Handling Software Engineering Tasks By Means of Soft Computing Techniques

Abhaya Nand

Research Scholar, JNU, Jodhpur, Rajasthan

***Abstract*— Software development Organizations today are looking for technologies that reduce costs, increase reliability, productivity, reusability, security and enhance the Software Maintenance process. The technology is rapidly being deployed everywhere. Despite the gloom in the global economy that has not spread any sector one thing that certainly seems to have stayed put on the agenda of most organization are the strategy sessions on how to increase reliability, productivity, maintainability while bringing the cost down. At present, Software development organizations are under continuous pressure to decrease costs and improve productivity. Software industry has to make reliable software to sustain their potential buyer and to face global competition. Well-planned, systematic and efficient Software Engineering tasks makes possible a more effective use of available resources and results in increased productivity and reduced wastage of money, material and manpower. Software failures of any kind should be avoided. In this paper, the evolution of Soft Computing techniques and its development & application to Software Engineering framework has been suggested. Software development organizations can certainly get benefit by applying these latest and important ideas of Soft Computing into their development and economic scenario.**

***Keywords*— Failure, Maintenance, Productivity, Reliability, Reusability, Security, Soft Computing, Testing.**

I. INTRODUCTION

The discipline of Software Engineering is striving to meet the ever growing challenges by acquiring new knowledge, devising tools and methods for gathering requirements, designing architecture and algorithms, implementing the code, verifying and validating systems and offer suitable means supporting their evolution over time. Soft Computing techniques have provided us with a unique opportunity to establish a coherent Software Engineering environment in which uncertainty and partial data and knowledge are systematically handled. In fact a traditional view of Software Engineering has been predominantly, if not exclusively, focused on quantitative and precise approaches, neglecting the fact that software constructs are inherently human centric, and thus uncertainty and partial knowledge are unavoidable.

In fact Soft computing is a collection of methodologies that aim to exploit the tolerance for imprecision and uncertainty to achieve traceability, robustness, and low cost solution. Soft computing has been presented not only with the theoretical developments but also with a large variety of realistic applications to software products and industrial systems. Applications of soft computing have provided the opportunity to integrate human-like vagueness and real-life uncertainty into an otherwise hard computer program. Soft computing is likely to play major role in building software models, thereby reducing time, cost and maintenance. Soft computing is based on natural as well as artificial ideas. It is referred as a computational intelligence. It differs from conventional computing that is hard computing. It is tolerance of imprecision, uncertainty, partial truth to achieve tractability, approximation, robustness, low cost solution, and better rapport with reality. In fact the role model for soft computing is human mind. Soft computing is a consortium of methodologies that works synergistically and provides, in one form or another, flexible information processing capabilities for handling real life ambiguous situations. Its aim is to exploit the tolerance for imprecision, uncertainty, approximate reasoning and partial truth in order to achieve tractability, robustness, low cost solutions, and close resemblance to human like decision making. Fuzzy Logic, Rough Sets, Artificial Neural Networks and Evolutionary Algorithms are the principal components of soft computing paradigm. The relevance of soft computing research in various activities of Software Engineering applications is evident worldwide.

Recently, due to the advent of internet technology, and the need for web intelligence, the research on soft web mining demanded immediate attention for the development of future generation information technology (IT) systems and services. This includes topics like data mining, natural language processing, E-commerce and Bioinformatics. Besides these, domains like intelligent manufacturing systems, telecommunication, and engineering applications may also be considered as prospective candidates for soft computing applications.

A. Neural Networks

Neural networks are being used increasingly in a wide range of real world applications. However, existing studies have reported major problems in the use of neural network in the field of software engineering. Here we analyse some of these problems, and describes how existing software engineering practices can address some of them. Other problems are identified that require new approaches tailored to neural network development. A unified framework is presented, in the form of neural model, aimed at instituting good software engineering practice for neural network development. The framework should be of interest to neural network software

developers and suggests research directions. Few applications of neural network are to be found in software metrics, effort estimation, cost estimation, reliability prediction, software testing, and reusability, understand ability and software maintenance.

B. Fuzzy logic

Fuzzy logic is mainly associated to imprecision, approximate reasoning, and computing with words, neuro computing to learning and curve fitting and probabilistic reasoning to uncertainty and belief propagation. Fuzzy logic, is one of the major constituents of soft computing, plays a key role to automate software development processes. Fuzzy Logic is an attractive area of research because it does a good job of trade off between significance and precision - something that humans have been managing for a very long time. It has been used in many important investigations since its invention. Fuzzy logic uses the expert knowledge into fuzzy rules directly. The applications of fuzzy logic are to be found in problems of software estimation, software reusability, software performance prediction and software maintenance.

