International Journal of Computer Information Systems and Industrial Management Applications.
ISSN 2150-7988 Volume 9 (2017) pp. 162-172

An Adaptive Neuro-Fuzzy Model to Predict the Reliability before Testing Phase

Rajni Sehgal[1], Deepti Mehrotra[2] and Manju Bala[3]

[1,2]Amity School of Engineering and Technology
Amity University Uttar Pradesh, Noida, India
[1]rsehgal@amity.edu,[2]dmehrotra@amity.edu

[3]I.P college for women, University of Delhi, Delhi
[3]manjugpm@gmail.com

***Abstract*: Reliability of a system has a direct impact on the success of any software system. Predicting the reliability at an early stage helps to optimize the testing and maintenance of the software. Rapid changes in hardware and software technologies lead to inventions of new methodologies and needs developing and validating a reliability predicting model for each method, Machine Learning approach can provide a solution to this problem. Selecting suitable metrics that affect the reliability of the system input to a model, and output of the model should reflect whether a system is reliable or not. In this paper, software is developed based on the design of Service-Oriented Architecture (SOA) methodology, is considered where a set of components interact for autonomous services. Further, to predict the reliability of a component, the probability that a component executes without fault, Adaptive Neuro-Fuzzy Inference System (ANFIS) approach is used. The coefficient of determination is evaluated to validate the model.**

***Keywords*:** ANFIS, Reliability, Machine Learning, Faults, FIS, Service-Oriented Architecture (SOA).

I. Introduction

Today software is becoming an essential part of everybody's life. It is not possible to perform any daily activity without software, whether it is listening to music, traveling by air, or it is handling some medical equipment. This increased dependency on software means that its reliability also needs to be equally high. Predicting the reliability at an early stage of Software Development Life Cycle (SDLC) makes the software less costly and expedites its timely release. Software testing and software maintenance are two vital activities in the software development process. A lot of research has been conducted to minimize the cost and effort involved in these activities, without compromising on quality and accuracy. The prime objective of testing is to produce a reliable software product, and that of maintenance process is to ensure its reliability at the working end. Predicting the reliability of a product before testing can provide valuable insights about the relative measures that can improve the overall planning of the testing and the maintenance process.

Numerous reliability prediction models for software such as Musa's execution time model [26], Putnam's model [16], Rome Laboratory model [4], and many others are available in the current literature to predict the software reliability. These models are based on the aggregation of previous failure data, and they can predict the reliability after the testing phase. Recent research has emphasized on early prediction of reliability [20]. The researchers had used various approaches and techniques like statistical techniques, data mining, and machine learning methods, evolutionary algorithm and soft computing method for predicting the reliability of the software even before entering the testing phase. Studies have shown that machine learning gives remarkable results and among various machine learning techniques, the Adaptive Neuro-Fuzzy Inference System outperforms others [28]. Adaptive Neuro-Fuzzy Inference System has been used for training the system that can predict the output for a given set of inputs. A fault usually arises because of improper requirement analysis, design or some coding error apart from some dynamic fault that may occur because of an environment. The relationship between faults and design metrics are well established. As reliability is a probability of fault-free execution of a code, thus the prediction of reliability of a product is the likelihood of implementation of software without the occurrence of a fault.

The software development has undergone a complete paradigm shift -- from functional to aspect-oriented software. The design metrics used to measure the quality of software depends on the programming paradigm selected for development. By selecting suitable design metrics as input one can predict the (i) fault present in the software, (ii) possibly identify the residual fault, (iii) estimate the development cost, (iv) estimate time required for release, and (v) plan resource utilization. In this paper, a service-oriented software system is considered for the prediction of reliability. A service-oriented system is made of loosely coupled components that can be suitably used in a distributed environment. Service-Oriented Architecture (SOA) is a very popular paradigm where the services are provided to the other component through a communication network by a set of protocol using application component. As compared to the component-based system, here the components are much loosely coupled. The key feature of the component-based software system is that it considers each individual component as a software package. These components are put together so that each component

can communicate with another via their interfaces. To predict system reliability, one needs to understand the reliability of each individual component and their interconnected mechanisms. For Service-oriented architecture, the platform dependency and inter-operability also play a significant role as it works for distributed architecture. However, the atomic unit of SOA architecture is a component, and failure of a component may lead to failure of the system.

In this paper, a study has been undertaken to predict the reliability of the individual component. The failure pattern of the individual component depends on the system design and coding. Section 2 discusses software design metrics that are used for predicting the fault in current literature. Section 3 defines the proposed framework. In section 4, details of the design metrics considered in this study are discussed. In section 5, Adaptive Neuro-Fuzzy Inference System approach for predicting the reliability is given. Section 6 presents the steps to perform the experiment. In section 7, the results obtained during the experiment are discussed. Conclusion and future implications of the present study are given in section 8.

II. LITERATURE SURVEY

Predicting reliability for a component-based system is an important task. The software can only be reliable if it reuses the component which is reliable. Various authors have taken different parameters to predict the reliability of a component-based system. These parameters are categorized into:

(i) Prediction of Reliability based on faults
(ii) Prediction of Reliability based on design metrics (Coupling, Cohesion).
(iii) Prediction of Reliability based on the complexity of the software system.

Various authors have considered fault as a critical dimension to predict the reliability of a component-based system. Some of them are discussed here. R. Sehgal et al. [1] predicts the faults in a component-based system using the Halstead Software Science. Faults can cause more damage at the testing stage, so they should be identified before the testing phase for making the system more reliable. C. Catal and B. Diri [2] propose an approach based on artificial immune system and semi-supervised learning to predict the fault from the system. JM1, KC2, PC1 and CM dataset have been taken from promise repository to introduce a new model for fault prediction. Y. Jianget al. [3] extracted the metrics from the unstructured textual requirement and developed a new model for fault prediction. These metrics provide better project management by finding the faults in the early life cycle data. S. Kanmaniet al. [4] used the design metrics of a software system and proposed a fault prediction model based on neural network. This model identifies the faults which may cause failure in the system. F. Fioravanti and P. Nesi [5] predict the fault proneness in object-oriented system by proposing a new approach, whose measures are coupling and cohesion. Z. Xu et al.[6] predicts the reliability for a dynamic system by proposing a new approach called Monte Carlo Strategy. A. Mahaweerawat et al. [7] proposed a new method to predict the fault using the back propagation neural network technique. Polymorphism and inheritance in an object-oriented system are two major reasons to introduce the faults in the system.

