Journal of Software Engineering and Applications, 2014, 7, 581-591
Published Online June 2014 in SciRes. http://www.scirp.org/journal/jsea
http://dx.doi.org/10.4236/jsea.2014.77054

Reliability Estimation of Services Oriented Systems Using Adaptive Neuro Fuzzy Inference System

Ashish Seth[1], Himanshu Agarwal[2], Ashim Raj Singla[3]

[1]Department of Computer Science, Punjabi University, Patiala, India
[2]University College of Engineering, Punjabi University, Patiala, India
[3]Department of Information Technology, Indian Institute of Foreign Trade, New Delhi, India
Email: ashish_may13@rediffmial.com, himagrawal@rediffmail.com, arsingla@iift.ac.in

Received 1 May 2014; revised 29 May 2014; accepted 5 June 2014

Abstract

In order to make system reliable, it should inhibit guarantee for basic service, data flow, composition of services, and the complete workflow. In service-oriented architecture (SOA), the entire software system consists of an interacting group of autonomous services. Some soft computing approaches have been developed for estimating the reliability of service oriented systems (SOSs). Still much more research is expected to estimate reliability in a better way. In this paper, we proposed SoS reliability based on an adaptive neuro fuzzy inference system (ANFIS) approach. We estimated the reliability based on some defined parameter. Moreover, we compared its performance with a plain FIS (fuzzy inference system) for similar data sets and found the proposed approach gives better reliability estimation.

Keywords

Reliability Estimation, SOA, Fuzzy, Rule-Based, Reliability Model, Soft Computing

1. Introduction

Reliability is one of the most important non-functional requirements for software. Accurately estimating reliability for service oriented system (SOSs) is not possible. Moreover, soft computing techniques can help to solve problems which are uncertain or unpredictable. Many researchers have proposed different approaches to SOS reliability estimation [1]. IEEE 610.12-1990 [2] defines reliability as "The ability of a system or component to perform its required functions under stated conditions for a specified period of time". The primary objective of reliability is to guarantee that the resources managed and used by the system are under control. It also guarantees

How to cite this paper: Seth, A., Agarwal, H. and Singla, A.R. (2014) Reliability Estimation of Services Oriented Systems Using Adaptive Neuro Fuzzy Inference System. *Journal of Software Engineering and Applications*, **7**, 581-591.
http://dx.doi.org/10.4236/jsea.2014.77054

that a user can complete its task with a certain probability when it is invoked.

Software reliability management is defined in IEEE 982.1-1988 [3] as "The process of optimizing the reliability of software through a program that emphasizes software error prevention, fault detection and removal, and use of measurements to maximize reliability in light of project constraints such as resources, schedule, and performance". Thus any reliable system is one that must guarantee and take care of fault prevention, fault tolerance, fault removal, and fault forecasting. The most suitable models for reliability of Service Oriented Architectures (SOAs) are the ones based on architecture.

Although the reliability of SOA systems cannot be completely estimated, we can estimate the reliability to a larger extent by analyzing the SOA characteristics and identifying the corresponding requirements. This paper is the result of work done in continuation to our previous study [4] to estimate the reliability of service oriented systems. We started with the identification of important factors for SOA followed by estimating the reliability of such systems through fuzzy inference system (FIS) using Matlab fuzzy tool box, followed by the present work which we have extended to provide more accurate reliability estimation by using an adaptive neuro fuzzy inference system (ANFIS). The rest of the paper is organized as follows. In Section 1, we discussed the basic definition of SOA, services, fuzzy logic and ANFIS. Section 2 covers the work already done in this area in different research studies. Section 3 discusses the research approach for our work. The experimentation and evaluation results are discussed in Section 4. Finally, the conclusion is drawn in Section 5.

1.1. Service Oriented Architecture

SOA provides a design framework for realizing rapid and low-cost system development and improving total system quality. SOA uses the Web services standards and technologies and is rapidly becoming a standard approach for enterprise information systems. SOA is a architectural software concept whose core working is based on services, a functionality that can perform any specific task and facilitates to support business requirements. In a SOA environment, resources are made available to other participants within the network as independent services that are accessible across the network in a standardized way. Overall, a business centric, SOA approach delivers a number of benefits, which includes the following: reduced time to market, improved business alignment for growth, reduced costs, reduced business risk. Each Service Oriented Architecture plays one or more of three roles as service brokers, service registers and service providers as follows [5]:

- A service provider has to make trade-offs between availability & security. It is a web service responsibility for deciding the type of information exposed;
- Service broker or service register is responsible for making information available to a requestor. A service broker has to decide the amount of information transfer;
- The service requestor or Web service client requests for a service and binds to the service provider in order to call upon one of its Web services.