C. Proposed Fuzzy model

In this paper we have proposed fuzzy model and trained a Fuzzy Inference System (FIS) with these four inputs, namely changeability, interface complexity, understandability of software and documentation quality to predict software reusability as output, as shown in Figure 1.

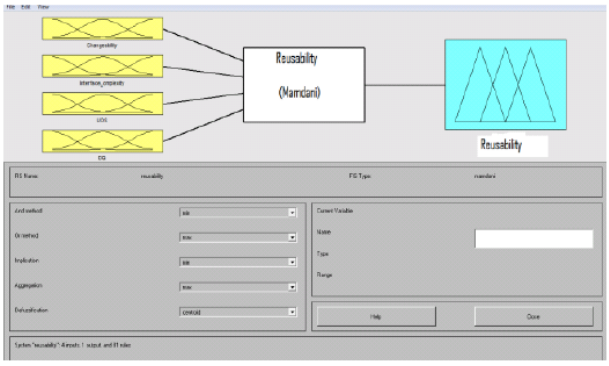

Figure 1: Software Reusability Model (Fuzzy Inference System) with 4 inputs, 1 output, and 81 rules

All inputs can be classified into fuzzy sets viz., Low, Medium and High. The output software reusability is classified as very Low, Low, Medium, High and Very High. In order to fuzzify the inputs, triangular membership function (trimf) is chosen namely Low, Medium and High. Similarly the output variable i.e. software reusability has five membership function (trimf) namely Very Low, Low, Medium, High and Very High. The proposed model considers all four inputs and provides a crisp value of software reusability using rule base. All 81 rules are inserted and a rule base is created as shown in figure 2, which represents all possible combination of inputs which leads to 34 i.e. 81 sets.

Figure 2: Rule Base for Proposed Fuzzy Model

Mamdani style of inference is used as shown in figure 1. Using the rule viewer, output i.e. software reusability is observed for particular set of inputs using MATLAB Fuzzy Defuzzification of the above output can be obtained by finding the Centre of Gravity of the above fuzzy output. The effect of these rules was observed by simulating the model in MATLAB Fuzzy Tool Box. The proposed fuzzy model is used for creating the inputs set with respective output as an input to the Neural Network for further validation.

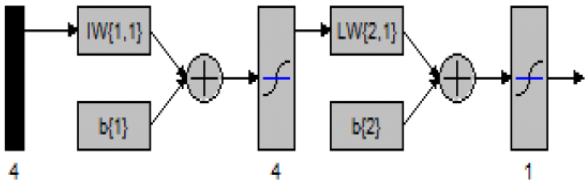

Figure 3: Trained Neural Network Model with 04 Inputs, 1 Output and 4 Neurons at the Hidden Layer

D. Proposed Neural Network Model

The data sets generated by the fuzzy model now acts as input datasets to our proposed Neural Network Model. It is known that NN model can be used to model any arbitrary input output mapping and are capable of approximating any measurable function. So NN should be able to model the functionality of the software reusability also. In a broad sense, the neural network itself is a model because the topology and transfer functions of the nodes are usually formulated to match the current problem. The NN model is further validated on the two validation sets generated by the fuzzy models. In our study back propagation supervised algorithm is used. The model is trained with four inputs namely: Changeability, Interface Complexity, Understandability of Software and Documentation Quality with one output i.e. software reusability. Trainbr, training function is used to train the network. We have used the linear transfer functions Tansig for the experimentation. The network is best trained with four input, one output and 4 neurons at the hidden layer as shown in figure 3.

The experiment is conducted to explore the possibilities of usage of NN for assessing software reusability level. The network used in this work belongs to multilayer feed forward networks and is referred to as M-H-Q network with M source nodes, H nodes in hidden layer and Q nodes in the output layer. The summary of ANN used in this study is shown in Table 1.

TABLE 1.
NN MODEL SUMMARIES

Architecture	
No. of Neurons at Layer 1 (hidden layer)	04
No. of Neurons at Layer 2 (output layer)	01
Input Units	04
Output Unit	01
Training	
Algorithm	Back Propagation
Training Function	Trainbr
Transfer	Tansig

The network is trained with four inputs by using trainbr as training functions with an output i.e. software reusability. The training and validation data set is randomly generated from the proposed fuzzy model. The ANN was trained on these data sets (140 sets of different data sets generated by the fuzzy model) by the standard error back propagation algorithm at a learning rate of 0.005, having the minimum square error as the training stopping criterion, as shown in figure 5.

