https://dx.doi.org/10.24001/ijaems.3.9.10 *ISSN: 2454-1311*

Analyzing the Effects of the Different Defuzzification Methods in the Evaluation of Java components' Customizability for Reusability

Ajayi Olusola Olajide, Aderele Tolulope Busayo, Elemese Tolulope Olawale

Department of Computer Science, Faculty of Science, Adekunle Ajasin University, Akungba-Akoko, Ondo State, Nigeria

Abstract— *Customizability is one of the key factors that determine reusability of components in component-based software engineering. In tandem with measuring component reusability based on predicted customizability of the components selected, this study also analyze the effects of the application of different defuzzification methods in the process of achieving this evaluation. We deployed five (5) defuzzification methods: Centroid, Bisector, Largest of Maximum, Smallest of Maximum, and Mean of Maximum. The results of the experimentation carried out using MATLAB as tool and four (4) components extracted from a third party organization, proved that the applied defuzzification methods yield different reusability values, attesting to the fact that, the type of defuzzification method applied in the evaluation of component reusability has effects on its result.*

Keywords—defuzzification, reusability, customizability, Java, fuzzy logic.

I. INTRODUCTION

The demand for new software applications is currently increasing at the exponential rate, as is the cost to develop them. The numbers of qualified and experienced professionals required for this extra work are not increasing commensurably (Smith, et al 1998), Software professionals have recognized reuse as a powerful means of potentially overcoming the above said software crisis (Basili, 1989; Boehm et al, 1989) and it promises significant improvements in software productivity and quality (Lim, 1994;Mili et al, 1995). Software reuse is the improvement efforts of the productivity of the software because reuse can result in higher quality software at a lower cost and delivered within a shorter time (Anderson, 2003), The reusability of high quality software components at an affordable cost and within in a limited time scale is always desired by reuser (Fazal-e-amin et al, 2011).

However, a great deal of research over the past several years has been devoted to the development of methodologies to create reusable software components and component libraries, where there is an additional cost involved to create a reusable component from scratch. That additional cost could be avoided by identifying and extracting reusable components from the already existing environment. By the reusability the component can have better qualified, cheaper cost, improved performance. The reusable software component works better than the existing software as they are created with overcoming of the existing software module. Software reuse solves the several software crisis. In order to reuse a software system component technology is the important technology. Components are nothing but the smaller module which consist of classes and services which is defined in an application software system. Components are betrothed for reuse. Component Based software systems cogitate in decay of software system into function and logical component into distinct interface. Component Based Software Development (CBSD) process the software organization is split up into several categories based on the role.

However, recently many techniques such as fuzzy logic, Neuro-fuzzy have taken lead due to their power of predictability. It seems that in the domain of defuzzification (Fuzzy Logic) a designer has too wide possibilities of choices, so that some indicators in connection of defuzzification approach are welcome. The defuzzification technique selection essentially influences the output value determined by selected method, hence analyzing the effects of the different defuzzification methods in the evaluation of component reusability will be unavoidably useful.

The objective of this paper is to analyze the effect of the different deffuzzification methods in the evaluation of component's customizability in determining reusability.

II. BACKGROUND

Fuzzy logic with expert systems, neural networks, probabilistic reasoning, belief networks, genetic algorithms, chaos theory and parts of learning theory makes

https://dx.doi.org/10.24001/ijaems.3.9.10 *ISSN: 2454-1311*

complementary partnership of disciplines and technologies known as softcomputing, which gives methods for solving complex problems in designing intelligent systems with the ability to exploit the tolerance for imprecision, uncertainty and partial truth, to achieve tractability, robustness and low solutioncost.

Fuzzy logic is a logical system, which is the extension of multivalued logic. In a wider sense fuzzy logic is almost synonymous with the theory of fuzzy sets, a theory which relates to classes of object with unsharp boundaries in which membership is a matter of degree. So, soft computing is broadening the situations to which the artificial intelligence theories can be applied. The fuzzy sets theory – the soft computing constituent, can be used in modeling information pervaded with imprecision and uncertainty. Those characteristics make fuzzy models useful in great variety of applications.

Fuzzy systems(Zadeh, 1971; Cox, 1994; Šaletić et al, 2000) handle imprecise and uncertain information using the theory of fuzzy sets, (Kaufmann, 1974). The output of a fuzzy system is represented by an output fuzzy set. In applications, it is often needed to map the output fuzzy set onto a crisp output value that is done in a defuzzification process. Fuzzy inference is the process of formulating the mapping from a given input to an output using fuzzy logic. The mapping then provides a basis from which decisions can be made. Figure 1 presents the general FIS Architecture.