Many researchers find that Coupling is one of the most important parameters to introduce faults in the system. Higher the coupling, more the chances of fault induction, and lower the reliability of the system. K.Emam et al. [8] explored that coupling metrics is most associated with faults. More coupled the system, higher the probability of occurrence of faults. M. Mie et al. [9] estimate the quality of the software system by applying two different neural networks, Ward Neural network and General Regression Neural Network (GRNN) on object-oriented metrics like coupling. Object-oriented metrics has a significant impact on faults. D. Poshyvanyk and A. Marcus [10] propose a new metrics to find out the coupling between two classes. This metrics finds more dependencies than structural coupling. I.Chowdhury and M.Zulkermine [11] identified that complexity cohesion and coupling of these object-oriented metrics makes the system more vulnerable. These are the metrics which can be identified at the time of design of the software system, so these metrics can protect a system from becoming vulnerable.

Cohesion has a positive effect on the system. According to many authors, this measure should be available in greater quantity to make the system more reliable. A. Marcus et al. [12] proposed a new approach to cohesion to find out the dependency within the same class. This method is a better predictor of faulty classes. J.Al Dallal [13] proposed a new metrics of Path Connectivity Class Cohesion (PCCC). This metrics is based on the number of the paths between the attribute and method of the class which is a better fault predictor than the previous cohesion metrics. K.K.Aggarwal et al. [14] explored the relationship between object-oriented metrics like coupling and cohesion and inheritance and faults. B. Goeland Y Singh[15] propose a model to find out which object metrics better predicts the faults like Number of classes(NOC)is not a good metrics to predict the fault, Coupling is a good predictor and Cohesion is not a good fault prediction metrics.

As the complexity increases, there is the probability that faults in the software system also get increased. So, measuring software complexity to improve the reliability is a major concern. Some of the authors who have taken Cyclomatic Complexity (CC) as a reliability parameter are discussed here. Y.Jiang et al. [16] have taken the data set from promise repository and faults are predicted to improve the quality of the code by considering the design metrics like Cyclomatic Complexity. N. E. Fenton and N. Ohlsson [17] find that Cyclomatic Complexity is a good predictor of faults in the software module. With the increase in the size of the module, complexity increases and consequently the number of faults. With Cyclomatic Complexity, faults can be found out/predicted before the testing phase which makes good quality software. N. Nagappanand A. Zeller [18] correlates the faults with the complexity of the code. Mc. Cab Cyclomatic Complexity predicts the faults both pre-release and post-release of the software. Y. Shin and L.Williams[19] statistically find the Pearson correlation between the complexity of the code and the vulnerabilities of the code.

Apart from these design metrics, the probability of usage of a component is equally important. A.K Pandey and N.K Goyal

[20] considered operational profile and fault as one of the important parameters of software reliability. A fault can cause failure only if it occurs at the time of operational usage. Faults which do not occur at the time of operational usage may not affect the reliability of the system. H. Koziolek [21] defines operational profile as a measure which describes how a system will be used. It depends on the probability of how frequently a function in a program is called by a user. M. Gittens et al. [22] extend the definition of an operational profile to increase its applicability. A new model, Extended Operational Profile (EOP), is introduced which consists of process profile, structural profile and data profile. C.-G. Bai [23] applies extended Markov Bayesian network to an operational profile to predict the reliability of the system. Discrete failure data is considered for the prediction of reliability. S. Özekici and R. Soyer [24] discussed the optimization of testing with an operational profile. As there are random operations performed by the user on the software, these random sequences are modeled as Markov chain. J. Musa [25] defines operational profile as an indicator of termination of testing that is when reliable software is delivered to the customer.

To combine these factors and to create a model, various Machine Learning Approaches (MLA) are adopted. Among these, Fuzzy Inference System (FIS) is a popular Machine Learning approach which is augmented with Neural Network to develop ANFIS. M. S. Abdel Aziz et al. [27] use two ANFIS units for the location process. The first unit was used to determine the location of the faults while the second unit determined the distance of fault from the beginning. K. Tragi and A.Sharma [28] apply ANFIS approach on four parameters to estimate the reliability of a component-based system, namely following:-

(i) Reusability,
(ii) Component Dependency,
(iii) Application Complexity and
(iv) Operational Profile.

S. H. Aljahdaliand K. A. Buragga[29] have taken four connectionless models and applied the ANFIS approach to predict the reliability. Therefore, different authors have discussed that Faults, Cyclomatic Complexity, Cohesion, and Coupling are important parameters of software reliability. In the next section, a framework is proposed to predict the reliability with the identified parameter using the ANFIS approach.

III. Framework for Proposed Methodology

The proposed framework is intended to develop a system which predicts the reliability for a given set of inputs. Machine Learning Approaches are used to evolve an intelligent system that automatically predicts the system behavior based on a set of rules. Generation of this rule set relies on expert opinion, or is based on observation of the previous data. The set of input parameters that affect the system need to be selected. The output of Machine Learning approaches ultimately depends on the rule set designed by the industry experts. The framework proposed in this paper (Figure1) involves the following steps:

(i) Selection of design metrics that can be considered as input parameters.
(ii) Conversion of the design metrics into fuzzy input using respective membership function.
(iii) Designing a fuzzy inference system by defining rules based on expert opinion
(iv) Train Fuzzy Inference System (FIS) to Adaptive Neuro-Fuzzy Inference System (ANFIS)
(v) Predict the faulty profile of the components.
(vi) Predict the probability that the given software will perform the desired function without an occurrence of a fault, which reflects the reliability of the software.

