OPTIMAL ESTIMATION OF SOFTWARE RELIABILITY WITH NEURAL FUZZY MODEL

Rashmita Pradhan

Research Scholar, Mewar University

ABSTRACT

Software reliability is defined as the probability of software to deliver correct service over a period of time under a specified environment. This is becoming more and more important in various software organizations to discover the faults that occur commonly during development process. As the demand of the software application programs increases the quality becomes higher and higher and the reliability of these software becomes more essential. Hence Software reliability is mentioned to be as the one of the important factor during development. Many analytical models were being proposed over the years for assessing the reliability of a software system and for modeling the growth trends of software reliability with different capabilities of prediction at different testing phases. A Neuro Fuzzy based software reliability (SR) model is presented to estimate and assess the quality. Multiple datasets containing software failures are applied to the proposed model. These datasets are obtained from several software projects. Then it is observed that the results obtained indicate a significant improvement in performance by using neural fuzzy model over conventional statistical models (Fuzzy Model) based on non-homogeneous Poisson process.

1. INTRODUCTION

1.1 Background

Dependency on computer aided systems is increasing rapidly day by day and the software systems operating in it .However this quality of service by the system is degraded by some software failures or fails to meet the required level of performance this make many of the people to strike off these software. This model attempt to match product properties with the software quality attributes. Hence if a company is to develop high quality software, it is important to employ some efforts on software reliability and usability. However, this thesis focuses only on software reliability based models.

1.2 Software Reliability

The American Institute of Aeronautics and Astronautics (AIAA) defines SRE as "the application of statistical techniques to data collected during system development and operation to specify, predict, estimate, and assess the reliability of software-based systems"[8].Three kinds of identifiers for Software Reliability. They are a) Probability of failure free operation over a specified time interval. b) Mean time to failure (MTTF) the predicted elapsed time between inherent failures of a system during operation. c) Expected number of failures per unit time interval termed failure intensity.

Here in our work, a Neuro Fuzzy based SRGM is proposed. In order to test the accuracy of proposed model, real failure data of a software project is required. However, it is a very time consuming process to carryout software testing for a real project and could even take years. This is not feasible within the available time and thus secondary data which have already been collected and published.

1.3 Neuro Fuzzy Models

The idea of a Neuro Fuzzy system is to find the parameters of a fuzzy system by means of learning methods obtained from neural networks used to train the system. They cannot be applied directly to a fuzzy system, because the functions used in the inference process are usually not differentiable. There are two solutions to this problem: a) Replace the functions used in the fuzzy system (like min and max) by differentiable functions, or b) Do not use a gradient-based

neural learning algorithm but a better-suited procedure.

1.4 Objectives

The following objectives are set in our research:

- To collect the set of dataset program from running software with the appropriate runtime errors that is useful for the assessment.
- To formulate a theoretical analysis for the evaluation of the metrics those are used for assessment and develop model.
- To identify how availability and MTBF relates with the software reliability
- Calculate the metrics with the given dataset both analytically and programmatically.
- To train the neural network with some collected software reliability parameters (at design phase of SDLC) mapped to numerical data and are loaded into neural network at input layer.
- Assess and evaluate the performance of the trained network for software reliability at the design level with some numerically approximated values by using fuzzy membership function (sigmoid).
- The approximated Software Reliability is compared against the expected reliability approximation.
- The Neuro Fuzzy model was to adjust at the input layer has to minimize the difference between actual and expected values of reliability.
- Our proposed model performance compared against conventional FIS (Fuzzy Inference system) models based

on evaluation and validation metrics to prove that our proposed model is the promising one than the others.

2. SOFTWARE RELIABILITY ASSESSMENT USING NEURO FUZZY SYSTEM

2.1 Proposed Model
Figure below shows the proposed model of the research analysis, where the parameters concerned with the reliability assessment were given as the inputs for network. Based upon the outcome of validation the assessment will be finalized. A generalized block diagram is shown in the below figure, the parameters used were discussed in the next chapter. In this research work a Neuro Fuzzy interference model is designed for the assessment of reliability of a software growth model, the algorithm mainly focuses on MTBF and Availability which is analyzed and calculated theoretically and practically. Fuzzy rules employed for the proposed model