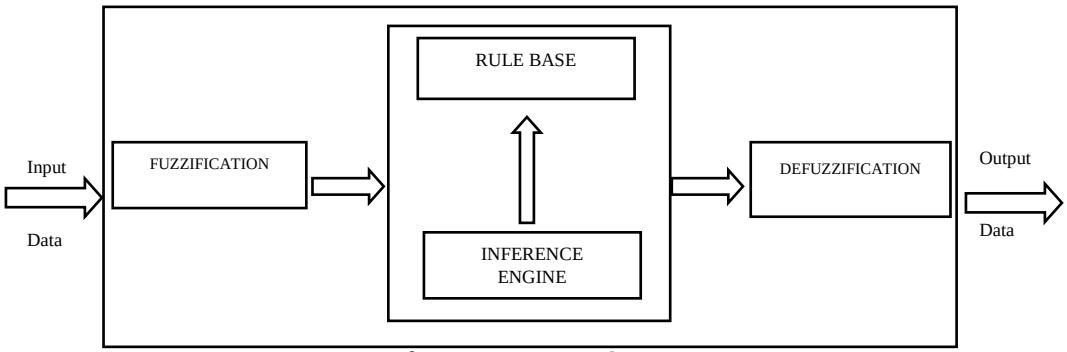

Fig.1: Fuzzy Inference System Architecture

III. RELATED WORKS

No doubt, notable works have been done relating to this work, for example (Dragan, et al, 2002): "Analysis of Basic Defuzzification Techniques" discussed improvement relative to the basic trial and error approach to the selection of the defuzzification technique. Features are given which are the base for a defuzzification techniques comparison. However, a considered example suggests that constrains on applicability of defuzzification techniques may exists. The researchers concluded that the result demands further research.

In Naaz et al (2011), the researchers implemented the fuzzy load balancing algorithm and compared the effect of using different defuzzification methods, they concluded that centroid, bisector and MOM methods are better as compared to the LOM, SOM, as there is more consistency in the results.

Some other researchers worked also on the effect of different defuzzification method. (Bajpai, et al, 2015)provided a detailed comparative analysis of FLC with different defuzzification methods. The performance of the drive investigated for speed control at load and at no load. Performance evaluation was carried out through simulation result. The system was dynamically simulated using Simulink/MATLAB Software.

IV. RESEARCH PROBLEM

As could be deduced from our literature, different fuzzification methods had been applied in different domain's problems to ascertain the effects of fuzzification methods. Works on reusability assessment has shown that customizability is a key factor that determine the reusability of components (Washizaki et al, 2003; Sharma et al, 2009; Sagar et al, 2010, Kumar et al, 2013; Goel et al, 2014). However, recent works also proved that different defuzzification methods present different results on evaluation. This prompted this work to delve into the analysis of the effects of different deffuzzification methods on the assessment of java components' reusability subject to their customizability.

RESEARCH AIM

This study extend the previous works by utilizing and analyzing more defuzzification methods and their effects in the assessment of component reusability taking component's customizability into consideration.

https://dx.doi.org/10.24001/ijaems.3.9.10 *ISSN: 2454-1311*

V. RESEARCH METHODOLOGY

1. Component Extraction. Four (4) Java Components were extracted.
2. Apply metrics. Customizability Metrics were applied to determine the reusable of the extracted components.
3. Evaluation. The evaluation was done with different deffuzzification methods as stated in the literature review. The FIS type used was Mamdani.

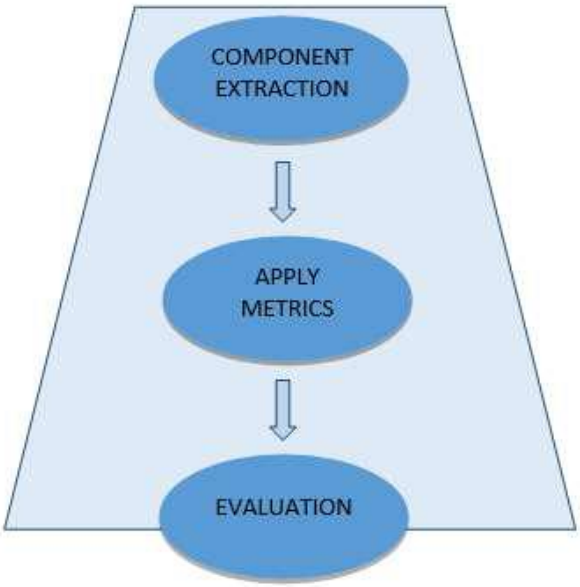

Fig.2: Study Methodology

VI. RESEARCH DESIGN

Architectural Design

Architectural design is a representation that enables a software engineer to analyze the effectiveness of the design in meeting its stated requirements (Pressman, 2001). Figure 3 presents the architectural design of our proposed system.