1.2. Service

Services are loosely coupled, autonomous, and reusable. They have well-defined platform-independent interfaces, and provide access to data, business processes, and infrastructure, ideally in an asynchronous manner, so that they can receive requests from any source, making no assumptions as to the functional correctness of an incoming request. Service is an implementation of a well-defined business functionality that operates independent of the state of any other service defined within the system. It has a well-defined set of interfaces and operates through a pre-defined contract between the client of the service and the service itself, which must be dynamic and flexible to be able to add, remove, or modify services, according to business requirements [4]. Services can be written today without knowing how it will be used in the future and may stand on its own or be part of a larger set of functions that constitute a larger service.

From a dynamic perspective, there are three fundamental concepts that are important to understand: The service must be visible to service providers and consumers; the clear interface for interaction between them is defined; and the real world is affected from interaction between services. These services should be loosely coupled and have minimum interdependency, otherwise they can cause disruptions when any service fails or changes.

1.3. Neural Networks and Fuzzy Logic

Neural Networks (NNs) and fuzzy logic are the two basic elements of soft computing techniques. Fuzzy means

unsure and ambiguous. Fuzzy systems are suitable for approximate reasoning, especially for the system whose mathematical model is hard to derive. Fuzzy logic allows decision making with estimated values under incomplete information. A fuzzy set is a generalization of an ordinary set by allowing a degree (or grade) of membership for each element. The membership-function m(x) of a set maps each element to its degree. A membership degree is a real number on [0, 1]. In extreme cases, if the degree is 0 the element does not belong to the set, and if 1 the element belongs 100% to the set.

Neural networks are a form of multiprocessor computer system, with simple processing elements, a high degree of interconnection, adaptive interaction between elements; it is also referred as an "artificial" neural network (ANN). According to Dr. Robert Hecht-Nielsen, a neural network is "...a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs". There are many different kinds of learning rules used by neural networks. ANNs can learn from data and feedback and have learning capabilities. On the other hand, fuzzy logic models are rule-based models and do not have learning capabilities, therefore so for learning, fuzzy inference system performs the following operations:

- fuzzification of the input variables;
- determination of membership functions for the parameters;
- application of the fuzzy operator in the antecedent;
- implication from the antecedent to the consequent;
- defuzzification.

1.4. Adaptive Neuro Fuzzy Inference System (ANFIS)

ANFIS was first defined by J.-S. Roger Jang in 1992. It is a techniques to learn about a data set, in order to compute the membership function parameters that best allow the associated fuzzy inference system to track the given input/output data. The toolbox function "*anfis*" constructs a fuzzy inference system (FIS) using a given input/output data set, for which membership function parameters are tuned (adjusted) using either a back propagation algorithm alone or in combination with a least squares type of method. ANFIS has the following advantages over an FIS as follows:

- Through learning algorithms, an ANFIS can optimize the parameters of a given FIS by simulating and analyzing the mapping relation between input and output data;
- An ANFIS has networks which involve nodes and directional links, along with some learning rules are also associated with these networks whereas an FIS has no network link and its behavior only depends on its membership functions;
- Learning method in ANFIS is much similar to that of neural networks whereas FIS has no learning capability.

2. Related Work

Most of the research on software reliability engineering focuses on system testing and system-level reliability growth models However, SOA is not taken into account in these approaches. Although there are some soft computing approaches have been developed for estimating the reliability of service oriented systems (SOSs). Goseva-Popstojanova, *et al*. (2001) and Gokhale (2007) did remarkable work for architecture-based empirical software reliability analysis in relation to architecture-based empirical software reliability analyses [6] [7]. Significant work done in the direction of estimating reliability of SOA is summarized below:

Danilecki, A., *et al*. [8] (2011) proposed a model named ReServE, which ensures that business processes are consistently perceived by client and services, transparently recovers the state of a business process. When a service fails, its SPU can initiate the rollback-recovery process.

Brosch, F., *et al*. [9], proposed SAMM (2010) to evaluate the impact of different component topologies on the system reliability, author concludes that not only the hardware, but also different allocation configuration have influence on the reliability prediction.