- If MTBF (Mean time Between Failure) >0.8 & availability >0.8 then reliability is very high
- If $0.7<$MTBF <0.8 & $0.7<$ availability <0.8 then reliability is high
- If $0.6<$MTBF <0.7 & $0.6<$availability <0.7 then reliability is moderate
- If $0.5<$MTBF <0.6 & $0.5<$ availability <0.6 then reliability is low
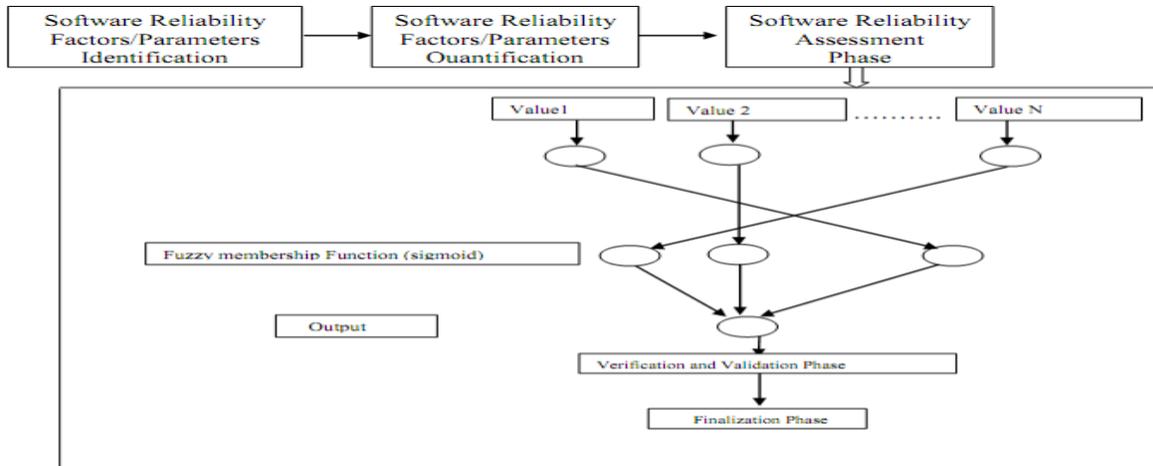- If $0.4<$MTBF <0.3 & $0.4<$availability <0.3 then reliability is very low

Figure 3: Proposed Model of software reliability estimation

2.2 Mathematical approximation of proposal model metrics

Formula: $Ca(x_i) = C(a) - h\,f(a)$
Where, C (a) = Set of Measured values. 'h' can be derived by, x1 + x0 n h
Where, n= no. of values in the dataset.   x0 = 0 and x1 = 1 (since the probability ranges from 0 to 1). Here 'x' is MTBF. f(a) can be function, denoted as f(a)=MTBF/(1+MTBF)
$Ca(x_i)$ is the set of values to be approximated.

Figure 4: Process flow of the proposed approach

Recognition:
At this stage the objectives are
1. To identify the reliability factors.
2. To evaluate a mathematical analysis for the approximation constraints

Correlation:
At this stage the objectives are
1. To identify the reliability factors with availability and MTBF

Procedure for 'h' Calculation: Let us take,   x0 = 0 and x1 = 1 then, 1= 0 + 17 * h
h= 1/17 = 0.058
Iterations: Perform at least 5 to 10 iterations to arrive at good approximated software reliability value. At every iteration, to calculate % of Reliability, use the following formula % of Reliability = (Average of Approximated values)/ (Average of Measured values) * 100 At final iteration, if we got 99.99% or 99.8% or 99.7%, then we can say that it is good approximation.
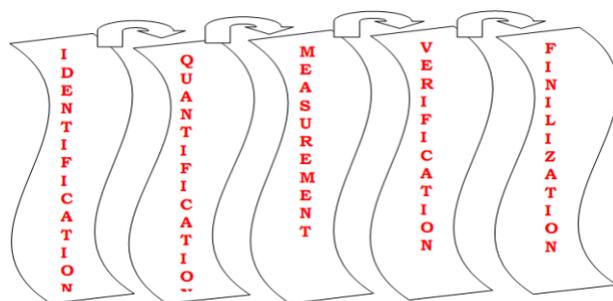
2.3 Review & Revisions
2. To evaluate a mathematical analysis for the relationship

Measurement:
At this stage the objectives are
1. To assess the metrics for the estimation of reliability
2. To validate the metrics

Reliability:
At this stage the objectives are
1. To estimate reliability
2. To validate the reliability

Finalization:
At this stage the objectives are
1. To incorporate the changes and suggestions
2. To finalize the metrics for evaluation

