ISSN:0975-9646

Parul Gandhi / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 7 (2) , 2016, 647-654

Assessment of Component Generality using Fuzzy Approach to Optimize Software Development Cost

Dr. Parul Gandhi
Department of Computer Applications
Manav Rachna International University
Faridabad, India

Abstract-Role of reusability has seen a rapid demand for software. Reusability has become an essential characteristic of Component-Based Software (CBS). Component Based Software Development (CBSD) is becoming popular as object-oriented concept alone is not enough powerful to cope with the upcoming requirements of today's software. CBSE is a paradigm with the objective to construct and design software using a pre-defined set of software components explicitly created for reuse. To check to what extent a component can be reused in CBSD, it is required to measure the generality of components. In this paper three new metrics namely-Component Generality, Average Component Generality, Cost Benefit Percentage have been proposed. These metrics help to evaluate the generality of the component rather than a particular class and will be helpful to reduce the overall software development cost. We apply fuzzy logic to estimate the reusability and software development cost. The result obtained describe the analysis in terms of quality factors related to generality, reusability that aids significantly in assessing the component reusability and optimizing the overall software development cost.

Keywords: Component Based Software, Component Reusability, Component Generality, Cost Benefit Analysis, Fuzzy Inference System.

1. INTRODUCTION

In the past recent years, CBSE due to its plug and play principle is getting accepted among software industry and many other numerous industries. It has been studied in 1969 by McIlroy on reusable components [1]. The major benefit of CBSD is the use of same components in different applications having different functionality. Component Based Software Development (CBSD) [1] is a modern and efficient approach for developing complex and large software as it makes the use of well tested components under varieties of situations explicitly created for reuse. Component Based Software (CBS) is capable of reducing various overheads like cost, duplication of work, time of implementation, efforts and to enhance standard compliance and reliability of the system [2,3]. In a recent survey conducted on component based software development from 118 companies from around the world, it is found that around 53% of the organizations are using component based approach in its development [4].In today's competitive world with plenty of components available, the major task of any software developer is to qualify the characteristics of components in order to evaluate suitable reusable components from component pool. Gill [6] discussed the importance of component characterization for better reusability. To realize the reuse of components effectively it is necessary to measure the component reusability aspects. Measurement is required in software engineering to assess the quality of software products as well as improvement of their performance[5].This issue can only be resolved by making the use of metrics that intended to measure the component characteristics quantitatively.

The main objective of CBSD is to reduce the overall cost of the system software by making the use of already existing and well tested components which in turn also enhance the quality of the concerned product and will save time and efforts too. The cost of software reuse need to be justified with the benefits expected [7]. Cost Benefit Analysis is one of the important criteria to determine whether the selected component will yield significant returns on the investment made on it. The overhead costs associated with software reuse include [7]

- Training of personnel in design for reuse and design with reuse
- Creation and operation of a reuse repository
- Royalties and license costs of externally acquired components
- Maintenance and updating of reuse components
- Increased documentation to facilitate reuse
- Converting a component into a standard component for reuse

The paper is organized as follows. In Section 2 we briefly surveyed a substantial literature on component reusability. Section 3 explains the concepts of various existing metrics for component reusability. Section 4 discusses the unresolved issues. Section 5 describes the proposed metric. Section 6 evaluates the proposed metrics using fuzzy approach. Section 7 discusses results and Section 8 contains the conclusion and future work.

2. RELATED WORKS

In 1968 McIlroy anticipate Software reuse, at a NATO Software Engineering Conference, and predicted that mass-produced components would end the software crisis [9]. The objective was to use of existing artifacts several time to create new one. Many of the work by various authors focus on metrics of component reuse and reusability. In 1996 Frakes and Terry was first person to propose metric and models on software reuse. He suggested models based on cost benefits, assessing the maturity level, reuse library metrics [8].Washizaki et all proposed a metric suite to measure various non functional attribute of any component[12]. Kim in his research discussed the difficulties occur in component based software reuse and to discuss the preactive measure to meet before practicing software reuse[14]. Sandhu and Singh proposed an approach based on metric for identification of a reusable software module[11].Gill in his research also discussed the variety of issues that must be resolved for better component reusability [7]. He also discussed the importance of component characterization [6]. Sharma et. al discussed the reusability concepts for components based systems and to explore the reusability metrics to measure reusability directly or indirectly [15]. K. S. Jasmine, and R. Vasantha in his research proposed quality metric for component based software products [17]. Gupta and Kumar also conducted a study on how to retrieve reusable software component from component repository. He also proposed the techniques for storage and retrieval of software components [10].Margano, J. and T. Rhoads.in his research discussed software reuse economics with cost benefit analysis on a large scale[16]. Barns and Bollinger suggested analytical approaches for making good reuse investment based on cost benefit analysis [13].