IV. Selection of Metrics

Good quality software possesses certain properties and specifications which are measured by software metrics. In a software industry, testing is a way to achieve highly reliable and good quality software. Testing is an execution of a program with the intention of finding an error which comes at the later stage of the Software Development Lifecycle; it takes a lot of time and effort. The software design metrics plays an important role in early prediction of the fault. There is some metrics on which the reliability of the software may depend. From the literature survey and expert opinion, the software metrics identified to develop the framework in this paper are Faults, Cohesion, Coupling, and Cyclomatic Complexity. These four identified metrics are represented in the linguistic variable. Thus 34 rules are developed. Although many other metrics are also suggested by different authors, but they are mainly dependent on the development stage. If additional metrics are included, with a wider range of linguistic variable, the number of rules will increase, resulting in more complexity.

A. Fault

The fault is an incorrect step, action or typing mistake done by a developer which discontinues the system to perform its particular task. Reliability and faults possess a close inverse relationship - less the faults, higher the reliability of the system. An occurrence of a fault can happen at any stage of the Software Development Lifecycle from Requirement phase to the Development phase. An execution of fault leads to failure of the system. Early detection of a fault in SDLC makes the system more reliable. Halstead Metrics is one of the metrics which predicts the fault before the system undergoes

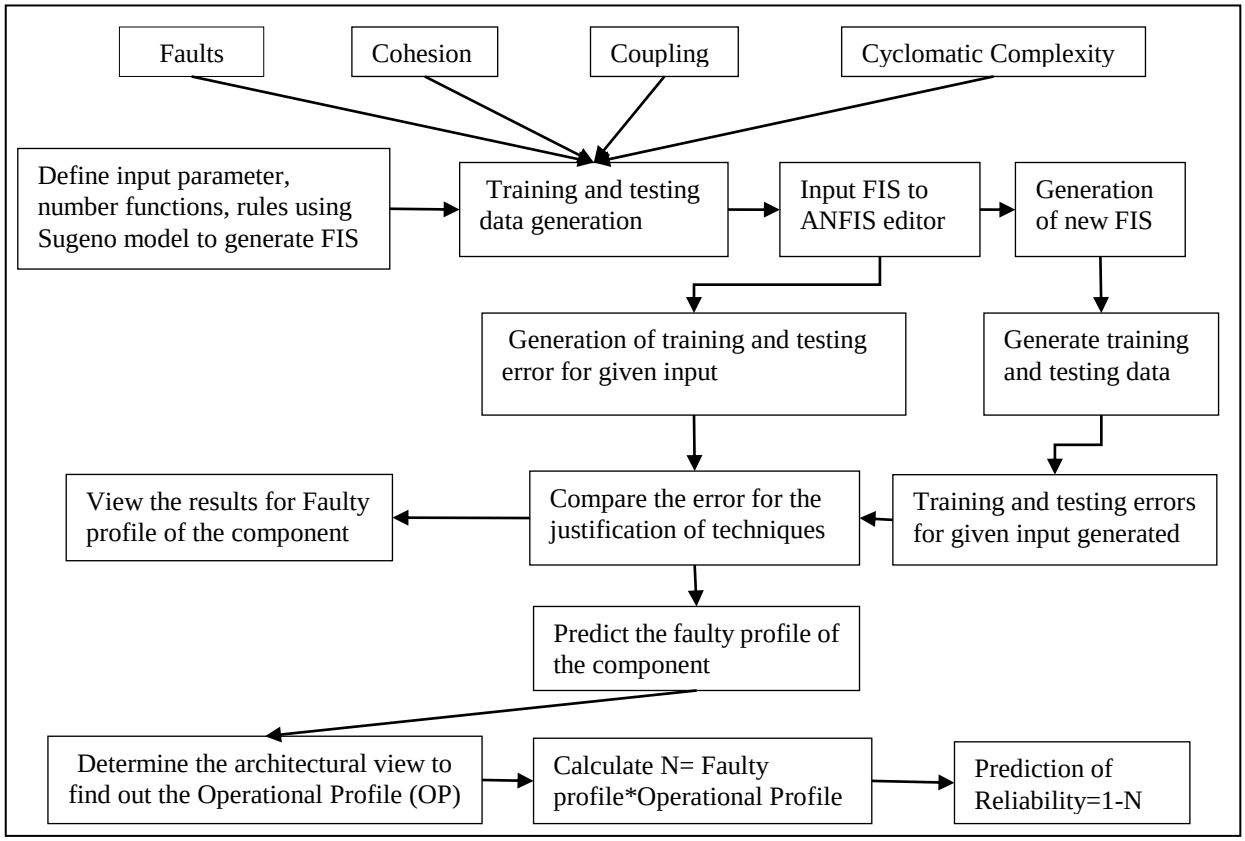


Figure 1 Framework to Predict Reliability

testing. Halstead software science predicts the Fault (Table1) based on the volume of the program, where volume is the number of operator and operand, and their occurrences.

B. Coupling

Coupling is the measure of the degree of interdependence between two modules (Table1). If coupling between two modules is more, then Faults from one module to another module propagate faster, thereby decreasing the overall reliability of the system. Reusability increases for that particular module which possesses less coupling. Faults introduced in a system due to coupling are difficult to remove than the simple faults.

C. Cohesion

Cohesion measures the strength of a module i.e. it shows the interconnection between the elements of a module. An ordinal measure of cohesion is low or high. For good software, cohesion should be high. Reusability is more for that module which is highly cohesive in nature. The main advantage of the highly cohesive modules is that they are easy to maintain. A component with high cohesion value is less faulty and highly reliable which is measured by using Table1.

