_i is absolutely more important than a_j	9
Intermediate values between adjacent scale values	2,4,6,8

When many pairwise comparison are performed, some inconsistencies typically arise. According to the assumptions of AHP technique, pairwise comparisons must be consistent. Acceptable inconsistency levels suggested by Saaty (1994) are as follows: for 3x3 sizes the matrix is 0.05, for 4x4 size the matrix is 0.08, and for all other matrix it is 0.1.

If the inconsistency ratio of pairwise comparison matrixes is lower than the suggested value, the matrix can be considered consistent. Otherwise, pairwise comparisons are to be reviewed and renewed by the decision maker.Consistency ratio (CR) is calculated depending on the consistency index (CI) and the random index (RI) values.

Consistency index is obtained by using the eigen value $\lambda_{max.}$

$CI = \frac{\lambda\, max - n}{n-1}$,Where n is the size of matrix.

Consistency ratio is calculated as,$CR = \frac{CI}{RI}$, RI (Random Index) depend on the size of matrix.

RI	0.0	0.0	0.58	0.9	1.12	1.24	1.32	1.41	1.45
n	1	2	3	4	5	6	7	8	9

Table 15: Value of Random Index

From figure 1, Reliability, Reusability and Maintainability are the main goal for this paper, and the respective sub characteristics are the sub goal for the ultimate goal which is the quality.

Now, we first make pairwise matrix for main goal, ieis reliability (Re), Reusability(R), and Maintainability (M).Tomake pairwise comparisons matrix of each of the main goals(based on the fuzzy approach "earlier part of paper") and sub-main goals with a 1-9 scale, and to calculate the local weights and consistency ratio (CR). In AHP technique, evaluations start withpairwise comparisons. In this study, too, pairwise comparisons were made according to the 1–9 scale proposed by Saaty by taking into account the constituted AHP model.

Main Goal	**R**	**Re**	**M**	**Local Weight**
R	1	5	7	0.726
Re	1/5	1	1/5	0.076
M	1/7	5	1	0.198
CR	0.374			1.000

Local weightrepresents the relative importance of each main goal. The sum of local weight is 1. For the present study, Reusability, Reliability and Maintainability are respectively 0.518, 0.305 and 0.177. The consistency ratio was found is 0.274 which is acceptable against AHP assumption.

Local weight are calculated

R= (Product of first row) $^{1/n}$ = $(1*5*7)^{1/3}$=3.271

Re= (Product of first row) $^{1/n}$ = $(1/5*1*1/5)^{1/3}$=0.341

M== (Product of first row) $^{1/n}$ = $(1/7*5*1)^{1/3}$=0.893

Sum of 1.000+1.709+0.584=4.505

Local weight for, R =3.271/4.505=0.726, in similar way, Re=0.076, and M= 0.198

Now, normalized the matrix to get λ_{max}

R = (1*0.726+5*0.076+7*0.198) =3.433

Similarly, we get for Re =3.434 and M=3.439

And mean of these three values, gives λ_{max}=3.319

$$CI = \frac{3.435-3}{3-1} = 0.217,$$

$$CR = \frac{0.217}{0.58} = 0.374$$

With 0.374 Consistency ratio, this pairwise matrix is quite acceptable against our assumption on which pairwise matrix is formed.

Now we make pairwise comparison matrix for each sub goal which is defined earlier in software quality model.

Reliability	**QRe1**	**QRe2**	**QRe3**	**Local Weight**
QRe1	1	5	1/7	0.453
QRe2	1/5	1	1/7	0.042
QRe3	7	7	1	0.505

Reusability	**QR1**	**QR2**	**QR3**	**Local Weight**
QRe1	1	5	7	0.731
QRe2	1/5	1	3	0.189
QRe3	1/7	1/3	1	0.080

Reliability	**QM1**	**QM2**	**QM3**	**QM4**	**QM5**	**Local Weight**
QM1	1	5	5	1/7	1/3	0.160
QM2	1/5	1	5	1/3	1/7	0.085
QM3	1/5	1/5	1	1	1/3	0.064
QM4	7	3	1	1	1/5	0.205
QM5	3	7	3	5	1	0.486

The table shown above is based on the pairwise comparison matrix and each local weight shows the effectiveness of the each sub goal. For example in QRe1 local weight is 0.453, which shows that the effectiveness of that particular characteristics is 45.3%.