Zibin, Z., *et al*. [10] (2010) proposed Collaborative Reliability Prediction of Service-Oriented Systems, in his work collaborative framework is proposed for predicting reliability of service-oriented systems which employs past failure data of similar service users for making reliability prediction for the current service user.

Wang, L., *et al*. [11], introduces unified reliability modeling framework (2009) and concludes service pools as

backup alternative, reliability of simple services is addressed by considering data reliability, authors used Time Markov Chains (DTMCs) are for analyzing reliability of service composition.

Wang, *et al*. [12] proposed Analyzed-stock market system (SMS) (2006), authors mapped to component failure probabilities and predicts the system reliability. In their work they derived transition probabilities from recorded transitions between components.

Tsai, *et al*. [13] proposed SORM (2004), Service-Oriented Software Reliability Model which tries to determine the reliability of each component and their relationship. It consists of two stages: group testing to evaluate the reliability of atomic services; and evaluation of composite services through the analysis of components and their relationships, author used a group testing technique from the medical field to detect faults.

3. Discussion and Research Approach

This work is an extension of our previous work done to identify the SOA adoption trends & implementation factors [14], followed by estimating the reliability of such systems through fuzzy inference system (FIS) using Matlab fuzzy tool box, followed by the present work which we have extended to provide more accurate reliability estimation by using a adaptive neuro fuzzy inference system (ANFIS). The present work is an extension of our previous research which includes three phases as follows:

1) In first phase, thorough review of articles and research has been done and identified the factors that are relevant to SOA implementation and the extent to which each factor is crucial to SOA implementation [4].

2) In second phase using GQM technique, metrics are proposed, and the responses are taken from 125 people in the industry. The data, which is based on the feedback and responses, is defined into the following three parameters [14]:

a) AR: adhoc requirements/dynamic binding/agility;

b) MG: migration/legacy system integration;

c) BI: business and IT collaboration.

The rules were defined for the inference engine. Three clusters were formed for the input factors (Low, Medium, and High), and five clusters were formed for the output reliability (Very Low, Low, Medium, High, and Very High). Therefore, with 3 clusters and 3 input factors, a total of 27 rules were formed that yield $3^3 = 27$ sets. These 27 sets or classifications can be used to form 27 rules using fuzzy model.

3) In third phase (*i.e*. the present work), we followed Sugeno-type inference, defined it for the fuzzy logic toolbox to estimate the reliability of service oriented systems.

3.1. Reliability Parameters for SOA-Based Systems with Its Constituent Factors

1) AR: A system capable of fulfilling the ad hoc on-demand changing requirement of the market is assumed to be efficient and reliable. It is based upon the way the rule engine within the model has been trained to perform dynamic binding whenever the demand changes or arises. This also covers agility, which is the important issue when someone moves from present legacy systems to SOA-based systems. It is further concluded that the more the system has capability to handle dynamic binding/ad hoc requirement/agility, the more system is assumed to be reliable. Therefore, SOA reliability α AR.

2) MG: It is observed that, although an SOA system is strong enough in terms of its capacity to handle the ad hoc market, if there is no provision of integrating the legacy system or migrating successfully from old system to new one within the system; it is not effective and will not guarantee system reliability. Moreover it is observed that mostly small and medium enterprises (SMEs) using the SOA system be developed from services developed from scratch. Since it is concluded that more migrations affects the system reliability. Therefore, SOA reliability α 1/MG.

3) BI: Within a system, if the collaboration between business process and strategies is aligned with IT capabilities, the system is assumed to be more reliable. Through surveys, it has been observed that, although the powerful IT system is there, it will not be of much valuable to the organization without proper integration within the business strategies. Therefore, SOA reliability α BI.

The factors described in the three parameters above assess different properties and characteristics associated with SOA model reliability. The values of these parameters cannot be used independently to measure reliability. Rather, an integrated approach that considers all three parameters and their relative impact is required for estimating a system's overall reliability [14].