3. Study Of Existing Component Based Metrics

There are few existing metrics which are used to measure the reusability of components. These metrics are:

Metric 1 Component Reusability (CR): This metric may be used at design phase in a component development process. CR is calculated by dividing sum of interface methods providing commonality functions in a domain to the sum of total interface methods. We define the CR as following formula

$$CR = \frac{\sum_{i=1}^{n}(count\ (CCMi))}{\sum_{i=1}^{n}(count\ (CIMj))}$$

Where:

Count (CCM): The count of each interface method for providing common functions among several applications in a domain

Count (CIM): The count of methods declared in interfaces provided by a component.

Metric 2 Reuse Leverage for Productivity (RL): This metric can be defined as Productivity with Reuse

$$RL = \frac{Productivity\ with\ Reuse}{Productivity\ without\ Reuse} * 100$$

Metric 3 Component Reuse Level (CRL): CRL is divided into CRLLOCs and CRLFunc. While CRLLOCsis measured by using Lines of Code (LOC). The CRLLOCs, expressed as a percentage, for a particular application is given by:

$$CRLLOCs = \frac{Reuse\ (C)}{Size(C)} * 100\%$$

where:

Reuse(C): The lines of code reused component in an application,

Size(C): The total lines of code delivered in the application.

The second sub-metric is CRLFunc, which is measured by dividing functionality that a component supports into required functionality in an application. This metric gives an indication of higher reusability if a large number of functions used in a component. The CRLFunc is expressed with:

CRLFunc(C) = Sum of supported functionality in a component

Metric 4 Generality of Class (GC): Generality of a class is the measure of its relative abstraction level. Higher the generality of a class more it is likely to be reused. GC can be computed as follows [21]:

$$GC = \frac{a}{al}$$

where:

a = Abstraction level of the class

al = Total number of possible abstraction levels

Metric 5 Reuse Benefit Metric (Rb): Devenbu et. al [22] considers the cost factor in assessing the benefits of reuse and proposed Reuse Benefit Metric Rb(S) of a system S, in terms of development cost. He proposed an indirect measure and to analytically evaluate the metric, he also proposed a set of properties of reuse benefit measure and evaluated the metric in terms of its compliance with these properties. The proposed metric still needs an empirical evaluation and validation to be used in actual software organizations.

4. Unresolved Issues

Generality is defined in IEEE Standard 610.12, as "the degree to which a system or component performs a broad range of functions." Generality increases the reusability of a component [18, 20]. Návrat and Filkorn also comment on the importance of generality with respect to reusability by stating that, things can only get reused if they are general and allow turning to specifics in a clear and straightforward manner [19].

Effective component reusability metric should:

(i) measure the generality of individual component
(ii) determine how collectively the different components affects a system.
(iii) define factors that influence each of the reusability of components
(iv) determine how these metrics collectively determine reusability of components.
(v) determine metrics for cost benefit percentage

No such generality measure could be found in literature. Therefore, in spite of the existence of other reusability measures, and various component based reusability metrics in literature, the issue of effectiveness of generality of individual components remains unresolved.

5. PROPOSED METRICS

This study is going to propose a reusability metrics suite that finds out the generality of the individual component by considering generic factors available in any component that directly affects the reusability and in turns affects the overall cost of software development.

- *Generality*

Metric 1 Component Generality (CG) : Generality of a component is the measure of generic attributes. This metric can be defined as the ratio of generic attribute (class, variable, method) available in an individual component to the total number of attributes of that component.

$$CG = \frac{No.\,of\ generic\ attributes}{Total\ no.\,of\ attributes}$$

Metric 2 Average Component Generality(ACG) : To measure the average generality of system above proposed metric is to be used and it can be defined as:

$$ACG = \sum_{i=1}^{n} \frac{CGi}{n}$$

Where:
CGi: it is the generality measure of ith component.
N: it is the total number of components used to build a system

- *Cost Benefit Analysis*

Software reuse is only relevant when it has positive economical impacts in organizations. In our previous research paper we proposed a Hierarchical Cost Estimation Model to provide a framework to estimate the effect of component reusability in the overall development process [23]. Model describes various costs involved in calculating the actual cost of developing the software. This paper makes the use of HCEM model [23] to propose a new metric that will help the developer to evaluate the Cost Benefit Percentage of developing the system by making the use of components available in the component repository.

Metric 3 Cost Benefit Percentage (CB%) **:**This metric based on the components available in the component repository as well as the components needs to be developed from scratch. It can be defined as:

$$CBP = \frac{\sum_{i=1}^{n} TC - (\sum_{i=1}^{m} Cwm + \sum_{i=1}^{n-j} Cm + \sum_{i=1}^{s} Cs)}{\sum_{i=1}^{n} TC} * 100$$

Where:
n:Total number of components required to develop a software.
m: Number of components available in repository and can be used without modification
s:Number of components developed from scratch
j=m+s: Number of components available in repository and require some modification
Cwm: Cost required to adapt the available component available in repository without any modification
Cs:: Cost required to develop the component from scratch
Cm: Cost required to modify the component available in repository.
TC: Total cost to develop the software without using components

6. FUZZY INFERENCE SYSTEM

The solution for the unresolved issues mentioned above is to use fuzzy logic linguistic variables. In this section of paper, the fuzzy logic approach has been discussed for the assessment of software reusability. Fuzzy Inference System comprises of four stages.