D. Cyclomatic Complexity

Cyclomatic Complexity measures the complexity of the program. There is a direct correlation between Cyclomatic Complexity and programing errors lower the complex nature of the program, lesser the faults in a program and vice versa. Thus Reliability possesses an inverse relationship with Cyclomatic Complexity. It is measured by developing a control flow graph and calculating its linearly independent path, where nodes represent the number of statements in a program, and an edge represents the flow of data from one account to another. This can be calculated by using the formulae given in Table1 (Equation (1), (2), (3), (4)).The less complex system is easy to maintain and reuse. Operational Profile is another dimension which is also considered in this paper to design the framework for the prediction of the reliability of each and every component. In software reliability, engineering operational profile plays an important role. It is a quantitative description which describes the way a system is in use, J.Musa [25]. The Reliability of any software system depends on different operations performed by the customer. Therefore, a set of activities with their probability of occurrence is known as an operational profile. Reliability of any software system depends on the execution of the faulty component. Components which are more in use should be fault-free to improve the reliability of the system. With a different operational profile, reliability may be different for any software system. In the component-based system, an operational profile is an input given to the component and how the user uses it. Sometimes, a user executes the same program with the same input, but it gives a different output because it is running in a different environment with distinct resources, known as a run type. Thus operational profile is the probability distribution of over-run types, input given to the system in a particular environment, and resource parameters of the software system.

Table 1 Formulae to calculate the values for Identified metrics[31][32]

Metrics	Formula
Halstead Metrics to predict the fault	$F = \frac{V}{S_0}$ (1) where V is the volume of the program which constitutes the number of unique operator and operand.S_0 is number of decisions which is equal to 3000
Coupling	$\frac{\left\|U_{1\le i\le m \cap i=j} NU_{j.i}\right\|}{\left\|N_j\right\| + \left\|U_j\right\|}$ (2) Where N_J and U_J are set of methods and instance variable of component
Cohesion	$COM_i(c) = \frac{\left\|s_i(c)\right\|}{\left\|w_i(c)\right\| * \left\|N_i(c)\right\|}$ (3) Where $N_i(C) \equiv \{n_i1, ni2 \ldots\ldots n_im\}$ are set of method members and w_i (C) is set of instance Variables wi(C) ≡ {wi1, wi2 …win}, si(c) is set of pairs (w_i, N_i) for each instance variable w in W(C)
Cyclomatic Complexity	Cyclomatic Complexity $(CC) = E - N + 2$ (4) E = number of edges of the graph N= number of nodes of the Graph

V. Adaptive Neuro-Fuzzy Inference System Approach

The faulty profile of the given component can be predicted using the Adaptive Neuro-Fuzzy Inference System. ANFIS is a hybrid approach developed by combining Fuzzy Inference System that has a high level of reasoning capability, which is made more robust using the low-level computational power of a neural network. The fuzzy logic which is a multi-defined logic defines the Intermediate value between true and false. Thus it can range from very low, low, high, to very high depending on the scale considered. Using the number of fuzzy rules (IF-THEN), a knowledge base is formed in a fuzzy reasoning system.

Ten experts from different industries were consulted for this purpose, and their inputs were collected as linguistic variables. Formation of rules has been done by considering these expert opinions and to further control the discrepancies between the participants, majority voting approach was adopted. The decision about whether a system is reliable or not is dependent on this rule set. Initially, Fuzzy Inference System structure is generated using the genfis2 function with the help of Matlab Fuzzy Logic Toolbox wherein the input constituted four design metrics namely Faults, Coupling, Cohesion and Cyclomatic complexity. Fuzzy Inference System is trained using Artificial Neural Network and output collected from Adaptive Neuro-Fuzzy Inference System. There are two types of Fuzzy Inference Systems, namely Mamdani type and Sugeno type. In this study, Sugeno Fuzzy Inference System is used to develop the model.

VI. Experimental Setup

In this study, the ANFIS model to predict the reliability is created by using the fuzzy toolbox of MATLAB. A case study considered in this paper is medium-sized software having 23 components which have approx. 25,000 Lines of Code (LOC). Values of identified design metrics like Coupling, Cohesion and Cyclomatic Complexity are calculated using the formulae presented inTable1 which is shown in Table 3. The Software Architect's opinion is taken to convert the design metrics to respective Triangular Fuzzy Number (TFN). To predict the reliability, the following steps are adopted for the implementation of the proposed framework given in Table2.

Table 2 Experimental Setup

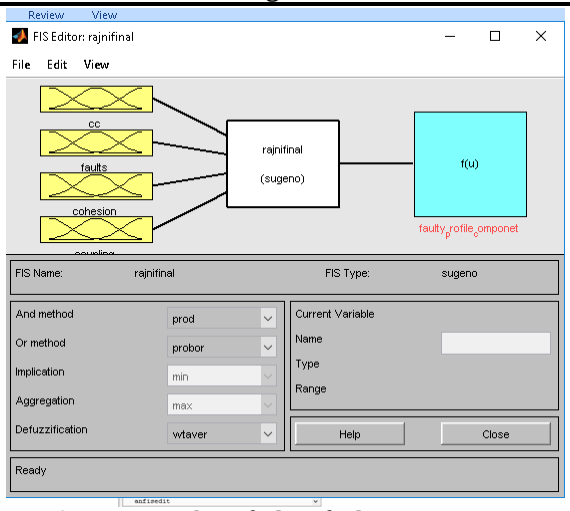
Steps for experimental setup	Figures
(i)The identified parameter is given as an input to Sugeno Fuzzy Inference System. Fuzzy Inference Engine is a system which maps a given input space to an output space using fuzzy logic is given in Figure 2.1. Defuzzification of output variable is done to get the crisp value from the fuzzy set.	