3.2. Proposed Approach

1) Conduct a thorough survey of literature to identify the factors that are relevant to SOA implementation and the extent to which each factor is crucial to SOA implementation.
2) Identify reliability parameters in SOA context among these factors.
3) Cluster reliability parameters into three domain clusters of reliability factors.
4) Assemble a database for the value of these factors.
5) Design an inference engine based on the rule for identifying reliability clusters.
6) Using Sugeno system, perform the following operations

- Plot the number of inputs, outputs, input membership functions, and output membership functions.
- Load FIS or generate FIS from loaded data using your chosen number of MFs and rules or fuzzy.
- Train FIS after setting optimization method, error tolerance, and number of epochs. Training adjusts the membership function parameters and plots the training (and/or checking data) error plot(s) in the plot region
- Test data against the FIS model.
- Anticipate the FIS model output versus the training, checking, or testing data output.

We have a training data set that contains desired input/output data pairs of the target system to be modeled. These training and checking data sets are collected based on observations of the target system and are then stored in separate files. It has been observed that only the checking data set is corrupted by noise.

4. Result and Discussion

For present modeling we used the Fuzzy Logic Toolbox neuro-adaptive learning techniques incorporated in the anfis command. The parameters could be chosen so as to tailor the membership functions to the input/output data in order to account for these types of variations in the data values. Our experiments simulated the effect of rules with the MATLAB Fuzzy Logic Toolbox; the reliability for the values obtained is found to be very close to the calculated value, thus result obtained justifies our approach by giving better estimates in comparison to FIS [15]-[17].

For the analysis of result obtained with the experiment we used covariance method to compare the closeness of the value obtained with the experiment with the values collected from original sample data set. Covariance provides a measure of the strength of the correlation between two or more sets of random variants. The covariance for two random variates X and Y, each with sample size N, is defined by the expectation value

$$\operatorname{cov}(X,Y) = \left\langle (X-\mu_x)(Y-\mu_y) \right\rangle \tag{1}$$

$$= \left\langle X\ Y \right\rangle - \mu_x\ \mu_y \tag{2}$$

where $\mu_x = \langle X \rangle$ and $\mu_y = \langle Y \rangle$ are the respective means, which can be written out explicitly as

$$\operatorname{cov}(X,Y) = \sum_{i=1}^{N} \frac{(x_i - \bar{x})(y_i - \bar{y})}{N}$$

The comparison table for the ANFIS and original data set is shown in the table (**Table 1**).

Covariance matrix for (Anfis, ori) =

0.0137 0.0135

0.0135 0.0137

Average Testing Error is = 0.021%

Since the covariance is positive we can say that we get results closer to the original values. We generated a plot for test data against FIS, the FIS is trained after setting optimization method, error tolerance and number of epochs. **Figure 1** shows the plot of testing data against FIS, testing data appear on the plot in blue color while the FIS output is shown in red color.

After creating the ANFIS model, we compared the output reliability values for different input sets with the original values. We calculated Average Testing Error for the output obtained by the FIS and the output obtained by the ANFIS with the original output. ANFIS reduces the error to 0.021%. Hence, the ANFIS performs better than the FIS. In ANFIS, we first trained the FIS, on the basis of training data the rules were formed to produce the output of the trained model. We observed during experiments that for large data sets its execution is little complex. Our results show that the ANFIS model gives a more accurate measure of reliability than the FIS

model. **Table 2** illustrates the comparison chart for FIS, ANFIS and original. Similarly graph shown in **Figure 2** indicates ANFIS is closer to original values than FIS.

Table 1. Comparison of original data set with ANFIS using Sugeno method.

Input1	Input2	Input3	Ori	ANFIS
0.0101	0.0101	0.0101	0.4321	0.4514
0.0202	0.0202	0.0202	0.4321	0.4503
0.0303	0.0303	0.0303	0.4231	0.4494
0.0404	0.0404	0.0404	0.4212	0.4486
0.0505	0.0505	0.0505	0.4321	0.448
0.0606	0.0606	0.0606	0.4251	0.4474
0.0707	0.0707	0.0707	0.4421	0.4469
0.0808	0.0808	0.0808	0.4899	0.4465
0.0909	0.0909	0.0909	0.4548	0.4461
0.0101	0.0101	0.0101	0.4748	0.4458
0.1414	0.1414	0.1414	0.4321	0.4447
0.1818	0.1818	0.1818	0.4243	0.444
0.2121	0.2121	0.2121	0.4889	0.4436
0.2525	0.2525	0.2525	0.6321	0.4427
0.2828	0.2828	0.2828	0.4521	0.4501
0.3434	0.3434	0.3434	0.4646	0.4706
0.3636	0.3636	0.3636	0.4436	0.4799
0.4747	0.4747	0.4747	0.5831	0.5932
0.4949	0.4949	0.4949	0.6321	0.6444
0.5455	0.5455	0.5455	0.5591	0.5692
0.5960	0.5960	0.5960	0.4999	0.5218
0.5758	0.5758	0.5758	0.5215	0.5371
0.6768	0.6768	0.6768	0.4652	0.4842
0.6869	0.6869	0.6869	0.4461	0.4811
0.7172	0.7172	0.7172	0.4142	0.473
0.7374	0.7374	0.7374	0.4651	0.4684
0.8081	0.8081	0.8081	0.21	0.2314
0.8283	0.8283	0.8283	2121	0.2222
0.8788	0.8788	0.8788	0.1712	0.1913
0.8990	0.8990	0.8990	0.1744	0.1744
0.9292	0.9292	0.9292	0.1542	0.1533