- *Fuzzification:* During this stage fuzzification module transforms crisp input(s) into fuzzy inputs by
  membership function created using membership function editor.
- *Rule Base:* At this stage the domain expert provides the fuzzy if-then rules.
- *Inference System:* During this stage the input values are processed by interface engine based onrules supplied by the domain expert(s).
- *Defuzzification:* At this stage, defuzzification module transformed the output from fuzzy domain to crisp domain.

6.1 Proposed Fuzzy Model for Component Reusability

Proposed model makes the use of all the GUI tools of Fuzzy Inference System to map a given input to an ouput using fuzzy based approach.

6.1.1 FIS Editor:

Mamdani fuzzy system has been used to evaluate software reusability. Our fuzzy model accounts the effect of Component Generality and Average Component Generality on the overall reusability. Block diagram of the proposed model is shown in Figure 1.

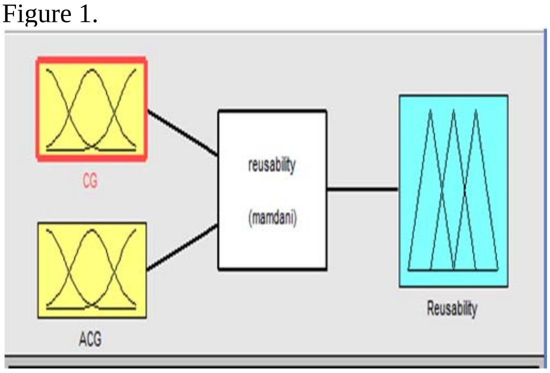

Fig. 1: Fuzzy Model for Reusability

- **Dependent and Independent variable:**

The ouput variable, reusability is the dependent variable which comes after fuzzification method. The two proposed metrics Component Generality (CG) and Average Component Generality (ACG) act as independent variables. For simplification measurement values of these input metrics taken in the range of 0 to 1. The values of these metrics will be used as an input to the fuzzy inference system (FIS) for measuring the reusability of the system. Table 1 summarizes the system information in terms of input and output variables used in the system.

Table 1 System Information in MATLAB for Reusability

Name	**"Reusability"**
Type	**Mandani**
No. of Inputs	**2**
No. of Outputs	**1**

6.1.2 Membership function for input and output parameter

In this work, two input and one output variable Component Generality (CG), Average Component Generality (ACG) and Reusability respectively has been taken in the scale of 0 to 1 and member functions as Very Low (VL), Low (L), Medium (M), High (H) and Very High (VH).Range of membership functions Very Low (VL), Low (L), Medium (M), High (H) and Very High (VH) are0.0-0.2, 0.2-0.4, 0.4-0.60.6-0.8, 0.8-1.0 respectively. Figures 2, 3 and 4 show fuzzy sets taken for each input and output variables in Mamdani model.

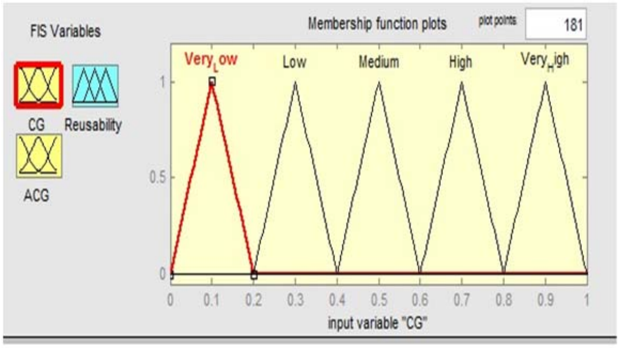

Fig. 2: Input variable "CG"

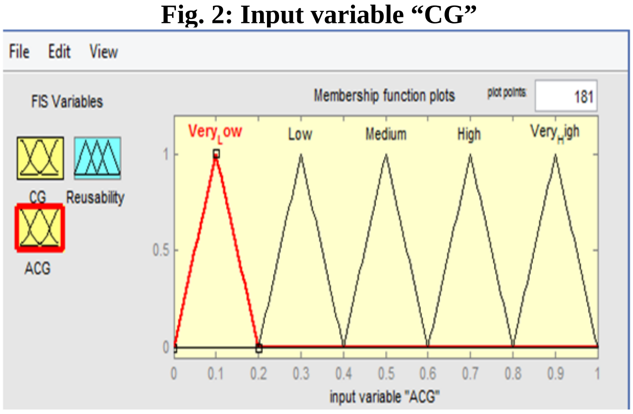

Fig. 3: Input variable "ACG"