Figure 2.1.*Value of Identified Design Metrics*

(ii)Fuzzy Profile as per the expert opinion of system architect for identified Design Metrics is defined as

Value	Faults	Cylomatic Complexity	Cohesion	Coupling
Low	0,0,2	0,0,50	0,15,30	0,15,30
Medium	1.5,3,4.5	40,70,100	20,40,60	20,45,70
High	4,5,6	90,120,150	55,80,100	60,80,120

Table2.1. Range of each Metric and corresponding linguistic variable

(iii)Membership functions (low, medium, high) are determined for each parameter. The fuzzy set theory is used for generating Membership function that works well with human nature. The triangular membership function is given in Figure 2.2.

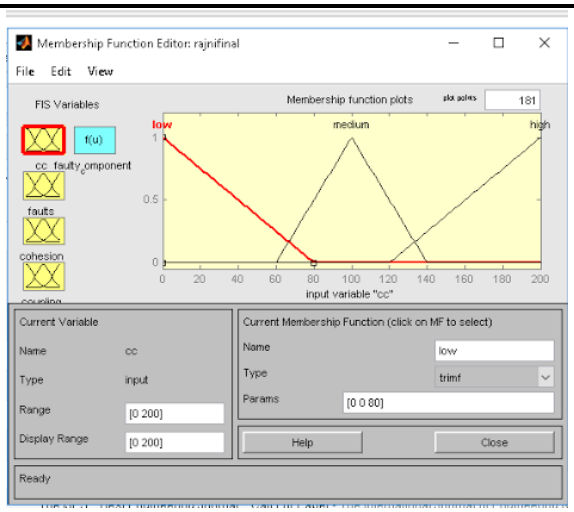


Figure 2.2.*Membership Function*

(v) Input data i.e. training and testing data which is based on fuzzy rules is evaluated with ANFIS editor is shown figure 2.3.

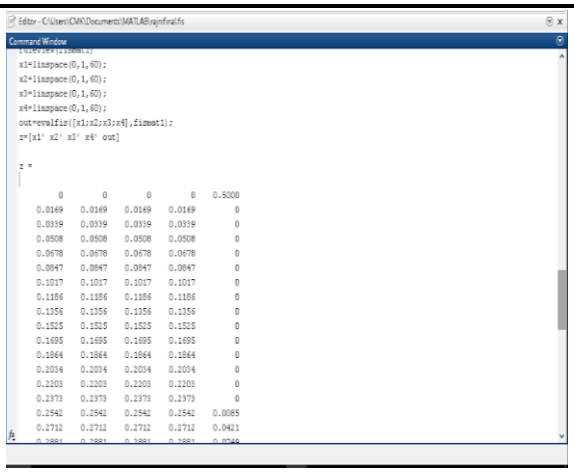


Figure 2.3.*Input data based on rules*

(vi) Evaluation of training and testing error in ANFIS editor is done and given in Figure 2.4. Data generated is used to train and Test the ANFIS. 80% of the data is used for training and rest 20% for testing the network. ANFIS is trained using the data set. Error obtained in results using FIS is 3% while it is 0.0005 % using ANFIS.

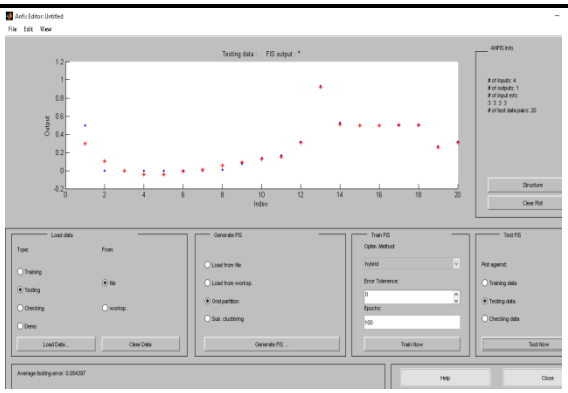


Figure 2.4.*Testing error*

(vii) A new FIS is created based on the training and testing data, which is shown in Figure 2.5

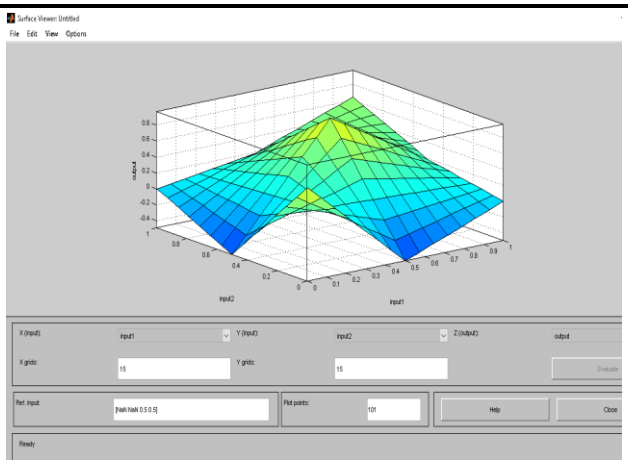


Figure. 2.5 *Surface with new FIS*

(viii) Figure 2.6 give the Faulty profile of the component is evaluated by providing the value of the parameter in ANFIS editor.

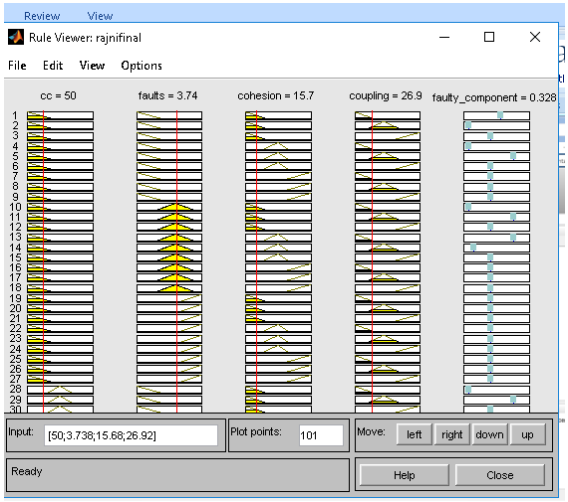
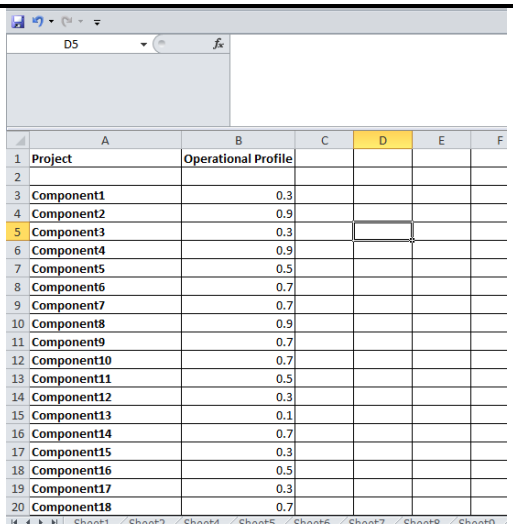