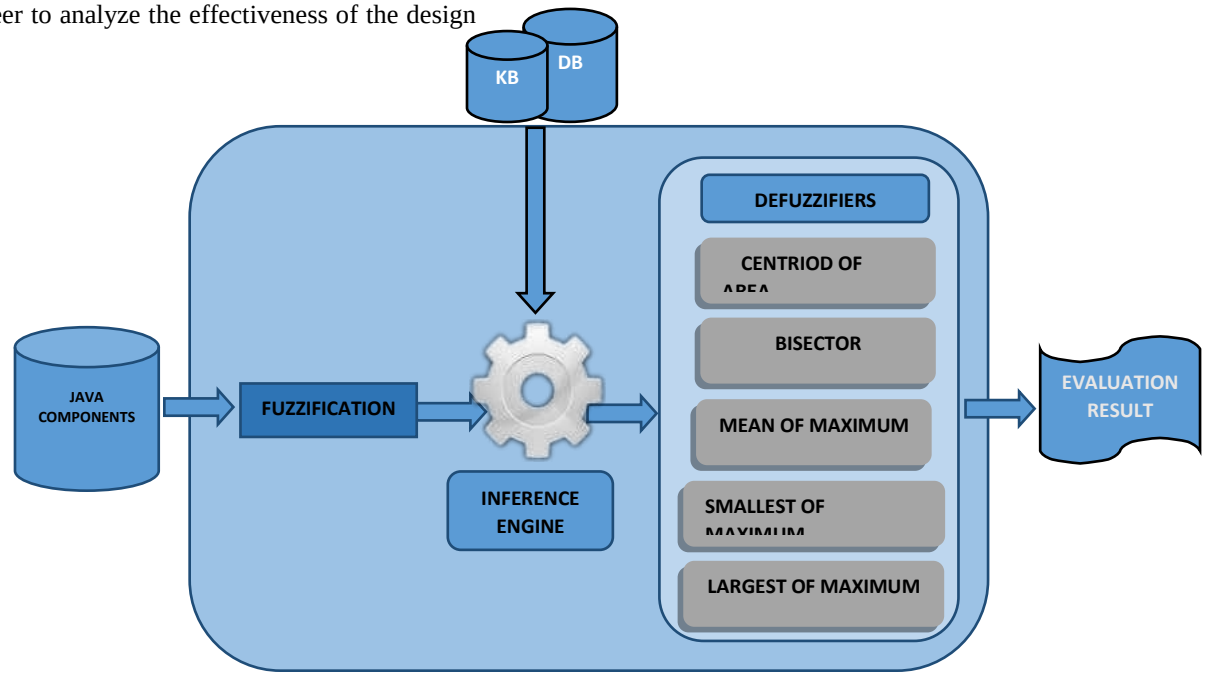

Fig.3:Architectural Design of the Proposed System

https://dx.doi.org/10.24001/ijaems.3.9.10 *ISSN: 2454-1311*

Defuzzification

Defuzzification entails transforming the resultant fuzzy values into numerical values. This results into generation of output. It is the opposite of fuzzification and could therefore be described as the conversion of fuzzy quantity to a precise quantity. Naas et al, 2011 simply put it as the process of converting the resultant fuzzy values into crisp values. We have varying types of defuzzification methods, these include:

i. Centroid Of Area Method(Center of Gravity / Centre of Area)

The output fuzzy set membership function is treated as adistribution is the main characteristic of this method, for which the average value is evaluated. Due to that heuristicapproach, the output has continuous and smooth change for the change of values of input variable in the universe ofdiscourse.

Equation 1 is the mathematical model of Centroid Membership Function.

$$x^* = \frac{\sum_{i=1}^{n} \mu A(x) * x}{\sum_{i=1}^{n} \mu A(x)} \quad \text{... (1)} \quad \text{(Yen and Langari, 1999)}$$

where:

x is the output variable

$\mu A(x)$is the aggregated membership function

x* is the crisp (defuzzified) output

n is the number of variable, while i is the variable counter

ii. Bisector Method

The bisector is the vertical line that divides the region into two sub-regions of equal area. It is sometimes, but not always coincident with the centroid line.

$$\int_{\alpha}^{z_{BOA}} \mu_A(z)dz = \int_{z_{BOA}}^{\beta} \mu_A(z)dz$$

... (2)

iii. Mean Of Maximum

This method gives as a result of defuzzification an element from a fuzzy set core. A fuzzy set core (designated as *core*) consists of elements of a universe of discourse on which that set is defined with the highest degree of membership to the fuzzy set.

$$Z^* = \frac{a+b}{2} \quad \text{…(3)}$$

Those techniques are convenient for the general fuzzy expert systems. They are computationally efficient,

iv. Smallest Of Maximum

It selects the smallest output with the maximum membership function as the crisp value. In other words, Smallest of Maximum chooses smallest among all z that belong to [z1, z2]

v. Largest Of Maximum

Largest of maximum takes the largest amongst all z that belong to [z1, z2] as the crisp value.

Components Data

One principle that guides component-based software development, is that, tested components could be purchased rather than built from scratch (Sharma et al, 2006; Sharma et al, 2009; Bharwaj, 2010; Kumar et al, 2013). In view of this, we extracted four (4) components from www.jidesoft.com. Table 1 shows the sources, nature and numbers of such components.

https://dx.doi.org/10.24001/ijaems.3.9.10 *ISSN: 2454-1311*

Table.1: Components Used

S/N	Component Source	Nature of Components	Number of Components	Date Extracted
1.	www.jidesoft.com	Java Components	4	01/09/2016

The extracted features shall be used in the computations of the metrics in other to determine reusability.