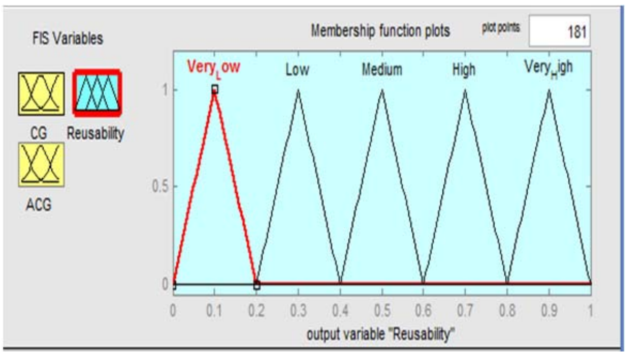

Fig. 4: Output variable "Reusability"

6.1.3 Fuzzy rules

In this study our main objective is to measure reusability. As generality of component has a great impact on reusability so we have taken two factors Component Generality (CG), Average Component Generality (ACG) contributing in reusability. In order to find effect of these factors we have used fuzzy logic and have designed various fuzzy rules. Figure 5 shows mamdani method for defining fuzzy rules.

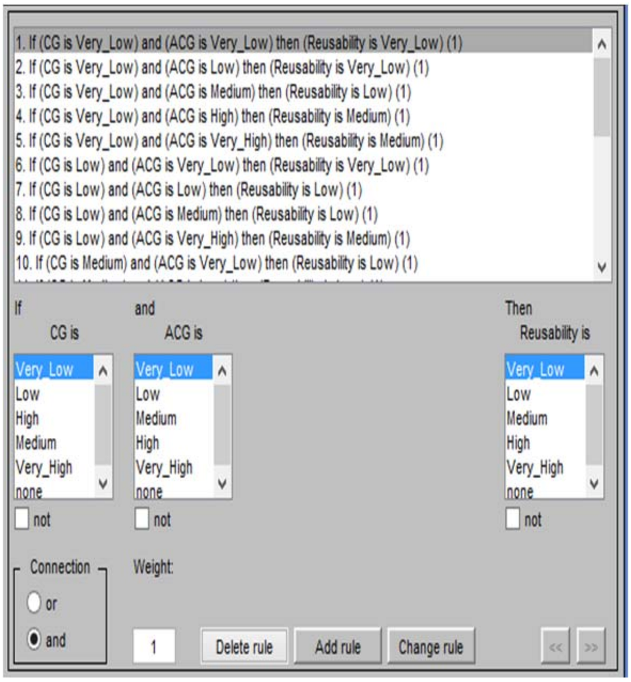

Fig. 5 Rule Editor

Some sample rules are shown in table 2. First column of the table represent rule number, second column is for input variables CG, ACG and third column is for output variable Reusability. These rules are also called knowledge base as these are totally designed making use of experience and expertise of the concern area.

Table 2. Some sample Rules for Reusability

Rules	**Inputs**		**Output(Reusability)**
	CG	**ACG**	
1	Low	Low	Low
2	Low	High	Medium
3	Medium	Low	Low
4	Medium	High	Medium
5	High	Very High	Very High

6.1.4 Rule Viewer

Figure 6 shows the rule viewer of proposed fuzzy model. There are three columns, the first two columns represents the input variable and third column represents the membership function for output variable of the proposed FIS. The bold vertical lines represent the defuzzified output.

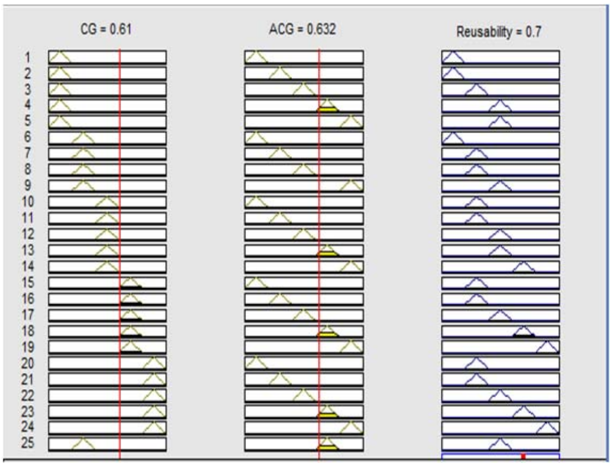

Fig. 6: Rule Viewer

6.1.5 Surface Viewer

Figure 7 show the three dimensional surface view for proposed fuzzy model. It shows the output surface for the set of two metrics (CG and ACG) for reusability evaluation.