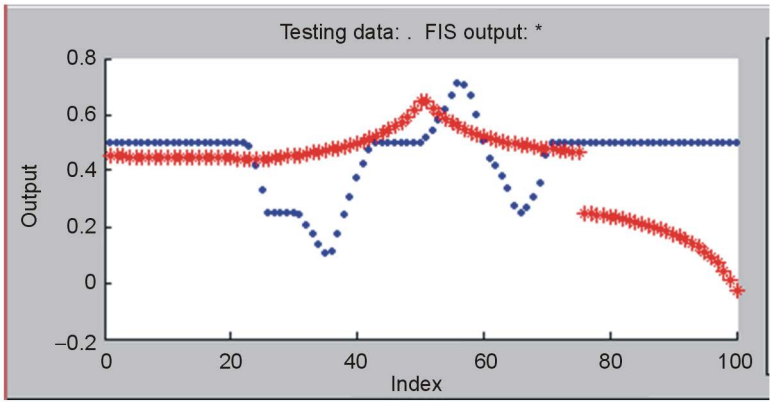


Figure 1. Plot of testing data: Original vs. ANFIS.

Table 2. Plot of FIS, ANFIS and original.

Input1	Input2	Input3	Ori	ANFIS	FIS
0.0101	0.0101	0.0101	0.4321	0.4514	0.4
0.0202	0.0202	0.0202	0.4321	0.4503	0.4
0.0303	0.0303	0.0303	0.4231	0.4494	0.4
0.0404	0.0404	0.0404	0.4212	0.4486	0.4
0.0505	0.0505	0.0505	0.4321	0.448	0.4
0.0606	0.0606	0.0606	0.4251	0.4474	0.4
0.0707	0.0707	0.0707	0.4421	0.4469	0.4
0.0808	0.0808	0.0808	0.4899	0.4465	0.4
0.0909	0.0909	0.0909	0.4548	0.4461	0.4
0.0101	0.0101	0.0101	0.4748	0.4458	0.4
0.1414	0.1414	0.1414	0.4321	0.4447	0.4
0.1818	0.1818	0.1818	0.4243,	0.444	0.4
0.2121	0.2121	0.2121	0.4889	0.4436	0.4163
0.2525	0.2525	0.2525	0.6321	0.4427	0.4397
0.2828	0.2828	0.2828	0.4521	0.4501	0.4621
0.3434	0.3434	0.3434	0.4646	0.4706	0.5163
0.3636	0.3636	0.3636	0.4436	0.4799	0.5378
0.4747	0.4747	0.4747	0.5831	0.5932	0.6801
0.4949	0.4949	0.4949	0.6321	0.6444	0.6961
0.5455	0.5455	0.5455	0.5591	0.5692	0.6795
0.5960	0.5960	0.5960	0.4999	0.5218	0.5395
0.5758	0.5758	0.5758	0.5215	0.5371	0.5821
0.6768	0.6768	0.6768	0.4652	0.4842	0.5894
0.6869	0.6869	0.6869	0.4461	0.4811	0.6116
0.7172	0.7172	0.7172	0.4142	0.473	0.6694
0.7374	0.7374	0.7374	0.4651	0.4684	0.6865
0.8081	0.8081	0.8081	0.21	0.2314	0.1456
0.8283	0.8283	0.8283	2121	0.2222	0.1982
0.8788	0.8788	0.8788	0.1712	0.1913	0.1521
0.8990	0.8990	0.8990	0.1744	0.1744	0.1681
0.9292	0.9292	0.9292	0.1542	0.1533	0.1341