Customizability

Customizability is one of the quality factors that determines component reusability. It implies the adaptability capability of the component. In another tongue, it can be refer to as the modifiable characteristics of the component. Washizaki et al (2003) viewed it as the ease with which a component can be adapted to fulfill a requirement that differs from that for which it was originally developed. It indicates the built-in capability for supporting the customization and configuration of a component's internal functional features. Kumar et al (2013) sees it as the ease of modification in component when needed in application. Customizability can be measured on the basis of writable properties available in a component as shown in equation (4).

$$\text{Customizability (CZ)} = \frac{\text{No. of writable method}}{\text{Total no of properties}} \quad \ldots(4) \quad \text{(Washizaki, 2003)}$$

Customization values range from 0 to 1. Past empirical works (Washizaki et al, 2003; Kumar et al, 2013) established that, the higher the customizable level, the higher the level of reusability. Table 2 shows the computed CZ.

Table.2: Customizability Values

S/N	Component Name	No of writable property (NWP)	No of property (NP)	CZ = NWP / NP
1.	Textview	2	3	0.67
2..	splash	2	3	0.67
3..	Preferencepane	2	4	0.50
4..	fileapplication	4	5	0.80

VII. IMPLEMENTATION

MATLAB software was used as tool for the implementation of this work. The Input Variable named CZ (CUSTOMIZABILITY) has three input linguistic variables Low (0,0.25,0.50), Medium (0.25,0.50,0.75) and High (0.50,0.75,1.0) while the output variable, Reusability has also three variables as its output linguistic, namely Low (0,0.25,0.50), Medium (0.25,0.50,0.75) and High (0.50,0.75,1.0). With one (1) input variable for the experimentation and three (3) linguistic values, we have 3^1 rules (3 rules) generated. These were formulated as:

If (CZ is Low) then (REUSABILITY is Low) (1)

If (CZ is Medium) then (REUSABILITY is Medium) (1)

If (CZ is High) then (REUSABILITY is High) (1)

For cost effectiveness, Mamdani FIS type was used for this work, with different defuzzification methods deployed for each experimental setup. Table 3 shows the FIS structure for the study.

Table.3: FIS Structure

[System]
Name='CZ'
Type='mamdani'
Version=2.0
NumInputs=1
NumOutputs=1
NumRules=3
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

https://dx.doi.org/10.24001/ijaems.3.9.10 *ISSN: 2454-1311*

Figures 4 to 5 further show the FIS specifications for the some of the used defuzzification methods (Input and Output).

Fig.4: Triangular Membership Specification (Centroid-Input)

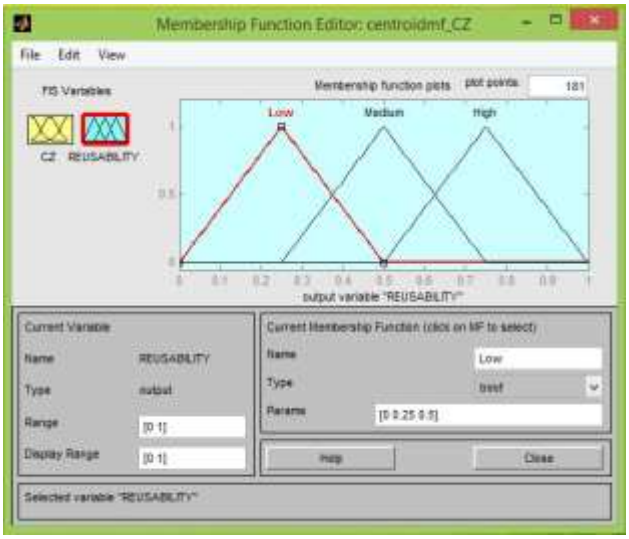

Fig.5: Triangular Membership Specification (Centroid-Output)

The reusability outputs of some of the different defuzzification methods are presented in Figures 6 to 17, while Table 4 shows the entire results according to the defuzzification methods applied.

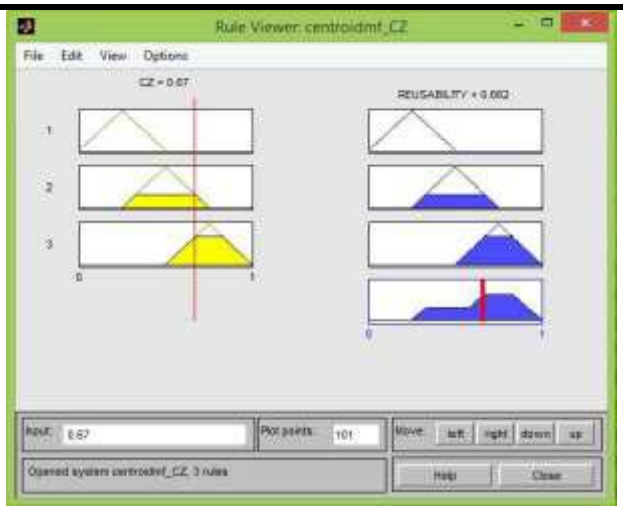

Figure 6: CZ Reusability Results (Centroid– Component1)