3. EMPIRICAL VALIDATION

3.1 INTRODUCTION

Below figure the practical implementation of the FIS model in MATLAB software tool using FIS. The NF system is trained using a hybrid learning algorithm using both least squares method and back propagation algorithm. In the forward pass the consequent parameters are identified using least squares and in the backward pass the
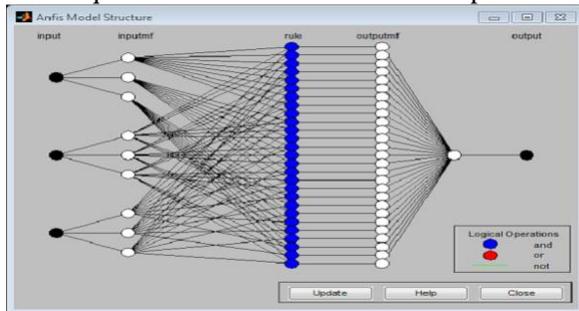

Figure 5: Real time design of Neuro Fuzzy structure

3.2 Dataset

To validate our model, we had taken 17 programs of Glace EMR Medical Billing Software and find out the MTTF (Mean Time to Failure), MTTR (Mean Time to Repair) and MTBR (Mean Time Between Repair) and Software Reliability Approximated value based on the program execution observations. We input these 3 values as input to input layer of Neural Network and apply sigmoid fuzzy membership function at the hidden layer of neural network and try to find out the software reliability approximated value. The previous values assessed using conventional traditional software reliability growth models and our Neuro Fuzzy systems based model are compared and we found to be our model is the promising one. Software reliability is measured in terms of mean time between failures(MTBF).MTBF consists of mean time to failure (MTTF) and mean time to repair(MTTR). MTTF is the difference of time between two consecutive failures and MTTR is the time required to fix the failure. Let us take Software Reliability for good software is a number between 0 and 1. Reliability increases when errors or bugs from the program are removed or minimized. For example, if MTBF = 1000 hours for average software, then the software should work for 1000 hours for continuous operations. The dataset contains failure observations of 17 programs in

premise parameters are identified using back propagation [12]. The trained NF system is then tested for the fifteen inputs and it shows 0.1571, 0.2140 as NRMSE, RMSE (equations can be found in parameters to be evaluated section) values respectively. The plot of the expected and the output of the NF system for the different inputs are shown.
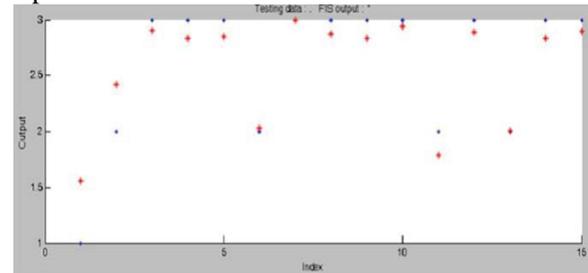

Figure 6: Test Data Vs FIS Output

Glace EMR Billing Software, in time series (i, Xi) and is used to predict the performance of the proposed model. Where, i = Program serial number.

3.3 Parameters used for Validation

Software Reliability: Software reliability is measured in terms of mean time between failures (MTBF).MTBF consists of mean time to failure (MTTF) and mean time to repair (MTTR). MTTF is the difference of time between two consecutive failures and MTTR is the time required to fix the failure. Let us take Software Reliability for good software is a number between 0 and 1. Reliability increases when errors or bugs from the program are removed or minimized. For example, if MTBF = 1000 hours for average software, then the software should work for 1000 hours for continuous operations.

MTBF = Average time between consecutive software system failures = MTTF+MTTR

MTTR = Average time taken to repair the system after the occurrence of failure. Software Reliability = MTBF / (1+MTBF)

Availability = MTBF/(MTBF+MTTR) , is the likelihood that a software system will work at a given time.