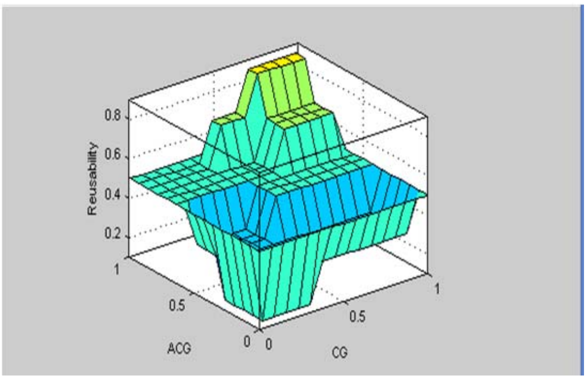

Fig. 7 Surface Viewer to Evaluate Reusability

6.2 Proposed Fuzzy Model for Software Cost Estimation

6.2

6.2.1 FIS Editor:

Figure 8 shows the block diagram of the proposed model. In this model our objective is to evaluate software development cost. This fuzzy model accounts the effect of Component Generality, Reusability Cost of component and Component Development Cost on the overall software cost development.

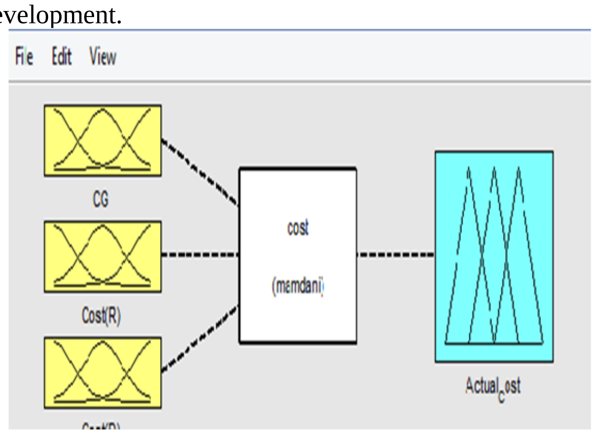

Fig. 8: Fuzzy Model for Cost Estimation

- **Dependent and Independent variable:**

Output variable Actual Cost is the dependent variable which comes after fuzzification method. Component Generality (CG), Reusability Cost(Cost(R) and Development Cost (C(D)) act as independent variables. Simplification measurement values of these input parameters taken in the range of 0 to 1. The values of these metrics will be used as an input to the Fuzzy Inference System (FIS) for measuring the actual cost of developing the software of the system. Table 3 summarizes the system information in terms of input and output variables used in the system.

Table 3 System Information in MATLAB for Cost Estimation

Name	**"Actual Cost"**
Type	**Mandani**
No. of Inputs	**3**
No. of Outputs	**1**

6.2.2 Membership function for input and output parameter

In this work, three input and one output variable Component Generality (CG), Reusability Cost(Cost(R) and Development Cost (C(D)), Actual cost respectively has been taken in the scale of 0 to 1 and member functions as Low (L), Medium (M), High (H) .Range of membership functions Very Low (VL), Low (L), Medium (M), High (H) are 0.0-0.3, 0.3-0.6, 0.6-1.0 respectively. Figures 9, 10, 11 and 12 show fuzzy sets taken for each input and output variables in Mamdani model.

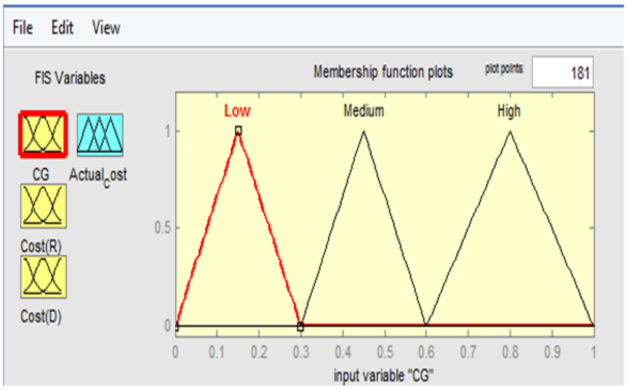

Fig. 9: Input variable "CG"

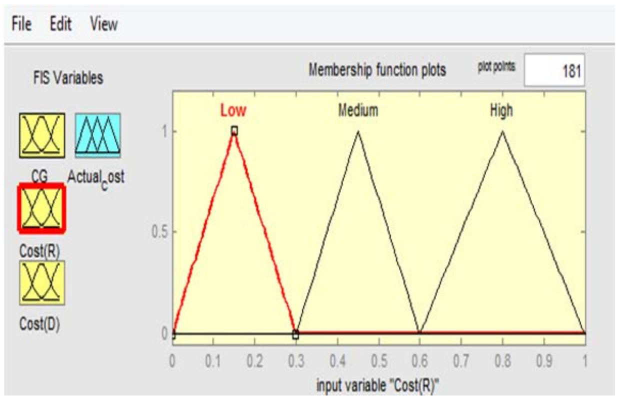

Fig. 10: Input variable "Cost(R)"

