Applying Neuro-fuzzy Approach to build the Reusability Assessment Framework across Software Component Releases - An Empirical Evaluation

Vijai Kumar
Product Engineering Services, Aricent Technologies, Gurgaon, India

Rajesh Kumar
School of Maths and Computer Applications, Thapar University Patiala, India

Arun Sharma
Krishna Institute of Engineering and Technology, (UPTU), Ghaziabad, INDIA

ABSTRACT
To reduce the development time, software reuse methodologies have been used across the software industries. Software reuse is a method to assemble the software components from the existing software. To take advantage of reuse concept, it is necessary to measure the software reusability of the existing components. Although there are various statistical methods exists to find the reusability of the components but soft computing has not been explored for component reusability. The aim of this paper is to formulate, build, evaluate, validate and compare neuro-fuzzy approach in prediction of software reusability of software components during the subsequent releases of a software development process. In this research we have applied neuro-fuzzy approaches which yield to better accuracy than the standalone fuzzy and neural approach. We have taken four main dependent factors to estimate the reusability of software components. This proposed approach has also been validated against different releases of open source development. Also we have proposed a framework for component reusability Management in software component intermediate releases using the neuro-fuzzy approach. The analysis and results of the study shows that neuro-fuzzy provides better results as compare to Fuzzy Inference System and neural network but applicability of best approach depends on the data availability and the quantum of data.

General Terms
Software Metrics

Keywords
Components, Component based system, neuro-fuzzy, reusability, Software Metrics, Prediction.

1. INTRODUCTION
Software Development process contains various phases during the development of a software entity. In component based systems development (CBSD), reusability of a component is an important aspect, which gives the assessment to reuse the existing developed component. If an existing component is used after proper assessment, it reduced the risk, time and cost of the project development process. To reuse the components, it is necessary to predict or asses the component reusability with better accuracy, so that the actual advantage of component based systems and design can be taken in a project. After assessment of a component, if component reusability does not comes out to be upto a threshold level then it may not be good to reuse the component as it can lead to overwork and may increase the risk, integration time and cost. Due to these types of requirements in software development process, researchers have been trying to find the component reusability using statistical and other conventional techniques. Recently interdisciplinary techniques such as fuzzy logic, ANN, Neuro-fuzzy have taken lead due to their power of predictability.

In this paper, the section 1 provides the introduction to the reusability for CBS. The basic fuzzy logic introduction is given in section 2. Section 3 describes the concept and variable used. Section 4 gives the detailed explanation about the proposed approach. Section 5 explains the reusability framework across releases, section 6 discusses the results and application of approach. Conclusion is given in section 7.

2. FUZZY TECHNIQUE
Fuzzy Logic is the main constitute of the soft computing approaches. Fuzzy Logic can be used in conjunction with neural networks, genetic algorithm, probabilistic reasoning etc. It is a good tool standalone and also in some cases efficient if combined with other methods. Fuzzy Logic is a mathematical tool for dealing with uncertainty and also it provides a technique to deal with imprecision and information granularity [1]. Fuzzy logic offers a particularly convenient way to generate a mapping between input and output spaces by using natural expressions [2]. In direct contrast to neural networks, which take training data and generate opaque models, fuzzy logic is based on if-then rules, which are designed by considering the opinion of experts from that domain. It has been found that the most accurate prediction models are based on analogy and experts opinion. Expert-based estimation was also found to be better than all regression-based models [3]. Henceforth the use of fuzzy logic in reusability prediction is desirable since expert knowledge can be incorporated into the fuzzy reusability prediction models.