3.4 Experimental Results

3.4.1. Results of validation
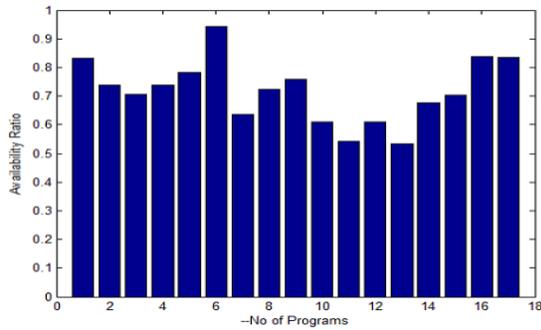We have the following results based on the various parameters

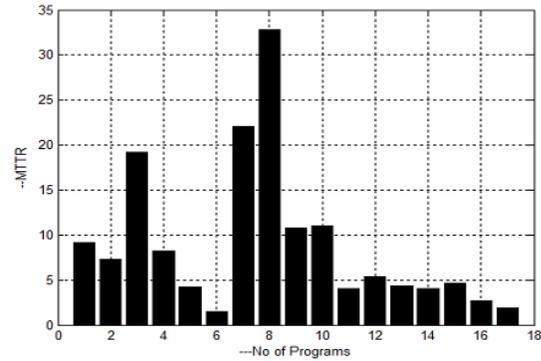*Figure 7: Analysis of Availability Ratio w.r.t. Number of Programs*

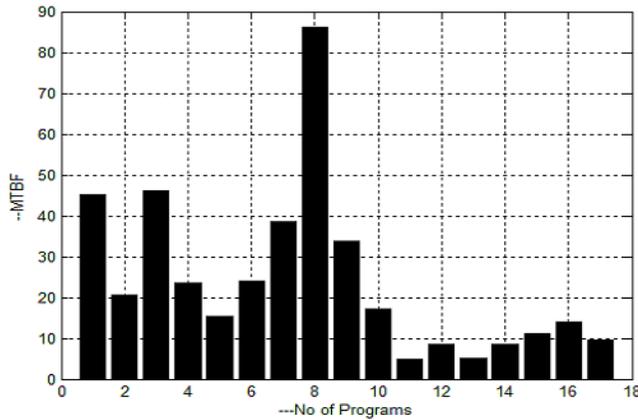*Figure 8: Analysis of MTTR Ratio w.r.t. Number of Programs*

Figure 9: Analysis of MTBF Ratio w.r.t. Number of Programs

3.5 Theoretical Validation
We have performed four iterations and at the end of it we have the following results

At 4th Iteration

% Reliability after 4th iteration = (24.439/ 24.44)*100=99.99

% Reliability= (Average of Approximated vales/ Average of observed Values) x 100

Overall percentage of Reliability= (24.28/ 24.44)*100=99.70

In 4th iteration, we got 99.99%, so we stop iteration process because we got good approximated % of reliability.

3.6 Practical Validation

The experiment was conducted with 17 programs of Glace EMR Medical Billing the analysis was done using FIS (fuzzy interference system) and the proposed Neuro Fuzzy model. The model structure and error tolerance graphs are depicted below.
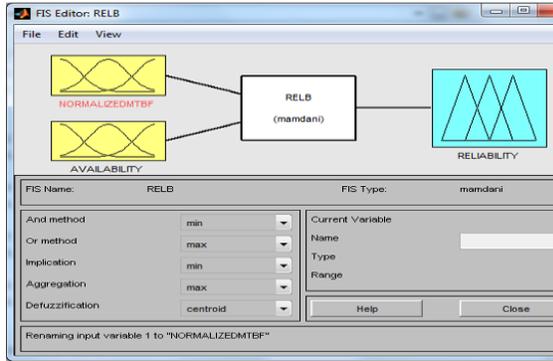
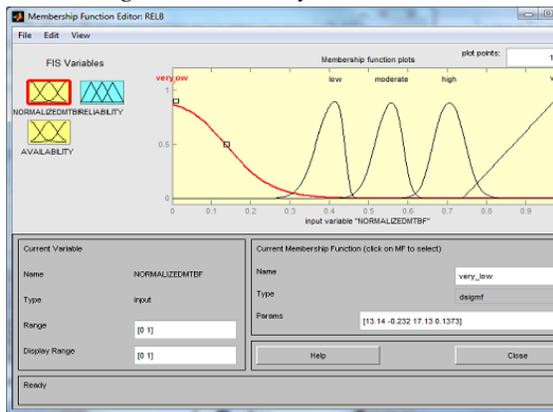*Figure 4.7: FIS System Model*