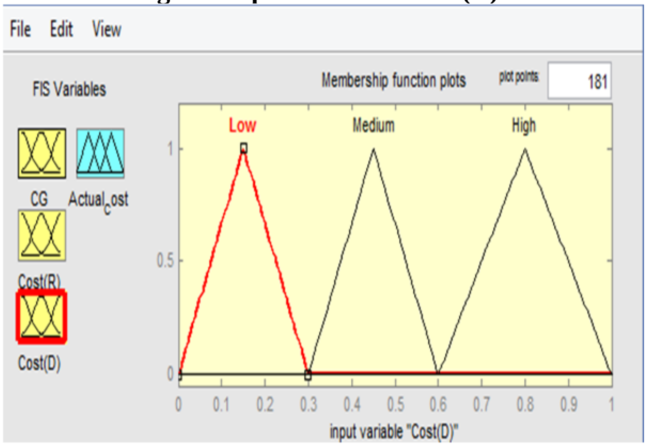

Fig. 11: Input variable "Cost(D)"

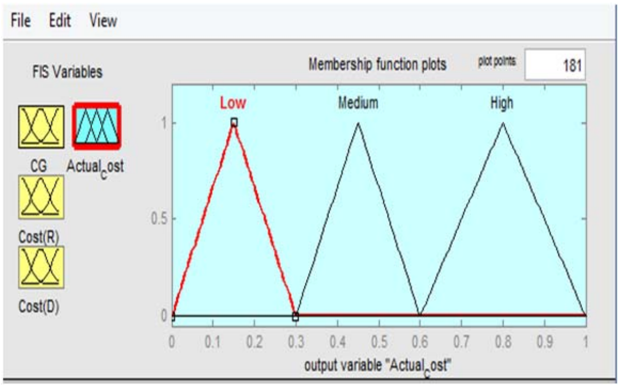

Fig. 12: Output variable "Actual Cost"

6.2.3 Fuzzy rules

In this study our main objective is to measure actual cost of software. We have taken three factors Component Generality (CG), Reusability Cost(Cost(R) and Development Cost (C(D)) contributing in cost of the software. In order to find effect of these factors we have used fuzzy logic and have designed various fuzzy rules. Figure 13 shows mamdani method for defining fuzzy rules.

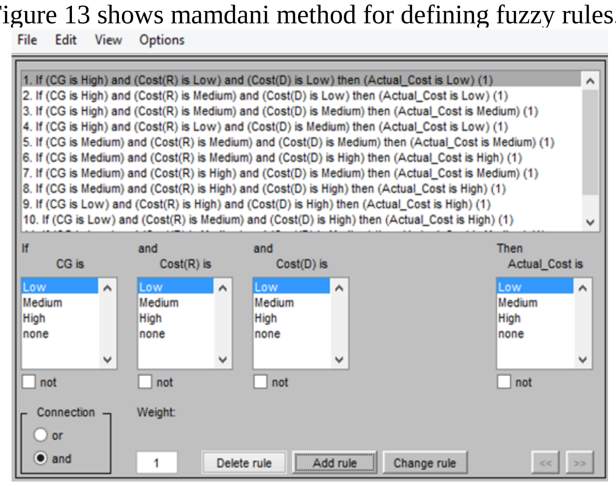

Fig. 13: Rule Editor

Some sample rules are shown in table 4. First column of the table represent rule number, second column is for input variables CG, Reusability Cost(Cost(R) and Development Cost (C(D)) and third column is for output variable Actual Cost.

Table 4. Some sample Rules for Actual Cost

Rules	Inputs			Output(Actual Cost)
	CG	Cost(R)	Cost(D)	
1	Low	High	High	High
2	Low	Medium	High	High
3	High	Low	Low	Low
4	High	Low	Medium	Low
5	High	Medium	Medium	Medium

6.2.4 Rule Viewer

Figure 14 shows the rule viewer of proposed fuzzy model for cost estimation. There are four columns, the first three columns represents the input variable and fourth column represents the membership function for output variable of the proposed FIS. The bold vertical lines represent the defuzzified output.