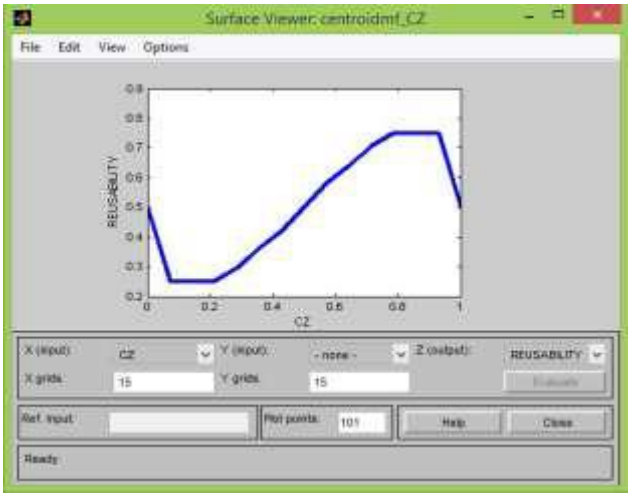

Figure 7: CZ Reusability Chart (Centroid – Component1)

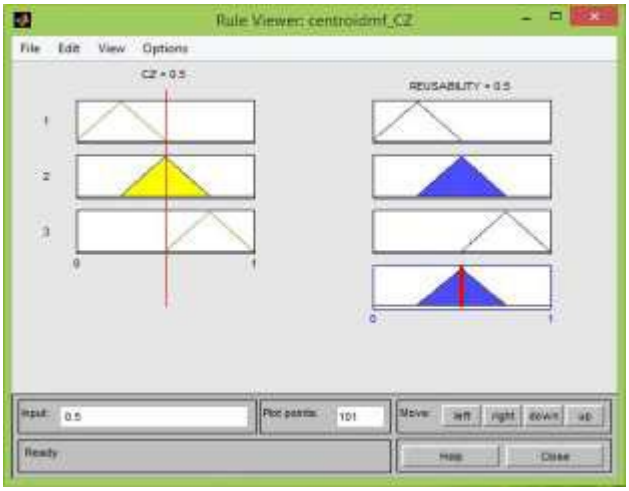

Figure 8: CZ Reusability Results (Centroid– Component 3)

https://dx.doi.org/10.24001/ijaems.3.9.10 *ISSN: 2454-1311*

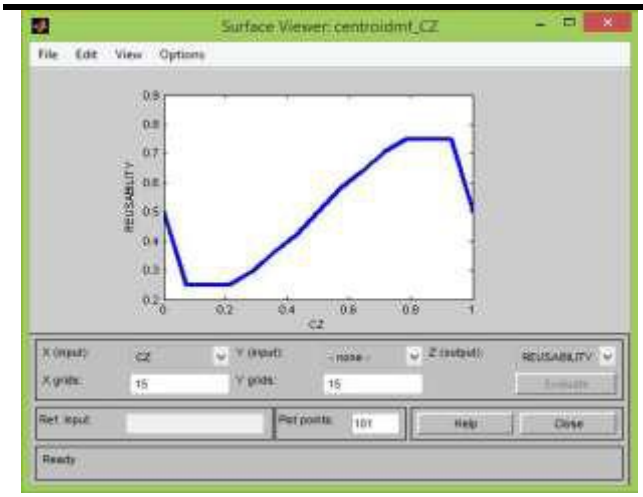

Fig. 9: CZ Reusability Chart (Centroid– Component 3)

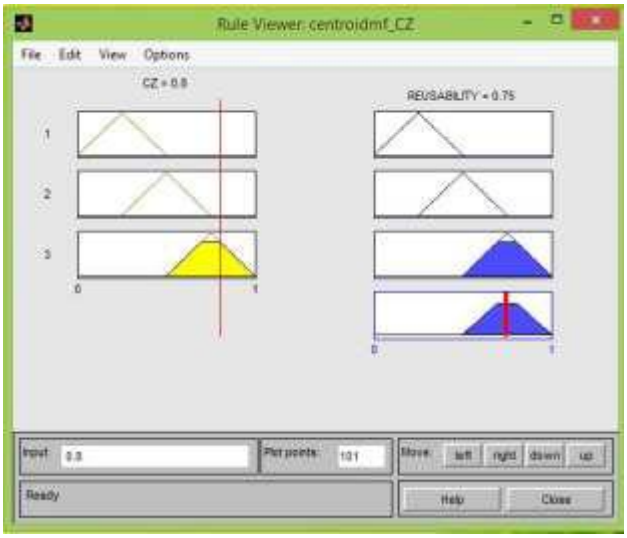

Fig.10: CZ Reusability Results (Centroid– Component 4)