Figure 4.8: Membership Function for MTBF and Availability
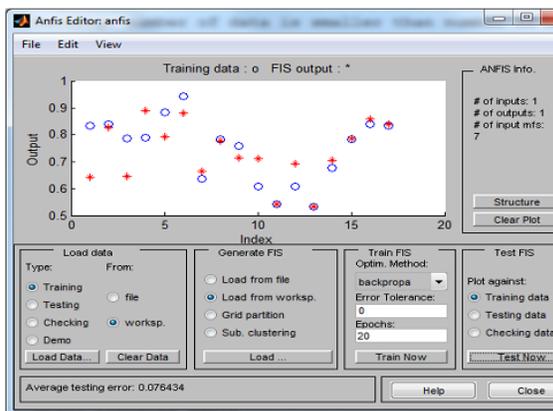

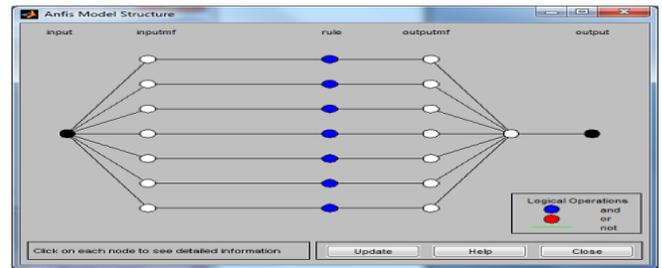*Figure 4.12: Performance Analysis*

Figure 4.9: Neuro Fuzzy Inference Model
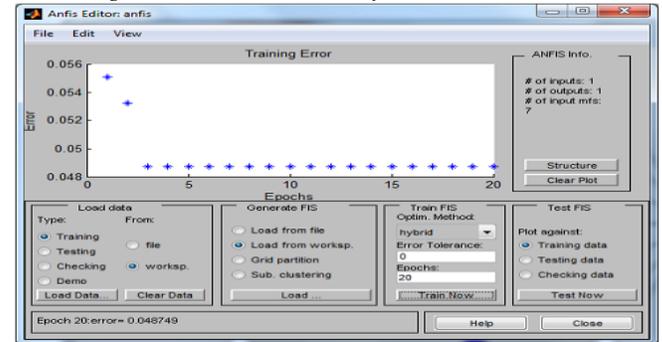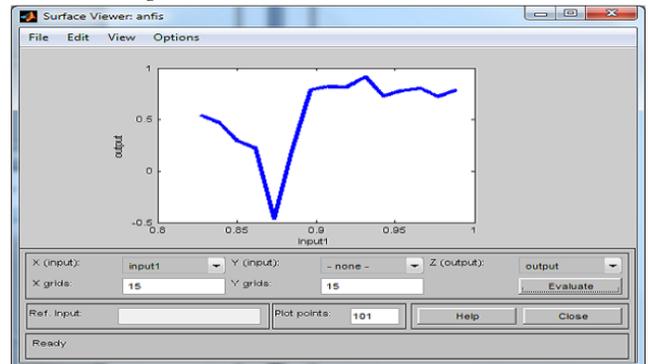

Figure 4.10: Neuro Fuzzy Structure

Figure 4.11: Error Tolerance

3.7. Comparison of proposed approach with Conventional Fuzzy system

The inputs to the Neuro Fuzzy system are Normalized MTBF and availability which is show in the figure 23. The outcome of the conventional system is 84.5 % and the proposed approach is 95.5 % of reliability which is evaluated with MATLAB software tool, when we run the program in MATLAB environment(See Appendix-A). From the above the performance assessment for which an improvement of 11% is achieved with the current proposal.

MSE= ((Theoretical validation- practical Validation)/total no of readings)^2

= ((99.7-95.5)./17).^2=0.061038

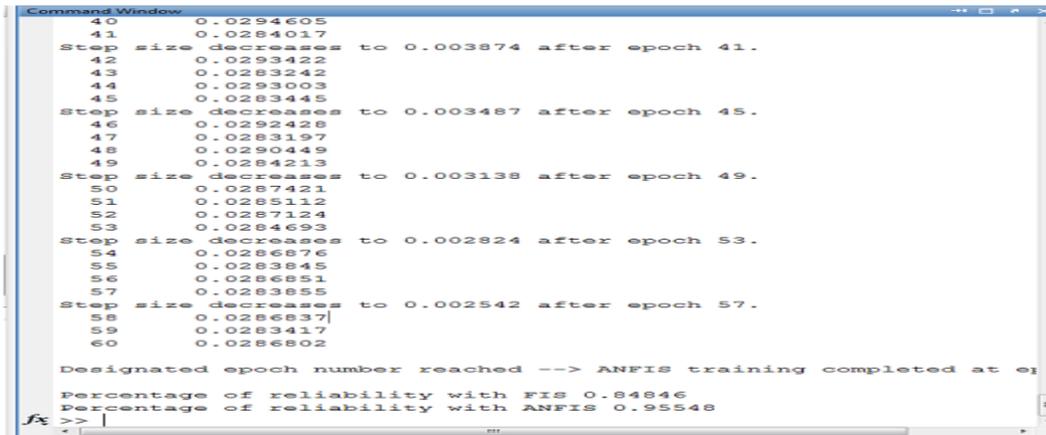**Average error = ((Theoretical validation- practical Validation)/total no of readings)=0.247;**