Figure.2.6*Output of ANFIS*

(ix) There is some component in the system which is used more than the other component which is called operational profile (OP). Value of operational profile is provided by the system architect, which is multiplied by the faulty component i.e. ANFIS*OP



	A	B	C	D	E	F
1	**Project**	**Operational Profile**				
2						
3	**Component1**	0.3				
4	**Component2**	0.9				
5	**Component3**	0.3				
6	**Component4**	0.9				
7	**Component5**	0.5				
8	**Component6**	0.7				
9	**Component7**	0.7				
10	**Component8**	0.9				
11	**Component9**	0.7				
12	**Component10**	0.7				
13	**Component11**	0.5				
14	**Component12**	0.3				
15	**Component13**	0.1				
16	**Component14**	0.7				
17	**Component15**	0.3				
18	**Component16**	0.5				
19	**Component17**	0.3				
20	**Component18**	0.7				

Figure. 2.7 *Operational profile: Architect view*

(x) Value of the predicted reliability is calculated with the Equation(5)

$$\text{Predicted Reliability} = 1 - (ANFIS * O.P) \quad (5)$$

(xi) Finally, the value of predicted reliability with ANFIS and FIS is compared to the experimental value. To calculate the experimental value of reliability, error log of six months is created by bugzilla. Mean Time Between Failure (MTBF) is calculated by Equation(6)

$$\text{MTBF} = \frac{\text{Mission time} * \text{Sample Populations}}{\text{Failures within mission time}} \quad (6)$$

Where mission time = time for which the failure is observed
Sample populations=23
Failure= Number of failures observed with in the time frame.
Based on the MTBF reliability is calculated with Equation (7)

$$\text{reliability}=e^{-\text{planned service life}/\text{MTBF}} \quad (7)$$

Where planned service life =is the time for which software system is serviced without any failure

VII. Results

The software system with 23 components is considered for this study. The software metrics where evaluated using formulas given in table 1 (equation (1),(2),(3),(4)). These values are given as input to the ANFIS created in the last section is given in Table 3. Evaluation of these metrics can be done before and after the testing phase. The Fault Profile (FP) of a component is the output obtained from ANFIS which is given in step 7 in Table 2. By considering the relative importance, FP provides the aggregate contribution of all components, but it does not give the crisp value of faults present in the system. Along with faulty profile another important factor that contributes to the reliability of the software is its usage. Not all components are used in the software with the same frequency. The reliability of the software system will fall drastically if components with the high fault profile are more in use. The usage frequency can be predicted using the sequence diagram. The expert opinion of system architects which has considered various UML diagram, is used to predict the probability of usage of the individual component. The input is collected as linguistic variable and represented using the scale of 0 to 1 as given in Table 3. The 0.1 indicates least frequently used, 0.3 also have less used, 0.5 indicates medium usage and 0.7 and 0.9 respectively represent high and very high usage of the component respectively and is given in Figure 2.7. Using step 8 given in Table 2, the probability of occurrence of fault will be a product of Faulty Profile and Operational Profile. The probability that component executes without failure is evaluated equation (5) of step 10. The fault-free execution of component is directly proportionate to the reliability of the software which is presented in the last column of Table 3.

In the given software the ANFIS system predicts that component 17 is highly reliable and four is least reliable. Thus more rigorous testing is required for those components which have less reliability. The component 8 is more in use. Thus the fault-free execution of these components is the prime need of the testing team. Estimation of reliability using equation (7) for each element is performed by taking the error log of six months which is given in Table 4.

Table 3 Value of Design Metrics and predicted Reliability with ANFIS

	Faults	Cyclomatic Complexity	Cohesion	Coupling	Faulty Profile (FP) for Component with ANFIS	Operational Profile	Predicted reliability of component
Component1	3	100	45	50	0.97	0.3	0.70
Component2	1.8	71	34.8	70	0.02	0.9	0.97
Component3	3.7	50	15	26.9	0.32	0.3	0.90
Component4	1.6	71	15	51	0.31	0.9	0.71
Component5	2	19	8	42	0.68	0.5	0.65
Component6	1.1	59	40	65	0.39	0.7	0.72
Component7	0.6	28	23	22	0.05	0.7	0.96
Component8	4	108	60	66	0.45	0.9	0.58
Component9	4	126	29	100	0.24	0.7	0.82
Component10	3	138	40	83	0.55	0.7	0.61
Component11	4.5	77	7.5	10	0.36	0.5	0.81
Component12	2.3	28	43	10	0.60	0.3	0.81
Component13	5.03	105	78	68.5	0.51	0.1	0.94
Component14	1.34	56	18	20	0.21	0.7	0.86
Component15	3	92	34	68	0.36	0.3	0.89
Component16	3	56	15	45	0.68	0.5	0.65
Component17	1.6	37	48	26	0.32	0.3	0.90
Component18	3.74	95	38	33	0.97	0.7	0.32
Component19	1.62	89	29.3	20.3	0.47	0.3	0.85
Component20	3	50	36	22.3	0.37	0.5	0.81
Component21	1	56	17	19	0.29	0.5	0.85
Component22	3.5	141	90	53	0.64	0.5	0.67
Component23	0.785	135	18.4	48.5	0.28	0.5	0.86