Major advantage of this approach is that it is less dependent on historical data. Fuzzy logic models can be constructed without any data or with little data [4] [5]. This makes fuzzy logic superior over data-driven model building approaches such as neural network, regression and case-based reasoning. In addition, fuzzy logic models can adapt to new environment when data become available [6]. Zadeh [7] introduced the concept of fuzzy logic to represent vagueness in linguistics and to further implement and express human knowledge and inference capability in a natural way. Fuzzy logic starts with the concept of a fuzzy set. A fuzzy set is a set without a crisp, clearly defined boundary. It can contain elements with only a partial degree of membership. A Membership Function (MF) is a curve that defines how each point in the input space is mapped to a membership value (or degree of membership) between 0 and 1. The input space is sometimes referred to as the universe of discourse. Triangular and trapezoidal membership functions are the simplest membership functions formed using straight lines. The most important thing to

realize about fuzzy logical reasoning is the fact that it is a superset of standard Boolean logic.

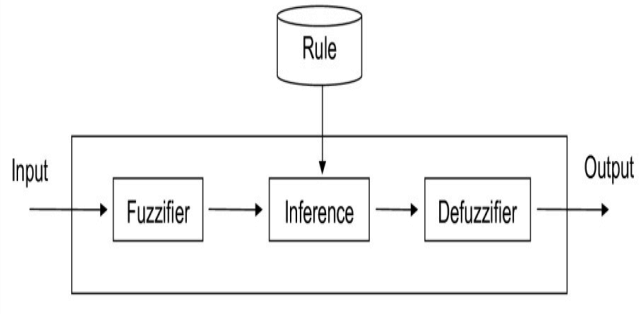

Fig 1: Fuzzy Inference System

Figure 1 elaborates the fuzzification and defuzzification. The inference rules are defined and the decision is made based on these if-then rules. The input values are defined and then output comes based on the rules defined in the inference system.

3. RELATED WORK TO PREDICT REUSABILITY

There is a number of metrics available to measure the reusability for object-oriented systems. These metrics focus on the object structure, which reflects on each individual entity such as methods and classes, and on the external attributes that measures the interaction among entities such as coupling & inheritance. But there are some difficulties in applying existing object oriented metrics into the component development and CBSD. Object oriented metrics cannot be used to measure the component's quality. An important issue in choosing the best component for reusability is deciding which components is more easily adapted. Generally, good guidelines for predicting reusability are: small size of code, simple structure and good documentation. Starting from the assumption that two functions have the same functionality, these three guidelines are used in our system to rank candidate functions for reuse.

Kumar et al. [15] presented an exhausted review on quality aspects of the component based systems. Reusability is an important and main factor of software quality. The author conducted a review of the research papers related to quality of components based systems based on various factors including practicability, validation proof etc. and concluded that soft computing approaches has not been explored in this are so far. In the earlier research by Kumar et al. [14] gives the chance to decrease the software defect density in a release by predicting it based on the input from proposed approach for the subsequent releases of a software product. The outcome proved ANN results better than FIS in predicting the defect density. Sharma et al. [20] present the ANN based method to predict the reusability of components. They concluded that more number of components for may produce better results for the training and testing. Gill [13] discusses the various issues concerning component reusability and its benefits in terms of cost and time- savings. Paper also provides some guidelines to measure the level of software reusability in component-based development. These guidelines include detailed software reuse assessment to measure the potential for practicing reuse, cost-benefit analysis to decide whether or not reuse is a worthwhile investment, adoption of standards for components to facilitate a better and faster understanding of a component and a faster integration into a system, selecting pilot projects for wider development of reuse and identifying reuse metrics. Poulin et al. [17] presents a set of metrics used by IBM to estimate the efforts saved by reuse. The study suggests the potential benefits against the expenditures of time and resources required to identify and integrate reusable software into a product. Study assumes the cost as the set of data elements like Shipped Source Instructions (SSI), Changed Source Instructions (CSI), Reused source Instructions (RSI) etc. Paper proposes several other reusability metrics in terms of cost and productivity like Reuse cost avoidance, Reuse value added and Additional development cost, which can be used significantly for business applications.

Cho et al. [11] proposes a set of metrics for measuring various aspects of software components like complexity, customizability and reusability. The work considers two approaches to measure the reusability of a component. The first is a metric that measures how a component has reusability and may be used at design phase in a component development process. This metric, Component Reusability (CR) is calculated by dividing sum of interface methods providing commonality functions in a domain to the sum of total interface methods. The second approach is a metric called Component Reusability level (CRL) to measure particular component's reuse level per application in a component based software development. However, the proposed metrics are based on lines of codes and can only be used at design time for components.