Figure 4.13: Practical Validation of the Reliability Percentage obtained using MATLAB

Table 4.11: Performance Comparison between FIS & ANFIS of SR Estimation

Method	MSE	AE
FIS	0.799	0.894
ANFIS	0.061	0.247

5. CONCLUSION

From the research we found that Neuro Fuzzy model performs better in terms of less error in prediction as compared to existing analytical models and hence it is a better alternative to do software reliability test. As the weights are randomly initialized, thus the model gives different results for the same datasets and thus the performance of the model varies. The usefulness of a Neuro Fuzzy model is dependent on the nature of dataset up to a greater extent. The preliminary computational results in the MATLAB environment seem quite promising and give insight into the generalization capability of these models. The results of the fuzzy logic and neural networks

models were very promising. The error difference between the actual and estimated response was small. This finding gives a good indication of prediction capabilities of the developed fuzzy model and neural networks for assessing the software reliability.

REFERENCES

[1] Hoyer, R. W. and Hoyer, B. B. Y., "What is quality?" Quality Progress, no. 7, pp. 52-62, 2001.

[2] Crosby P.B," Quality is Free the Art of Making Quality Certain NY, McGraw –Hill-1979.

[3] Deming & Edwards W,"Out of Crisis", MIT Press-1986

[4] Feigenbaum A. V,"Total Quality Control", McGraw – Hill-1983

[5] IEEE Std 610.12 (1990). IEEE Standard Glossary of Software Engineering Terminology, NY.

[6] ISO/IEC Std 9126-1," Software Engineering – Product Quality ", Part 1: Quality Models, International Organization for Standards 2001.

[7] McCall J.A, Richards P.K & Walters G.F," Factors in Software Quality", Nat'1 Tech Information Services, - Vol 1-2, 1977

[8] AIAA/ANSI "Recommended Practice for Software Reliability, the American Institute of Aeronautics and Astronautics", Washington DC, Aerospace Center, R-013, 1992,-ISBN 1-56347-024-1.

[9] Michael R. Lyu," Handbook of software Reliability Engineering ", IEEE

[10] Wasserman, Gary . Reliability verification, Testing and analysis in engineering design. Marcel Dekker incorporated, Newyork, USA, pp2-10, 2002

[11] Schneidewind, N. F., 2001, „Modeling the fault correction processes, Proceedings of the 12th International Symposium on Software Reliability Engineering, pp. 185-190.

[12] Myrtveit, I., Stensrud, E. and Shepperd, M., 2005, „Reliability and validity in comparative studies of software prediction models", IEEE Transactions on Software Engineering, vol. 31, no. 5, pp. 380-391.

[13] Falcone, G. Hierachy-aware Software Metrics in Component Composition Hierarchies. PhD thesis, University of Mannheim.-2010

[14] Bass, L., Clements, P., and Kazman, R. "Software Architecture in Practice" , Second Edition. Addison-Wesley, Reading, MA, USA-2003

[15] Jain, R.   The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling. Wiley.-1991

[16] Smith, C. U. and Williams, L. G. (2000). Software performance anti patterns. In Workshop on Software and Performance, pages 127-136.

[17] Immonen, A. and Niemel• a, E.," Survey of reliability and availability prediction methods from the viewpoint of software architecture. Software and System Modeling-7(1):49{65)-2008

[18] AIAA/ANSI "Recommended Practice for Software Reliability, The American Institute of Aeronautics and Astronautics", Washington DC, Aerospace Center, R-013, 1992,-ISBN 1-56347-024-1.

[19] Rosenberg Linda, Hammer Ted & Shaw Jack, "Software Matrices and Reliability", ISSRE, 1998.

[20] N. E. Fenton and M. Neil. A critique of software defect prediction models. IEEE Transactions on Software Engineering, 25(5):675–689, 1999.