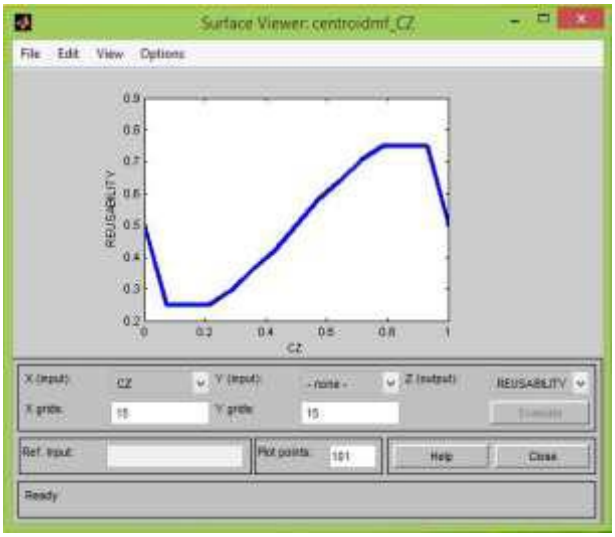

Fig.11: CZ Reusability Chart (Centroid– Component 4)

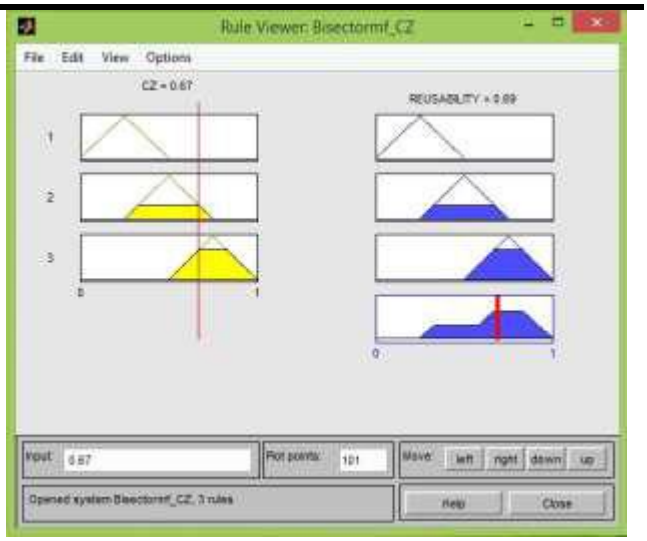

Fig.12: CZ Reusability Results (Bisector– Component1)

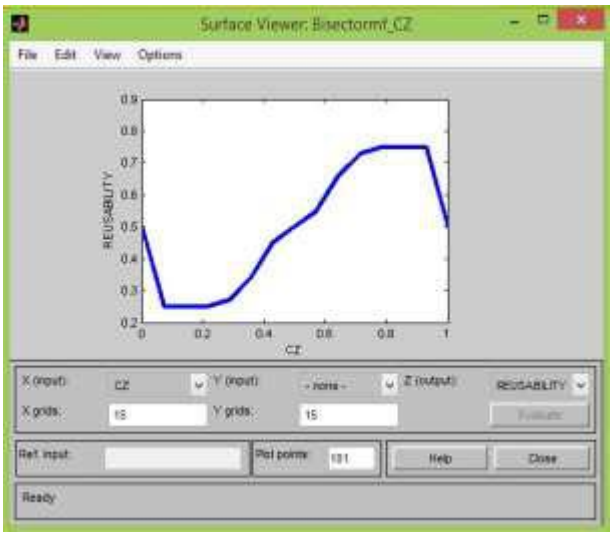

Fig.13: CZ Reusability Chart (Bisector – Component1)

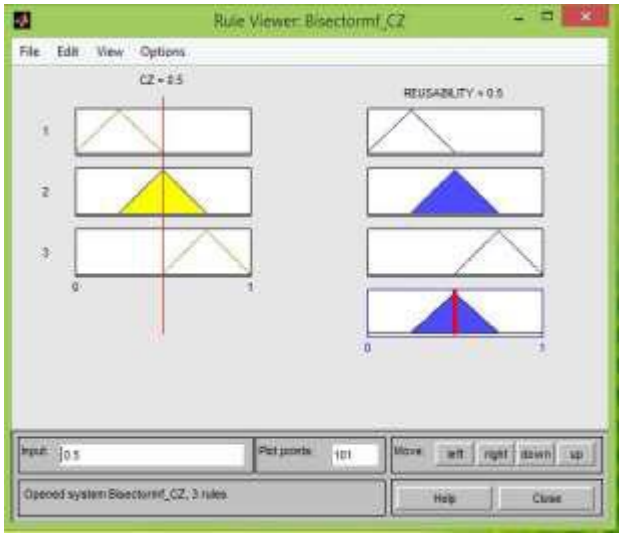

Fig.14: CZ Reusability Results (Bisector– Component 3)