Dumke and Schmietendorf [12] proposed a set of reusability metrics for JavaBeans components. The metrics are adapted from structured and object oriented design context and are based on the source code of the components. Therefore, it cannot be used by component integrators due to the non-availability of the source code. Washizaki et al [24] discusses the importance of reusability of components in order to realize the reuse of components effectively and propose a Component Reusability Model for black-box components from the viewpoint of component users. The model identified factors affecting reusability on the basis of an analysis of the activities carried out when reusing a blackbox component. The factors include understanding the functionality, adapting the component to the specific functional requirements and porting the component to a new environment. Authors also proposed several metrics related to these factors, namely Existence of Meta-Information (EMI), Rate of Component's Observability (RCO), Rate of Component's Customizability (RCC), Self-completeness of Component's Return Value (SCCr) and Self-completeness of Component's Parameter (SCCp). EMI and RCO metrics indicates that high value of readability will help user to understand the behavior of a component from outside the component. High value of RCC metric indicates the high level of customizability of component as per the user's requirement and thus leading to high adaptability. High values of SCCr and (SCCp) will lead to self completeness of a component and thus lead to high portability of the component. The paper also conducts an empirical evaluation of these metrics on various Java Bean components and set confidence intervals for these metrics. It also establishes a relationship among these proposed metrics. These metrics are applied on only for small Java Bean components and need to be validated for other component technologies like .NET, ActiveX and others also. Boxall and Araban [10] considered interfaces of the components to measure the reusability. Paper assumed that understandability of a component can be made through its interface properties and understandability affects the level of reuse. Paper also proposed some metrics by considering the size of the

interface, argument count, argument repetition scale and others. These metrics give a better understanding of the properties of component's interfaces, which may help in measuring the reusability of the component. However, proposed approach does not consider the other aspects in the interface, such as, argument complexity. Several other researchers also considered similar factors for measuring reusability. Like, Rotaru et al. [18] considered adaptability, compose-ability and complexity of a component to describe its reusability. Mili et al. [16] considered two aspects, reusability and usefulness while REBOOT (Reuse Based on Object-Oriented Techniques) [22] considered factors namely portability, flexibility, understandability and confidence to assess the reusability. Sagar et al. [19] used fuzzy logic to predict the reusability of component based system. Boetticher and Eichmann [9] considered factors namely adaptability, complexity and coupling and applied neural network based approach for measuring reusability of black box Ada components. However, results obtained from the experimentation were not quite appreciable with correlation between exact and experimented results was only 0.18, which was very low. Acharya and Sadananda [8] proposed Kohnen's self organizing maps (SOM) based approach from Artificial Neural Network to organize various software components into clusters based on their characteristics to promote software reuse. Singh and Saha [23] predicted the testability using the design metrics for object-oriented software. Shatnawi and Ziad [21] discussed many oversampling techniques that are used to improve the performance of prediction models and proposed to guide the oversampling process using the fault content (i.e., the number of faults in a module).

4. CONCEPT AND VARIABLE SELECTION

To develop the process to predict the reusability, the model needs to consider some dependent variables which can be given as an input to the developed method. We considered the same factors as were taken in previous work using ANN [20]. The following dependent factors are considered in the proposed ANFIS approach:

4.1.1 Customizability

Customizability is the ease of modification in component whenever needed in application. If customizability is upto expectation then the component shall be more reusable and also it will be easy to maintain the component for future rerelease and phases. The lifetime of the component will be high which will lead to a long term business perspective in terms of the component life time and the revenue generation for the long period of time. It is generally measured by writable properties of the component being assessed. Washizaki et al. [13] defined the formula to evaluate this metrics as:

$$\text{Customizability} = \frac{\text{No. of setter method}}{\text{Total no. of properties}}$$

This metrics can help to measure the adaptability in an interface that how much the interface is customizable. Hence this metrics can be used for reusability forecasting and values vary between 0 and 1. The high value means the component is highly customizable.