Table 4 Comparison of reliability values with different techniques

	Predicted Reliability with ANFIS	Predicted Reliability with FIS	Experimental Value	Error with ANFIS	Error with FIS
Component1	0.70	0.925	0.69	0.01	0.235
Component2	0.97	0.5887	0.75	0.22	0.1613
Component3	0.90	0.8521	0.75	0.15	0.1021
Component4	0.71	0.5572	0.68	0.03	0.1228
Component5	0.65	0.787	0.62	0.03	0.167
Component6	0.72	0.65	0.68	0.04	0.03
Component7	0.96	0.44	0.75	0.21	0.31
Component8	0.58	0.775	0.62	0.04	0.155
Component9	0.82	0.65	0.72	0.1	0.07
Component10	0.61	0.825	0.64	0.03	0.185
Component11	0.81	0.801	0.72	0.09	0.081
Component12	0.81	0.775	0.72	0.09	0.055
Component13	0.94	0.95	0.78	0.16	0.17
Component14	0.86	0.475	0.76	0.1	0.285
Component15	0.89	0.925	0.75	0.14	0.175
Component16	0.65	0.875	0.64	0.01	0.235
Component17	0.90	0.763	0.71	0.19	0.053
Component18	0.32	0.825	0.49	0.17	0.335
Component19	0.85	0.7822	0.74	0.11	0.0422
Component20	0.81	0.625	0.72	0.09	0.095
Component21	0.85	0.625	0.76	0.09	0.135
Component22	0.67	0.75	0.65	0.02	0.1
Component23	0.86	0.827	0.75	0.11	0.077

The SOA architecture is highly dynamic in nature. The proposed Model predicts the fault that exists in the component and does not take care of runtime error. Thus the observed reliability is much lower than the predicted one. Fenton coefficient of determination is used for the validation of this model which tells about what are the chances that future event will fall within the predicted outcome. To check the accuracy of the predicted values, the statistical value R^2 is calculated (Table5) [30] using the Equation (8) and (9) given below

$$R^2_{fis} = 1 - \frac{\sum_{i=1}^{n}(e_i - \hat{e_i})^2}{\sum_{i=1}^{n}(p_{fis} - \overline{p_{fis}})^2} \quad (8)$$

$$R^2_{fis} = 1 - \frac{\sum_{i=1}^{n}(e_i - \hat{e_i})^2}{\sum_{i=1}^{n}(p_{anfis} - \overline{p_{anfis}})^2} \quad (9)$$

Where $\hat{e_i}$ = arithmetic mean value of observed reliability where I =1, 2, 3……n; n€ N

$\overline{P_{fis}}$ = arithmetic mean of predicted reliability values calculated with fuzzy Logic.

$\overline{P_{anfis}}$ = arithmetic mean of predicted reliability values calculated with Adaptive Neuro Fuzzy Inference System.

Table 5 R^2 value

S. No	R^2 Value
1.	R^2_{fis} = 0.77
2.	R^2_{anfis} = 0.80

R^2 is a statistical measure; its value lies between 0 and 1. A value near to one indicates that ANFIS is a better predictor than FIS.

VIII. Conclusion

ANFIS can be suitably used for predicting the reliability. As new technologies come in the market to develop the software, a software architect designs new architecture for it as there is always a need for metrics which ensures the quality of the software. The software architect should identify the metrics that affect the reliability of the software. In the given case study, a Service-oriented architecture is considered, and the reliability of its constituent components is predicted at the early stage i.e. before testing of the software. This output can help in reducing the testing effort, cost and schedule required for software development and maintenance. The given study considers the faults that originate due to faults in the design and coding but does not deal with platform-dependent error, runtime error which includes concurrent and propagating errors.

References

[1] Sehgal, R., Mehrotra, D. "Predicting Faults before Testing Phase using Halstead's Metrics". In *development*, *9*(7) (2015).

[2] Catal, C., Diri, B. "A fault prediction model with limited fault data to improve test process". In *International Conference on Product Focused Software Process Improvement* (pp. 244-257). Springer Berlin Heidelberg (2008, June)..

[3] Jiang, Y., Cukic, B., Menzies, T. Fault prediction using early lifecycle data". In *Software Reliability, 2007. ISSRE'07. The 18th IEEE International Symposium on* (pp. 237-246). IEEE (2007, "November).

[4] Kanmani, S., Uthariaraj, V. R., Sankaranarayanan, V., Thambidurai, P. "Object-oriented software fault prediction using neural networks".*Information and software technology*, *49*(5), 483-492 (2007).

[5] Fioravanti, F., Nesi, P. "A study on fault-proneness detection of object-oriented systems". In *Software Maintenance and Reengineering, 2001. Fifth European Conference on* (pp. 121-130). IEEE.

[6] Xu, Z., Ji, Y., Zhou, D. "A new real-time reliability prediction method for dynamic systems based on on-line fault prediction". *IEEE transactions on reliability*, *58*(3), 523-538 (2009).

[7] Mahaweerawat, A., Sophatsathit, P., Lursinsap, C., Musilek, P. "Fault prediction in object-oriented software using neural network techniques".*Advanced Virtual and Intelligent Computing Center (AVIC), Department of Mathematics, Faculty of Science, Chulalongkorn University, Bangkok, Thailand*, 1-8. (2004)

[8] El Emam, K., Melo, W., Machado, J. C. "The prediction of faulty classes using object-oriented design metrics". *Journal of Systems and Software*, *56*(1), 63-75. (2001)

[9] Thwin, M. M. T., Quah, T. S. "Application of neural networks for software quality prediction using object-oriented metrics". *Journal of systems and software*, *76*(2), 147-156. (2005).