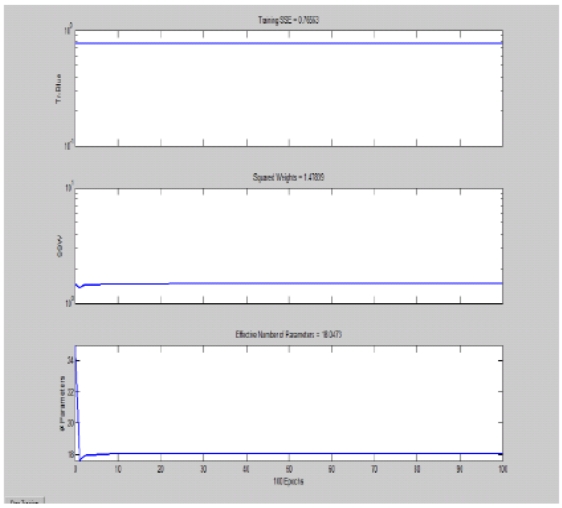

Figure 4: Training with Trainbr function

In this section we present the analysis performed to find relationship between independent variables (changeability, interface complexity, understandability of Software and Documentation Quality) and one dependable variable i.e. software reusability. The data set contains 140 different data sets generated by the rule base of fuzzy logic. The model is trained using training data sets and was evaluated on two validation data sets. Table 2 shows the MARE, and MRE results of ANN model evaluated on validation data set.

TABLE 2.
VALIDATION RESULTS OF ANN MODEL

Performance Measures	**Validation Set 1**	**Validation Set 2**
MARE	0.26	0.20
MRE	0.15	0.08

Comparison of the experimental results and measured software reusability of the two validation sets is shown in figure 5 and 6. From which it becomes evident that our proposed model produces reusability levels that are comparable to the measured reusability levels as predicted by the proposed fuzzy model.

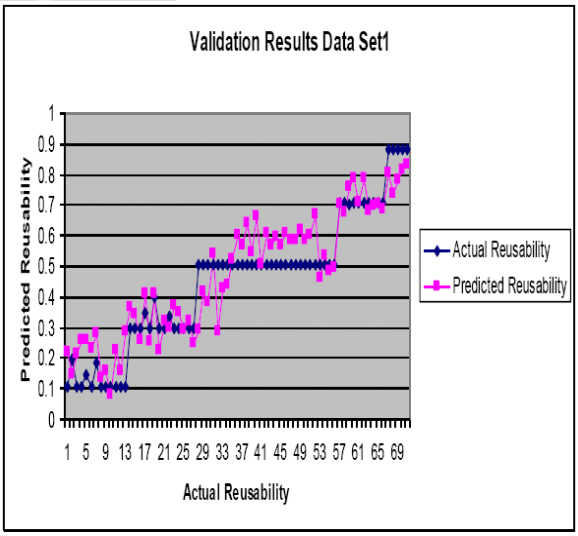

Figure 5: Validation Result (Data Set 1) with 04 neurons

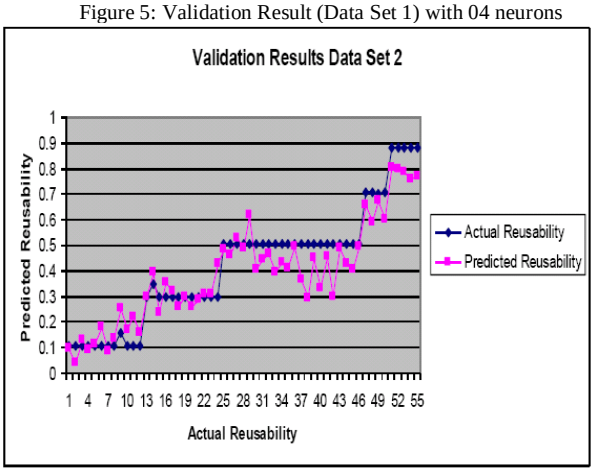

Figure 6: Validation Result (Data Set 2) with 04 neurons

II. OBJECTIVES

In our work neural networks, fuzzy logic principal constitutes of soft computing techniques that were widely used and found applicable to the planning, designing, testing and maintenance of Software Development Life Cycle(SDLC). The primary objective of our work was to investigate computational aspects in design and development of soft computing techniques in the field of Software Engineering specific to Software estimation, testing, reusability and finally the maintenance of the software development life cycle. The major findings of the study are briefly recapitulated point wise and certain points for future research have been discussed.