Fig. 14: Rule Viewer

6.2.5 Surface Viewer

Surface viewer shows the three dimensional surface view for proposed fuzzy model by considering two input parameters at one instance. Figure 15 shows the output surface for the set of two parameters (CG and Cost(R)) and Figure 16 shows the output surface for the set of two parameters (Cost(R) and Cost(D))

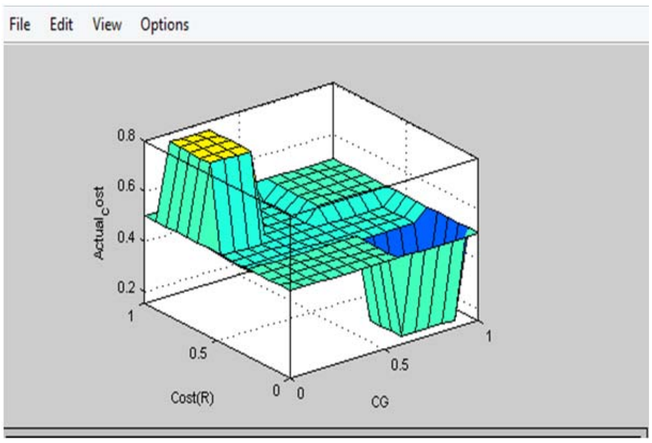

Fig. 15: Surface Viewer for CG and Cost(R)

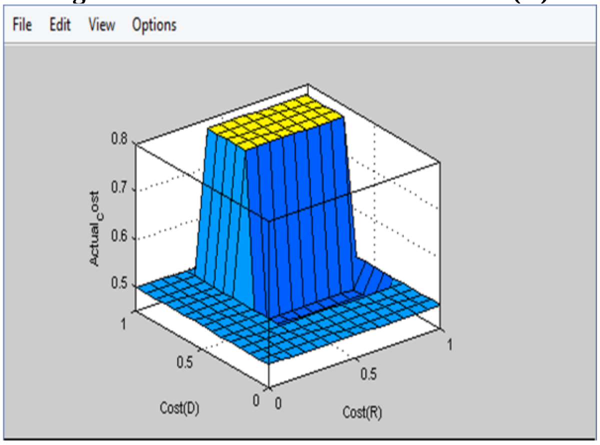

Fig. 16: Surface Viewer for Cost(R) and Cost(D)

7. Experimental Results

Experimental results based on firing some rules on the proposed fuzzy based reusability system are reported in Table 5. As an example, if CG =0.76 (H), ACG =0.62 (H) are input values then Reusability value is resulting as 0.7, which is high for output.

Table 5. Experimental Results for Reusability

Rules	Inputs		Output (Reusability)	Results
	CG	ACG		
1	0.15	0.14	0.1	Worst Appropriate
2	0.34	0.52	0.3	Not Appropriate
3	0.55	0.66	0.5	Somewhat Appropriate
4	0.76	0.62	0.7	Appropriate
5	0.76	0.83	0.9	Highly Appropriate

Experimental results based on firing some rules on the proposed fuzzy based cost estimation system are reported in Table 6. As an example, if CG =0.813 (H), Cost(R) =0.235 (L), Cost(D)=0.464 (L) are input values then Actual Cost value is resulting as 0.15, which is low for output.

Table 6. Experimental Results for Cost Estimation

Rules	Inputs			Output (Actual Cost)	Results
	CG	Cost(R)	Cost(D)		
1	0.283	0.536	0.813	0.8	Not Appropriate
2	0.223	0.488	0.464	0.45	Somewhat Appropriate
3	0.813	0.235	0.464	0.15	Highly Appropriate

7.1 Statistical Measure on Fuzzy output

We also apply statistical measure on fuzzy outputs generated by proposed fuzzy model for reusability and cost estimation.

Table 7. Relationship between CG and Reusability

S.No.	CG	Reusability
1	0.15	0.1
2	0.34	0.3
3	0.55	0.5
4	0.76	0.7
5	0.76	0.9

Correlation between Component Generality and Reusability results in 0.9713 which shows a high degree of positive correlation between these attriblues.

Table 8. Relationship between CG and Actual Cost

S.no.	CG	Actual Cost
1	0.283	0.8
2	0.223	0.45
3	0.813	0.15

Correlation between Component Generality and Cost Estimation results in -0.7897 which shows a considerable degree of negative correlation between these attributes.

The results obtained after applying statistical measure on fuzzy output shows that Component Generality (CG) is directly proportional to Reusability as well as inversely proportional to Actual Cost of developing the software. We can conclude from the above results that if component have high degree of reusability it will reduce the cost of developing the software to great extent.

8. Conclusions And Future Work

In this paper fuzzy logic approach has been used to access the reusability as well as software development cost. The motivation has been difficulties faced to find generality of the component to reuse it in order to minimize the software development cost. Using common generic framework defined for component based development new metrics have been defined. These metrics are used for measuring component generality, average component generality, cost benefit percentage of component based system. We have also proposed two fuzzy model, one for predicting the reusability and another for the actual cost of developing the software. Reusability and cost estimation factors further may be used as an indicator to other software quality attributes such as maintainability, complexity, adaptability and understandability. We also apply the statistical measure to show the correlation between reusability and cost estimation. The experimental results shows that the more generic the component will be the more will be its reusability which in turn further reduce the actual cost of software development to great extent. So, it is concluded that proposed model based on fuzzy will helps the software developer's to select the best quality of software in terms of reusability and cost estimation. Apart from fuzzy inference system, other soft computing techniques such as Artificial Neural Network (ANN), Neuro Fuzzy System can also be taken into consideration for the assessment of software reusability.