4.1.2 Interface Complexity

Components can be treated as black box, in which we have only doors to access and use it. In some cases source code of these components may be available but in case using the library and linking, the source code is also not available. The user or developer will have the only choice to use its interfaces. It leads to the inference that components should have well defined interfaces with less complexity as much as possible for high reusability. The high complexities of the interfaces will results to complex reuse and hence high efforts to change and understand the components. We used the approach proposed by Sharma et al. [25] to measure the interface complexity of components.

4.1.3 Understandability

In case of source code of the component is not available; the documentation is the only way to understand the feature and interfaces of the component. The documentation helps the user or developer in component integration and developing the module to interface with components.

If we have not detailed and clear documentation of an application program interface (API) of the component, we will be in big trouble and should get ready for high risk of over cost and over schedule. The document may be in form of manuals, demos, automatic help system and other information which can be important for the person who is going to reuse this. A well structured documentation should include the functional description, API reference manual, system admin guide, system high level architecture, complete system reference manual etc. Here an important aspect to be noted that there should not be any mistake or ambiguity in documentation. It has been experience a major issue in case of wrong pr ambiguous information the reference documents, which lead to a reliability and quality risk for a product which is being developed using these components. So the conclusion comes out that the documentation about the component should be easy to understand, completeness, unambiguous as much as possible for better reuse of the component. Not only the reusability but it will help in maintenance of the component as well as the product maintenance where it is being used.

4.1.4 Portability

Portability is the degree of ability that in case of environment change, the component is able to perform with defined requirement with no or little change whenever there is a need from development or business perspective. Early days there were not many types of platform available but as recently there are countless type of platform and environment are being developed. Although it may not be feasible that a component would work on all platform, there are certain standard bodies which defines some global platform standards. In this line of discussion, the component should work on the standard platform for wider reusability across the platforms. At least the component should perform on selected and specific standard defined platform without any change or if needed with little change and less efforts, cost. It can be concluded easily that for better and wide reusability of component, it should have high portability. If we can quantify the portability, it can be used as influencing factors for reusability assessment of a component.

4.1.5 Empirical Data

We collected the data for several components from web sources and in-house development. These components include very simple like calculator to complex like inventory management system and are developed by using different technologies ranging from Java beans, .Net to open source technologies. We analyzed each component carefully and extracted the data related to our parameters chosen for the study. The goal of our work is to develop a tool for measuring reusability of the software component. We consider that reusability is a measure of factors mentioned above. The

values of these factors can be measured by using the appropriate metrics. Customizability metric is the ratio of writable properties to the total number of properties. Documentation and portability can be classified from low to high categories. All these factors are normalized by using min-max normalization. Min-Max normalization is a linear transformation on the original data. It transforms the original input range into a new data range (typically -1 to +1 range) using the following formula:

$$v' = (\max_{target} - \min_{target}) * \frac{v - \min_A}{\max_A - \min_A} + \min_{target}$$

5. PROPOSED ANFIS APPROACH

In the current research, we have proposed two new approaches for component reusability. First approach is defined to predict the component reusability using neuro-fuzzy technique. Then further applied the prediction approach to define the component reusability management process across multiple and iterative releases of the component in software product.

The FIS and ANN have already been used by Sharma A. et. al. [3] to access the reusability of components but the author never applied the concept to use the approaches across the releases of components as well as not the neuro-fuzzy approach. It is proven in many proposed techniques that neuro-fuzzy give the better results as compare to standalone FIS or ANN because it uses the power of rules decision of FIS and adaptive nature of ANN in a single system together.

Let x_n, y_n, z_n, k_n be the values of the customizability, complexity, understandability and portability respectively then

$$R_{Cn} = F_{cn}[x_n, y_n, z_n, k_n]$$

Where R_{Cn} is the reusability of component C_n. F_{cn} is implemented using ANFIS in MATLAB taking x_n, y_n, z_n, k_n as input dependent variables.