The more important part is to make a unified model to obtain the quality of software for which we used the level of effectiveness [15]. The levels of effectiveness of sub-goals were calculated by using the global weights and evaluation scale in Table 15. Table 15 was used for evaluating the current situation of each sub-goal. This scale was adapted from a study in the literature (Yüksel & Dağdeviren, 2006). The scale consists of six levels. Levels in the first column of the scale indicate the realization levels of each sub-goal, and levels in the second column of the scale demonstrate the numerical values

Value of realization	**Value of effectiveness**
Very Good	1.0
Good(G)	0.8
Moderate(M)	0.6
Negative(N)	0.4
Very Negative(VN)	0.2
Non Evaluation	0.0

Table 16: Evaluation table

Sub Goal of Quality	Global Weight	Level of realization of subgoal	Value of realization Level	Level of effectiveness by Subgoals
QR1	0.529	0.8	G	0.4232
QR2	0.136	0.6	M	0.0816
QR3	0.058	0.8	G	0.0464
QRe1	0.034	0.8	G	0.0272
QRe2	0.0003	1.0	VG	0.0003
QRe3	*0.038*	0.6	M	0.0228
QM1	0.031	1.0	VG	0.0310
QM2	0.016	0.8	G	0.0128
QM3	0.012	0.8	G	0.0009

QM4	0.040	0.6	M	0.0240
QM5	0.096	0.8	G	0.0768
		total		0.7470

Calculating the level of effectiveness of software quality model based on the sub characteristics of software model.

Global weight is calculated by using product of local weight of main goal to the respective sub goal matrix local weight. The level of effectiveness is calculated by using Table 16 to assign the effectiveness of each sub goal in generic software system and its product with numerical value of realization to get the last column, which is the level of effectiveness of subgoal.

Conclusion

In earlier part of this paper we use fuzzy logic approach to find the effectiveness of each software attributes, but this model is not in unified way to describe the whole software quality with multiple attributes. So in the final section of this paper, we designed a multi objective decision criteria technique using AHP to make a unified model for software quality. In this we define an individual pairwise matrix for main goal and pairwise matrix for each sub goal; we then use realization of each level and assign the values by expert team with taking different assumption makes unified system of software quality in which we used three software quality attributes (Reliability, Reusability, Maintainability), and output for this model comes out 0.7470 which is 74.70% i.e., system with these three software quality attributes in generalized software system, software quality comes out 74.70% which is quite acceptable. Inthe future, we will use all software quality attributes to make systems more reasonable and givemore efficient approaches to software developers to make software with good quality.

Reference

[1] G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C.Lopes, J. M. Loingtier, and J. Irwin, "Aspect-OrientedProgramming, " *Proceedings of the 11th EuropeanConference on Object-OrientedProgramming*, pp.220-242, LNCS, Vol.1241, Springer-Verlag, June 1997.

[2] G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C.Lopes, J. M. Loingtier, and J. Irwin, "An Overview of AspectJ," *Proc. 13th European Conference on Object-Oriented Programming*, pp.220-242, LNCS, Vol.1241,
Springer-Verlag, June 2000.

[3].hema kashyap "Process to identify the crosscutting concerns in changing requirements through Aspect-Oriented Software Engineering",IJCA,April 2014

[4] Ravinder Kumar,Kiran Khatter, Arvind Kalia " Measuring Software Reliability : A Fuzzy Model ",ACM SIGSOFT Software Engineering Notes , November 2011

[5] Reena Dadhieh, Bhavesh Kumar,"Measuring Reliability of an aspect oriented software using Fuzzy Logic Approach."

[6] S. W. A. Rizvi, R. A. Khan," Maintainability Estimation Model for Object-Oriented Software in Design Phase "

[7]J.H. Hayes and L Zhao, "Maintainability Prediction: a Regression Analysis of Measures of Evolving Systems," Proc. 21st IEEE International Conference on Software Maintenance, 26 - 29 Sept. 2005, pp. 601 -604, 2005.

[8]Avadhesh Kumar, Rajesh Kumar, PS Grover,"

An Evaluation of Maintainability of Aspect-Oriented Systems: a Practical Approach"

[9]Daesung Park, Sungwon Kang, Jihyun Lee,"Design Phase Analysis of Software Qualities Using Aspect-Oriented Programming"

[10]Avadhesh Kumar, Rajesh Kumar, PS Grover "A Fuzzy Logic Approach to Measure Complexity of Generic Aspect-Oriented Systems."

[11] IEEE Standard Glossary of Software Engineering Terminology, IEEE Std 610.12-1990.

[12] R. Laddad, "Aspect Oriented Programming will improve Quality", 2003,published by IEEE Computer Society 0740-745.

[13]Pradeep Kumar Singh, Om Prakash Sangwan, Amar Pal Singh, Amrendra Pratap,"A Framework for Assessing the Software Reusability using Fuzzy Logic Approach for Aspect Oriented Software", IJITCS, vol.7, no.2, pp.12-20, 2015. DOI: 10.5815/ijitcs.2015.02.02

[14]Alessio Ishizaka and Ashraf Labib,"Review of the main developments in the Analytic Hierarchy Process"

[15]Mehmet Yüksel, "Evaluating the Effectiveness of the Chemistry Education by Using the Analytic Hierarchy Process"

[16]. D. Shukla, S. Fell, and C. Sells. (2002) Aspect-Oriented Programming Enables Better CodeEncapsulation and Reuse. MSDN Magazine.Available:http://msdn.microsoft.com/enus/maGazine

[17]. O. Papapetrou and G. A. Papadopoulos,"Aspect Oriented Programming for acomponent-based real life application: a casestudy," presented at the Proceedings of the2004 ACM Symposium on AppliedComputing, Nicosia, Cyprus, 2004.