References

[1] Mojica, I. J., Adams, B., Nagappan, M., Dienst, S., Berger, T., and Hassan, A. E. "A Large-Scale Empirical Study on Software Reuse in Mobile Apps". *IEEE Software*, vol. 31, no. 2, pp 78-86, 2014.

[2] Singh, A. P. and Tomar, P., "A new model for Reliability Estimation of Component-Based Software", in *Proc. IEEE 3rd International Advance Computing Conference* (IACC), pp. 1431-1436, Feb. 2013.

[3] Heinemann, L., Deissenboeck, F., Gleirscher, M., Hummel, B., and Irlbeck, M. "On the extent and nature of software reuse in open

source Java projects". *In Top productivity through software reuse Springer Berlin Heidelberg*. pp. 207-222, 2011.

[4] Paul Allen, "CBD Survey: The State of the Practice", a white paper by Cutter Consortium. Web: http://www.cutter.com/research/2002/edge020305.html

[5] Chawla, S., & Nath, R. (2013). *Evaluating Inheritance and Coupling Metrics*. 4 (7).

[6] Nasib Singh Gill, Importance of Software Component Characterization For Better Software Reusability", ACM SIGSOFT SEN Vol. 31 No. 1.

[7] Gill N.S., "Reusability Issues in Component-Based Development", ACM SIGSOFT Software Engineering Notes, 28(4), ISSN: 0163-5948, pp. 1-4, 2003.

[8] Frakes, William and Terry, Carol "Software Reuse: Metrics and Models"; ACM Computing Surveys, 28, 2 (1996), pp. 415-435.

[9] Imeri F.; Antovski L., "An Analytical View on the Software Reuse", ICT Innovations 2012, Web Proceedings of the 4th ICT – ACT Conference, Ohrid - Macedonia, ISSN: 1857 – 7288, pp. 213 – 222, 2012.

[10] Gupta S., Kumar A., "Reusable Software Component Retrieval System", International Journal of Application or Innovation in Engineering and Management, 2[1], pp. 187 – 194, 2013.

[11] Sandhu, P. S., and Singh, H., "Automatic reusability appraisal of software components using neuro-fuzzy approach", *International Journal Of Information Technology*, vol. 3, no. 3, pp. 209-214, 2006.

[12] Washizaki, H., Yamamoto, H., and Fukazawa, Y., "A metrics suite for measuring reusability of software components", in *Proceedings of IEEE Ninth International Software Metrics Symposium*, pp. 211-223, 2003.

[13] Barns, B. H., and Bollinger, T. B., "Making reuse cost-effective", IEEE *Trans. Software*., vol. 8, issue 1, pp. 13-24, 1991.

[14] Kim W., "On Issues with Component – Based Software Reuse", Journal of Object Technology, 4[7], pp. 45 – 50, 2005.

[15] Sharma A., Kumar R., Grover P., "Managing Component – Based Systems with Reusable Components", International Journal of Computer Science and Security, 1[2], pp. 60 – 65, 2007.

[16] Margano, J. and T. Rhoads. Software Reuse Economics: CostBenefit Analysis on a Large Scale Ada Project. In Proceedings, International Conference on Software Engineering, Melbourne, Australia, 11-15 May 1992: 338 -348.

[17] K. S. Jasmine, and R. Vasantha,‘ DRE A Quality Metric for Component based Software Products, World Academy of Science, Engineering and Technology 34 2007

[18] Gill, N. S., & Sikka, S. (2011). Inheritance Hierarchy Based Reuse & Reusability Metrics in OOSD. *International Journal on Computer Science and Engineering (IJCSE)*, 3 (6), 2300-2309.

[19] Návrat, P., & Filkorn, R. (2005). A Note on the Role of Abstraction and Generality in Software Development. *Journal of Computer Science* , 1 (1), 98-102.

[20] Sommerville, I. (2011). *Software Engineering* (9th Edition ed.). Boston: Pearson Education.

[21] Nasib S. Gill et al. / International Journal on Computer Science and Engineering (IJCSE) Vol. 3 No. 6 June 2011 pp 2300-2309

[22] P Devenbu , S Karstu, W Melo, W Thomas, "Analytical evaluation of Software Reuse Metrics", Proceedings of the 18th International Conference on Software Engineering (ICSE'96), IEEE, pp 189-199

[23] Parul Gandhi and Pradeep Kumar Bhatia, "Evaluating Impact of Component Reusability with New Hierarchical Cost Estimation Model", International Journal of Computer Engineering & Technology (IJCET), Volume 3, Issue 2, 2012, pp. 526 - 532, ISSN Print: 0976 – 6367, ISSN Online: 0976 – 6375.