In the proposed approach the neuro-fuzzy system is build, trained and tested in MATLAB:

- Build Fuzzy Inference System as FIS in MTALAB.
- Load the Training data in neuro-fuzzy system using MATLAB toolbox.
- NF based system is trained using both least squares method and back-propagation. In the forward pass the consequent parameters are calculated using least squares and in the backward pass the premise parameters are calculated using back-propagation.
- When training is completed, trained neuro-fuzzy system is validated with the testing data.

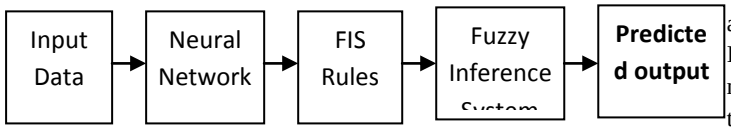

Fig. 2: Neuro Fuzzy modeling in MATLAB for reusability prediction

In the present work, the same variables have been applied in proposed neuro-fuzzy technique for the same data. A rule base is designed after getting the expert opinion on the relationship of these four variables with reusability by using Fuzzy Inference Engine and then data collected from 48 components were used to train the network by using back propagation neural network. Fuzzy Rule base along with Fuzzy membership function and ANFIS structure with four input parameters is shown below

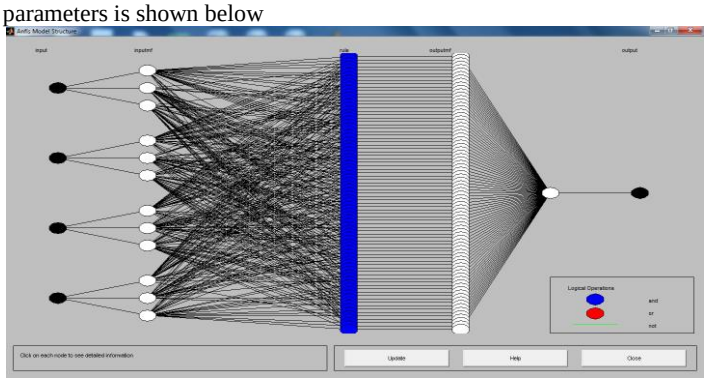

Fig. 3: ANFIS Model Structure

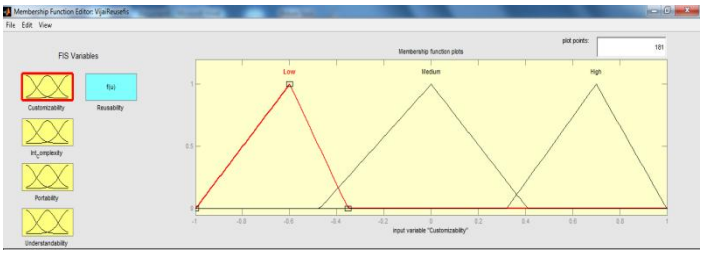

Fig. 4: Fuzzy Membership Function

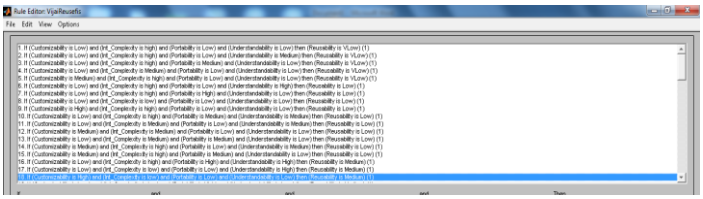

Fig. 5: Fuzzy rule Base

5.1 Proposed component reusability Management in software component intermediate releases using the ANFIS

Further to prediction of component reusability using neuro-fuzzy technique, the component reusability management is proposed which can be applied to various subsequent releases of component. To extend the application of the proposed efficient prediction approach, the approach is applied to different release of a single component. This process is applied to various components in the system being developed. If we use the proposed management approach, the management team can plan the effort and time distribution for the next software component release based on the prediction results using neuro-fuzzy approach. The following algorithm is conceptualized for the proposed method:

- Step-1: Collect the dependent data for input variables/attributes for the component Cn of first release R1 of the software system/product.
- Step-2: Apply the proposed neuro-fuzzy approach to predict the reusability of the component Cn for the next release R2.
- Step-3: Estimate the next release R2 Cn reusability based on dependent attributes in step 2.