https://dx.doi.org/10.24001/ijaems.3.9.10 *ISSN: 2454-1311*

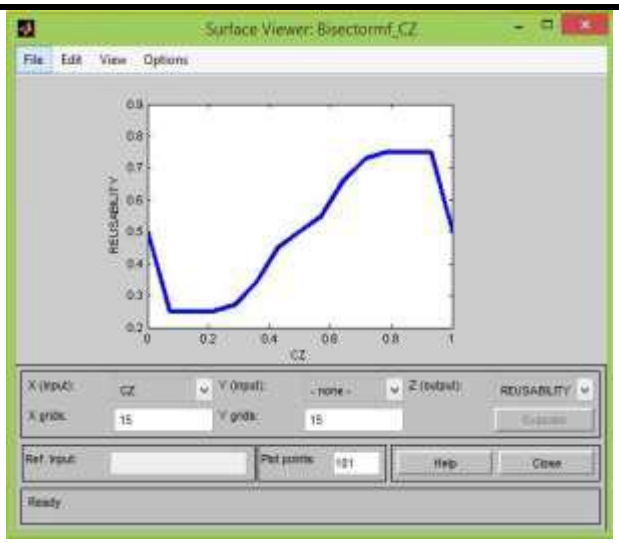

Fig.15: CZ Reusability Chart (Bisector– Component 3)

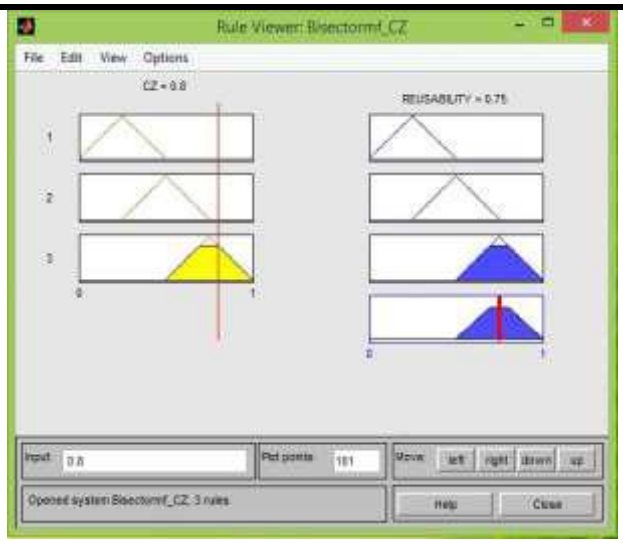

Fig.16: CZ Reusability Results (Bisector– Component 4)

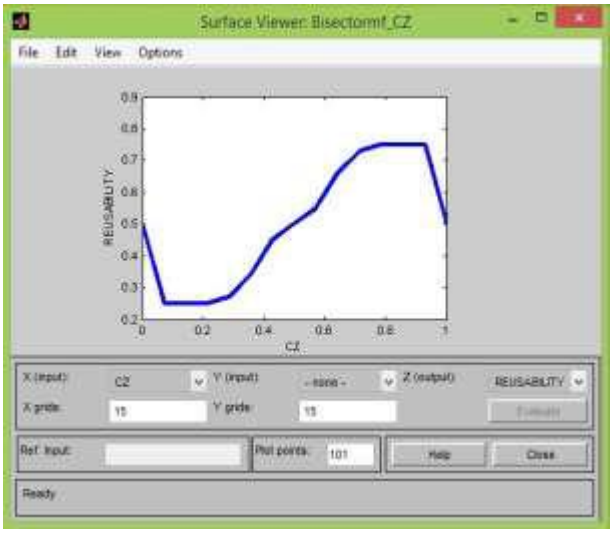

Fig.17: CZ Reusability Chart (Bisector– Component 4)

Table 4: Fuzzification Methods and their Customizabilty Reusability Outputs

Fuzzification Method	Component Type	Input Value	Reusability
Centroid	Component I	0.67	0.662
	Component II	0.67	0.662
	Component III	0.50	0.500
	Component IV	0.80	0.750
Bisector	Component I	0.67	0.690
	Component II	0.67	0.690
	Component III	0.50	0.500
	Component IV	0.80	0.750
Mean of Maximum (MOM)	Component I	0.67	0.745
	Component II	0.67	0.745

→ Highest Reusability Value for Component2

https://dx.doi.org/10.24001/ijaems.3.9.10 *ISSN: 2454-1311*

	Component III	0.50	0.500
	Component IV	0.80	0.750
Largest of Maximum (LOM)	Component I	0.67	0.820
	Component II	0.67	0.670
	Component III	0.50	0.500
	Component IV	0.80	0.800
Smallest of Maximum (SOM)	Component I	0.67	0.670
	Component II	0.67	0.670
	Component III	0.50	0.500
	Component IV	0.80	0.700

Highest Reusability Value for Component1

Highest Reusability Value for Component14

VIII. DISCUSSION

The results presented in Table 4 above show that LOM returned the highest reusability value for Component 1, while Centroid returned the least for that component. For Component 2, MOM predicted better than the rest method, while Centroid again returned the least for the component in used.All the methods returned same values for the reusability of Component 3. Lastly, LOM returned the highest reusability value for Component 4, while SOM returned the least value.