[10] Poshyvanyk, D., Marcus, A. "The conceptual coupling metrics for object-oriented systems". In *Software Maintenance, 2006. ICSM'06. 22nd IEEE International Conference on* (pp. 469-478). IEEE (2006, September).

[11] Chowdhury, I.,Zulkernine, M. "Using complexity, coupling, and cohesion metrics as early indicators of vulnerabilities". *Journal of Systems Architecture*, *57*(3), 294-313 (2011)..

[12] Marcus, A., Poshyvanyk, D., Ferenc, R.. "Using the conceptual cohesion of classes for fault prediction in object-oriented systems". *IEEE Transactions on Software Engineering*, *34*(2), 287-300 (2008).

[13] Al Dallal, J. "Fault prediction and the discriminative powers of connectivity-based object-oriented class cohesion metrics". *Information and Software Technology*, *54*(4), 396-416 (2012).

[14] Aggarwal, K. K., Singh, Y., Kaur, A., Malhotra, R.. "Investigating effect of Design Metrics on Fault Proneness in Object-Oriented Systems".*Journal of Object Technology*, *6*(10), 127-141 (2007).

[15] Goel, B.,Singh. Y, "Empirical investigation of metrics for fault prediction on object-oriented software". In *Computer and Information Science*(pp. 255-265). Springer Berlin Heidelberg (2008).

[16] Jiang, Y.,Cuki, B., Menzies, T., Bartlow, N.. "Comparing design and code metrics for software quality prediction". In *Proceedings of the 4th international workshop on Predictor models in software engineering* (pp. 11-18). ACM (2008, May)

[17] Fenton, N. E., Ohlsson, N. "Quantitative analysis of faults and failures in a complex software system". *IEEE Transactions on Software engineering*, *26*(8), 797-814. (2000).

[18] Nagappan, N., Ball, T., Zeller, A. "Mining metrics to predict component failures". In *Proceedings of the 28th international conference on Software engineering* (pp. 452-461). ACM (2006, May)..

[19] Shin, Y., Williams, L. "Is complexity really the enemy of software security?" In *Proceedings of the 4th ACM workshop on Quality of protection* (pp. 47-50). ACM (2009) (2008, October)..

[20] Pandey, A. K., Goyal, N. K.. *Early Software Reliability Prediction*. Springer, India (2015).

[21] Koziolek, H. "Operational profiles for software reliability". In*Seminar on Dependability Engineering, Germany(pp.1-17), Citeseer* (2005, July).

[22] Gittens, M., Lutfiyya, H., Bauer, M. "An extended operational profile model". In *Software Reliability Engineering, ISSRE, 15th International Symposium on* (pp. 314-325). IEEE (2004, November).

[23] Bai, C. G. "Bayesian network based software reliability prediction with an operational profile". *Journal of Systems and Software*, *77*(2), 103-112 (2005)..

[24] Özekici, S., Soyer, R.. "Reliability of software with an operational profile". *European Journal of Operational Research*, *149*(2), 459-474 (2003).

[25] Musa, J. D.. "The operational profile". In *Reliability and Maintenance of Complex Systems* (pp. 333-344). Springer Berlin Heidelberg (1996).

[26] Musa, J. D. "The operational profile in software reliability engineering: an overview". In *Software Reliability Engineering, Proceedings., Third International Symposium on* (pp. 140-154). IEEE (1992, October)..

[27] Aziz, M. A., Hassan, M. M., Zahab, E. A. "Applications of ANFIS in high impedance faults detection and classification in distribution networks". In *Diagnostics for Electric Machines, Power Electronics & Drives (SDEMPED), IEEE International Symposium on* (pp. 612-619). IEEE (2011, September)..

[28] Tyagi, K., Sharma, A. "An adaptive neuro fuzzy model for estimating the reliability of component-based software systems". in *applied Computing and informatics*, *10*(1), 38-51 (2014).

[29] Aljahdali, S. H., Buragga, K. A. "Employing four ANNs paradigms for software reliability prediction: an analytical study". *ICGST-AIML Journal, ISSN: 1687, 4846* (2008)..

[30] Fenton, N., Neil, M., Marsh, W., Hearty, P., Radliński, Ł., Krause, P. "On the effectiveness of early life cycle defect prediction with Bayesian Nets.". *Empirical Software Engineering*, *13*(5), 499 (2008)..

[31] Chen, J., Wang, H., Zhou, Y., Bruda, S. D.. "Complexity metrics for component-based software

systems". *International Journal of Digital Content Technology and its Applications*, 5(3), 235-244 (2011).

[32] Halstead, M. H. *Elements of software science* (Vol. 7, p. 127). New York: Elsevier (1977).

Author Biographies

Ms. Rajni Sehgal is a postgraduate from Banasthali Vidyapeeth and pursuing Ph.D. in Computer Science and Engineering from Amity University, Noida. Currently she is working as Assistant Professor, Amity School of Engineering and Technology, Amity University, NOIDA, India. . She has more than 10 years of experience in teaching and industry. She had published about 5 papers in national and international conference proceedings and Journal. Her research areas are software engineering and software reliability.

Dr. Deepti Mehrotra is a gold medallist and received PhD from Lucknow University. Currently she is working as Professor, Amity School of Engineering and Technology, Amity University, NOIDA, India. She has more than 18 years of experience in teaching, research and content writing. She had published more than 50 papers in international refereed Journals and conference Proceedings. Her research interests are information security, data mining and knowledge management.

Dr. Manju Bala is currently working as an Assistant Professor at Indraprastha College for Women, University of Delhi in the Department of Computer Science. She graduated from M.D.U., Rohtak followed by post graduation from IASE Deemed University, Rajasthan. She completed her M.Phil from JRN Rajasthan Vidyapeeth Deemed University and Ph.D. from J.N.U., Delhi. She has been associated with Indraprastha College for Women since 2007. She is an active researcher in the field of Computer Science. Her Research Area is Pattern recognition and Classification, Data Mining and Soft Computing Techniques