- Step-4: Resource planning and management of release R2, Apply the resource distribution and focus on the area, which is responsible for less reusable component software.
- Step-4: repeat from step 1 for release R2.

Therefore if the data set is available and metrics values has been captured during the development process, neuro-fuzzy can be used for better results and take the advantage of adaptability. This process is applicable across various component releases of a single project and also for the different projects.

The proposed approach can be show as per the process chart shown below:

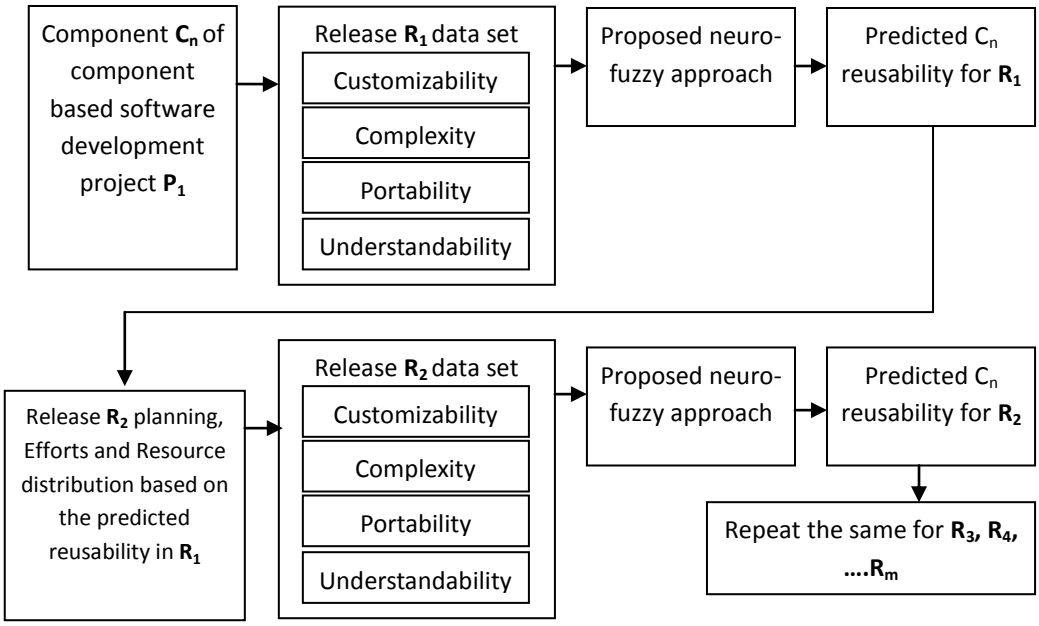

Fig 6: Proposed reusability prediction process framework across component releases

5.2 Comparison of Result with the Previous Approach

The best result obtained using ANN using the data from [20], the RMSE obtained equal to 0.1852. The data from [20] is used and applied ANFIS approach to predict the reusability of components and RMSE in this case is 0.1695, which shows that neuro-fuzzy based approach gives the better result for this prediction.

6. DISCUSSION OF PROPOSED APPROACH RESULTS AND PRACTICAL USAGE

Software reusability is a main concept in component based systems and design. To build and formulate the reusability metrics, the neuro-fuzzy engines are built to decide reusability of a component, which can be aggregated, for reusability at subsystem level of a product and hence reusability index of a software release. The outcome of the proposed research on metrics design shows that software industries can use neuro-fuzzy approach to predict the reusability of software components based of the data availability, which can help to maintain the better software product quality and resource management across releases.