III. LIMITATIONS AND FUTURE SCOPE OF THE STUDY

We believe that this study shall mark the beginning of a new era in the field of software engineering with application of soft computing techniques. ANN techniques have been widely used for traditional software estimations. Similar types of studies need to be carried out with large data sets to validate object-oriented and aspect-oriented design metrics, across different fault severities. ANN model was proposed in the paper to automate the software testing process. Further investigations are required to use neural networks and neurofuzzy techniques universally for all types of implementation under Testing. Software reusability is considered as the main criterion for selection of the component from technical point of view. Soft computing techniques were proposed in the paper to predict software reusability level. In the paper, data used to train and validate the network was simulated from the designed rules.

Data collected from the real life projects can be used to train and validate the network for better results. The fuzzy model was proposed to predict maintenance level of the software. The approach, in combination with neural networks can be extended to predict maintenance time also. Effective utilization of soft computing techniques should result in increased productivity and quality, reduced cycle time, and lesser costs in the long run. Therefore, the future work in this direction should be initiated to further explore various emerging soft computing technologies, including neural computing techniques such as Support Vector Machine (SVM), Adaptive Neuro-Fuzzy Inference System (ANFIS), hybrid soft computing models based on neural, fuzzy and evolutionary computation technologies.

IV. CONCLUSION

It is quite evident that the development of Soft computing techniques progresses at a very fast rate in several disciplines which includes Software Engineering at the top. Hence Software professionals must be aware of their roles and brace themselves for the greater advancement of soft computing techniques in the future. This paper has outlined different mechanisms of handling Software Engineering tasks with the help of various Soft computing techniques. The successful applications of soft computing and the rapid growth suggest that the impact of soft computing in the area of Software Engineering will be felt increasingly in years to come. It encourages the integration of soft computing techniques and tools into both every day and advanced applications of Software Engineering. It is hoped that this gentle review will benefit Software professionals who are keen to contribute their works to the field of Soft computing.

REFERENCES

[1] William Harrison and Harold Ossher. "Object oriented programming (a critique of pure objects)." In Proceedings of the Conference on Object-Oriented Programming: Systems, Languages, and Applications (OOPSLA), September 1993.

[2] William J. Opdyke and Ralph E. Johnson. "Refactoring: An Aid in Designing Application Frameworks and Evolving Object-Oriented Systems." SOOPPA Conference Proceedings, ACM Press, September 1990.

[3] Wolfgang Deiters and Volker Gruhn. "Managing Software Processes in the Environment MELMAC." Proceedings of the Fourth Symposium on Software Development Environments, December 1990.

[4] Wolfgang Emmerich, Petr Kroha, Wilhelm Schäfer. "Object-Oriented Database Management Systems for Construction of CASE Environments." Proceedings of the Conference on Database and Expert Systems Applications (DEXA'93), 1993.

[5] Aggarwal K K, Yogesh Singh (2001) Software Engineering Programs, Documentation Operating Procedure. New Age International Publishers.

[6] Aggarwal K K, Singh Yogesh, Chabbra J K (2002a) An integrated Measure of Software Maintainability. Proceedings of Annual Reliability & Maintainability Symposium RAMS-2002: International Symposium on Product Quality & Integrity, Seatle Westin, U.S.A, Jan 28-31, 2002, pp 235-241.

[7] Aggarwal K K, Singh Yogesh, Arvinder Kaur, Malhotra Ruchika (2006) Application of Neural Network for Predicting Maintainability Using Object- Oriented Metrics. Transaction on Engineering, Computing and Technology, Vol. 15.

[8] Aggarwal K K, Yogesh Singh (2008) Software Engineering: Program, Documentation and Operating Procedure. New Age International Publishers, 3rd edition.

[9] Albrecht A J, Gaffney J E (1978) Software Function, Source Lines of Code and Development Effort Prediction. Software Science Validation, IEEE Transactions on Software Engineering, 24, pp 345- 361.

[10] Terry Quatrani and Grady Booch. Visual Modeling with Rational Rose 2000 and UML. Addison-Wesley, 1999.

[11] Robert DeLine. "Avoiding Packaging Mismatch with Flexible Packaging." In Proc. International Conf. On Software Engineering, 1999.

[12] Michael D. Ernst, Jake Cockrell, William G. Griswold and David Notkin. "Dynamically Discovering Likely Program Invariants to Support Program Evolution." In Proc. International Conf. On Software Engineering, 1999.