IX. CONCLUSION

Though the results presented show that all defuzzification methods applied returned values that indicate high reusability of components, this was with varying degree of reusability outputs. As against Naaz et al (2011) submission that, Centroid, Bisector and MOM produced better and consistent results compared to others, our results proved that LOM returned the highest and most consistent in terms of outputting reusability values of the components used.

FUTURE RESEARCH DIRECTION

By this work, we have justified the need for considering the role and effects of defuzzification methods in the evaluation of components reusability. Beeping into the future however, it is hoped that we can further stretch this work to first, cater for larger number of components. Also, to analyze the effects of newer defuzzification methods on more reusability quality factors.

REFERENCES

[1] Bajpai, D., and Mandal, A. (2015). Effect Of Different Methods On The Performance Of Fuzzy Logic Controller For PMSM Drives. International Journal Of Engineering Research & Technology (IJERT) ISSN: 2278-0181,Vol. 4 Issue 02

[2] Basili, V.R. (1989). Software Development: A Paradigm for the Future. Proc. COMPAC '89, Los Alamitos, Calif.: IEEE CS Press, pp. 471-485.

[3] Boehm,B.W.,and Ross, R. (1989). Theory-W Software Project Management: Principles and Examples," *IEEE Trans.Software Eng*., vol.15, no. 7, pp. 902.

[4] Caldiera, G., and Basili, V. R. (1991). Identifying and Qualifying Reusable Software Components", IEEE Computer, pp. 61-70.

[5] Cox, E., (1994). *The Fuzzy Systems Handbook*, Academic Press Professional, Boston.

[6] Dragan Z.S., Dusan M.V, and Nikos E.M. (2002). Analysis of Basic Deffuzification Techniques

[7] Fazal-e-Amin, Mahmood, A. K., and Oxley, A. (2011). A review of Software Component Reusability. Trends in Applied Sciences Research, 7(2): 118-131, ISSN 1819-3579. Academic Journal Inc.

[8] Goel, S., and Shariva, A., (2014). Neuro-Fuzzy based Approach to Predict Components Reusability. International Journal of Computer Application, Vol 106, No. 5

[9] Kaufmann, A.(1974). *Introduction to the Theory of Fuzzy Subsets*, Vol. I, Academic Press,New York.

[10] Kumar, V., Kumar, R., and Sharma, A., (2013). Applying Neuro Fuzzy Approach to build the ReusabilityAssessment Framework across Software Component Releases. An EmpiricalEvaluation. International Journal of Computer Application, 70 (15): 41-47

[11] Lim W. (1994). Effects of Reuse on Quality, Productivity, and Economics, IEEE Software, vol. 11, no. 5, pp. 23,30.

[12] Mili, F., Mili and A., Mili. (1995). Reusing Software: Issues and Research Directions, IEEE Transactions on Software Engineering, Volume 21, Issue 6, pp. 528 - 562.

https://dx.doi.org/10.24001/ijaems.3.9.10 *ISSN: 2454-1311*

[13] Naaz S., Alam A., and Biswas R. (2011) Effect Of Different Deffuzification Methods In A Fuzzy Based Load Balancing ApplicationIJCSI International Journal Of Computer science Issues, Vol. 8, Issue 5, No 1, ISSN (online): 1694-0814

[14] Roger, S. Pressman, (2001), Software Engineering, A Practitioner approach. McGraw Hill, New York

[15] Safar, S., Nerurkar, N.W., and Sharma, A.(2010). A Soft Computing based approach to estimate reusability of software components. ACM SIGSOFT Software Engineering Notes, Vol-35, No 4, pp 1-5

[16] Šaletić, D.Z., Velašević. D.(2000) The formal description of a rule based fuzzy expert system, *Proceedings of Symposium onComputer Sciences and InformationTechnologies*, YUiNFO 2000, Kopaonik 27-31. III 2000,(on CD-ROM), pp. 251 – 220.htm (in Serbian)

[17] Sharivia, A., Kumar, R., and Grover. (2009). ReusabilityAssessment for Software Components, ACM SIGSOFT Software Engineering Notes. Vol- 34 Issue 2, pp 1-6

[18] Smith E., Al-Yasiri. A., and M. Merabti.(1998). A Multi-Tiered Classification Scheme for Component Retrieval*Proc. Euromicro Conference*, 1998, 24th Volume 2, 25-27 Aug. pp. 882 – 889.

[19] Washizaki, H., Yamarmofo, H., and Fukazawa, Y (2003). A Metric Suite for Measuring Reusability of software components. Proceedings of the 9th International Symposium Software Metrics, Sept 3-5, Sydney, Australia, PP 201-211

[20] Yen, J., and Langari, R. (1999). Fuzzy-Logic-Intelligence, Control and Information. Upper Saddle River, NJ: Prentice Hall

[21] Zadeh L.A. (1971). Toward a Theory of Fuzzy Systems, in: *Aspects of Network and System Theory*, Kalman,R.E., DeClaris,N. eds., Holt, Rinehart and Winston, New York, pp. 469-490