This paper presents the investigation using neuro-fuzzy as well as comparative study of the proposed approach with ANN. Neuro-fuzzy shows good results and stands out in case of reusability prediction but FIS can also be useful in case on partial data of no data. Therefore if the data set is available and metrics values has been captured during the development process, NF can be used for better results and in case of less data availability FL is also useful tool for prediction of reusability.

The followings are the application of the proposed approaches:

- Using the proposed approaches, the software reusability can be predicted and put some preventive actions and resource utilization on the less reusability component.
- Using the power of adaptation and learning, NF can be used, which is simple but it is expensive to collect the data set during the tight schedule of development.
- After first release or from the similar type of project data set, we will have the dependent attributes values ready for the system so it may be a best reusability predictor tool for the particular component in next release as generally we used to have 50 to 100 releases in a large and long term software projects.
- As it has been proved a good estimator for different domain project, if the system needs to enhance or add new feature, during the planning of software project the proposed approach will be able to predict the reusability for the other components.
- Even for a new project initiation we can use as a reusability predictor for various components. In fact, we generally used to have some rough figures for dependent variables for a new project by taking the reference from the similar type of previous projects.

- It can be a good evaluator for Component on the shelf (COTS) product. For a COTS product, we will have the values of input variables in advance; hence, the reusability of a component can be estimated using the proposed approach, which helps us to evaluate the COTS product quality. It is obvious that if we have less reusability then the components will not used effectively in next releases or in other projects.
- The proposed approach can be used to predict the reusability of a component for the next release of the software product. Hence, it can optimize the resource loading, efforts and schedules for the release. It can be a part of regular practice during the software development and maintenance as release management process.

NF approach is adaptive in nature; hence, the method can be trained for different environment based on the data nature and variable.

7. CONCLUSION

Reusability is one of the most important factors for the success of any component based software product. If developed software is not highly reusable, then it will not be suggested for the integration in any project because it will increase the difficulty instead of the easiness in software development. We have used soft computing technique to predict the software component reusability before it completes the development of its own. If we use the proposed approach in the software development process then the final developed software would have the high reusability index. The proposed approach gives a different paradigm than the existing approaches, it provides the prediction of reusability during the various releases of the component development, hence gives the chance of increasing the reusability. We can predict the component reusability for first release of the software component and use this as an input and planning for the next release of the component, the resource and time can be distributed among the various components based on the reusability input from the proposed approach. Neuro-fuzzy gives better results as compare to FIS and ANN approaches but depends on the availability of data during the development process. Also the proposed concept of using the reusability prediction across the release is a new approach for

8. ACKNOWLEDGMENTS

Thanks to Aricent Technologies for continuous support to carry out the research.

9. REFERENCES

[1] Sivanandam, S. N., Sumathi, S., Deepa, S. N., 2007. Introduction to fuzzy logic using MATLAB, Springer.

[2] Zadeh, L. A., 2002. From Computing with numbers to computing with words-from manipulation of measurements to manipulation of perceptions, International Journal of Applied Mathematics and Computer Science, Vol.12, Issue 3, pp: 307-324.

[3] Musilek, P., Pedrycz, W., Succi, G., Reformat, M., 2000. Software Cost Estimation with Fuzzy Models, ACM SIGAPP Applied Computing Review, Vol. 8, pp: 24-29.

[4] MacDonell, S. G., Gray, A. R., Calvert, J. M., 1999. FLSOME: Fuzzy Logic for Software Metric Practitioners and Researchers, In the Proceedings of the 6th International Conference on Neural Information Processing ICONIP'99, Perth, pp: 308-313.

[5] Ryder, J., 1998. Fuzzy Modeling of Software Effort Prediction, Proceedings of IEEE Information Technology Conference, Syracuse, New York, pp: 53-56.

[6] Sailu, M. O., Ahmed, M., and AlGhamdi, J., 2004. Towards Adaptive Softcomputing Based Software Effort Prediction, Fuzzy Information, Processing NAFIPS' 04, pp: 16-21.

[7] Zadeh, L. A., 1965. Fuzzy Sets, Journal of Information and Control, Vol. 8, 1965, pp: 338–353.

[8] Acharya, S. and Sadananda, R. (1997) "Promoting Reuse Using Self-Organizing Maps", Neural Processing Letters, Issue 5, , pp: 219-226.

[9] Boetticher, G. and Eichmann, D. (1993), A Neuro-Fuzzy Based Software Reusability Evaluation System with Optimized Rule Selection, Austrelian Conference on Software Metrics (ACOSM, 93), , pp-1-11.

[10] Boxall M. A. S. and Araban S. (2004), Interface Metrics for Reusability Analysis of Components, Australian Software Engineering Conference (ASWEC'2004), Melbourne, Australia, , pp: 40-46.

[11] Cho, E. S., Kim, M. S. And Kim, S. D. (2001), Component Metrics to Measure Component Quality, 8th Asia-Pacific Software Engineering Conference, Macau, , pp: 419-426.

[12] Dumke, R. and Schmietendorf, A. (2000) Possibilities of the Description and Evaluation of software Components, Metrics News, , Volume 5, Issue 1, pp: 13-26.

[13] Gill, N. S. (2003), Reusability Issues in Component-based Development, ACM SIGSOFT Software Engineering Notes, Volume 28, Issue 6, pp: 30-36.

[14] Kumar, V., Sharma A.and Kumar. R. (2013), Applying Soft Computing Approaches to Predict Defect Density in Software Product Releases: An Empirical Study, COMPUTING AND INFORMATICS, volume 32, No.1, pp: 203-224.

[15] Kumar, V., Sharma, A., Kumar, R. and Grover, P. S. (2012), Quality aspects for component-based systems: A metrics based approach, Software: Practices and Experience, John Wiley & Sons, December, Vol 42, Issue 12, pp: 1531-1548 .

[16] Mili, H., Mili, F. and Mili, A. (1995) "Reusing Software: Issues and Research Directions", IEEE Transaction on Software Engineering, Volume 21, Issue 6, , pp: 528-561.

[17] Poulin, J., Caruso, J. and Hancock, D. (1993), The Business Case for Software Reuse, IBM Systems Journal, Volume 32, Issue 40, , pp: 567-594.

[18] Rotaru, O. P. , Dobre, M., Petrescu, M. (2005), Reusability Metrics for Software Components, IEEE International Conference on Computer Systems and Applications (AICCSA-05), Cairo, Egypt, , pp: 24-29.

[19] Sagar, S., Nerurkar N. W., Sharma A., 2010. A soft computing based approach to estimate reusability of software components, ACM SIGSOFT Software Engineering Notes, Volume 35 Issue 5, September pp:1-5

[20] Sharma A., Kumar, R., and Grover, P. S. (2009), Reusability assessment for software components, ACM SIGSOFT Software Engineering Notes,Volume 34 Issue 2, March, pp: 1-6.

[21] Shatnawi, R. and Ziad, A. (2012), A guided oversampling technique to improve the prediction of software fault-proneness for imbalanced data , Int. J. of Knowledge Engineering and Data Mining, Vol.2, No.2/3, pp.200 - 214

[22] Sindre, G., Conradi, R. and Karlsson, E. A. (1995), The REBOOT Approach to Software Reuse, Journal of Systems and Software, Vol. 30, no. 3, , 201-212.

[23] Singh, Y. And Saha, A (2012), Prediction of testability using the design metrics for object-oriented software, Int. J. of Computer Applications in Technology, Vol.44, No.1, pp.12 – 22.

[24] Washizaki, H., Hirokazu, Y., Yoshiaki, F. (2003), A Metrics Suite for Measuring Reusability of Software Components, Proceedings of the 9th International Symposium on Software Metric, , pp: 211-223.

[25] Sharma, A., Kumar, R., Grover, P. S., 2008. Empirical Evaluation of Complexity for Software Components, International Journal of Software Engineering and Knowledge Engineering (IJSEKE), Vol. 18, Issue 5, pp: 519-530.