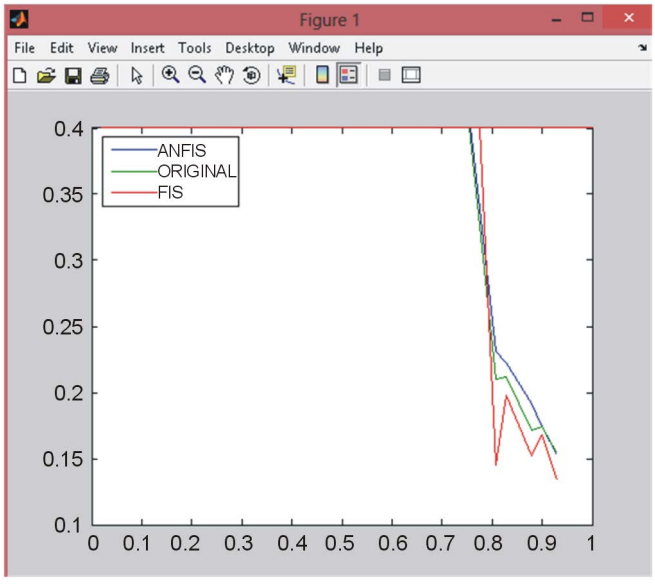


Figure 2. The graph of ANFIS is much close to original values than FIS.

The inference system, inference rules, fuzzy inference system, rule viewer and surface viewer for ANFIS using Sugeno method is shown in appendices.

5. Conclusion and Future Work

This paper proposes a neuro fuzzy approach for estimating the reliability of service oriented systems. Proposed approach is based on an ANFIS that requires less computational time than previously proposed FIS and other traditional approaches. Our results show that the ANFIS give more accurate estimation than FIS. Future scope may be to identify other relevant factors that should be used but currently we only have data available for the discussed factors. Our experience documented in this paper will be helpful for practitioners in collecting the data necessary for reliability prediction. Researchers are provided a demonstration on how the fuzzy logic toolbox can be used to find the reliability of such system on the basis of certain SOA features.

References

[1] Kirti, T. and Arun, S. (2012) A Rule-Based Approach for Estimating the Reliability of Component Based Systems Advances in Engineering Software. Elsevier, Amsterdam, 24-29.

[2] IEEE Standard 610.12-1990 (2014) IEEE Standard Glossary of Software Engineering Terminology. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=159342

[3] IEEE Standard 982.1-1988 (2014) IEEE Standard Dictionary of Measures to Produce Reliable Software. http://www.baskent.edu.tr/~zaktas/courses/Bil573/IEEE_standards/982_1_2005.pdf http://standards.ieee.org/findstds/standard/982.1-1988.html

[4] Ashish, S., Aggarwal, H. and Singla, A. (2012) Service Oriented Architecture Adoption Trends: A Critical Survey at 5th International Conferences on Contemporary Computing. Proceeding in Communications in Computer and Information Science, Springer, Berlin.

[5] Bianco, P., Kotermanski, R. and Merson, P. (2007) Evaluating a Service-Oriented Architecture. Software Engineering Institute. Prepared for The SEI Administrative Agent ESC/XPK, 5, Eglin Street Hanscom AFB.

[6] Gokhale, S.S. (2007) Architecture-Based Software Reliability Analysis: Overview and Limitations. *IEEE Transactions on Dependable and Secure Computing*, **4**, 132-140. http://dx.doi.org/10.1109/TDSC.2007.4

[7] Goseva-Popstojanova, K. and Trivedi, K.S. (2001) Architecture-Based Approach to Reliability Assessment of Software Systems. *Perform Evaluation*, **45**, 179-204. http://dx.doi.org/10.1016/S0166-5316(01)00034-7

[8] Danilecki, A., Holenko, M., Kobusinska, A., Szychowiak, M. and Zierhoffer, P. (2011) ReServE Service: An Approach to Increase Reliability in Service Oriented Systems. *Parallel Computing Technologies*, PaCT 2011, LNCS 6873, 244-256.

[9] Brosch, F., Koziolek, H., Buhnova, B. and Reussner, R. (2010) Parameterized Reliability Prediction for Component-Based Software Architectures. *Proceedings of the* 6*th International Conference on the Quality of Software Architectures* (*QoSA*'10), Springer, New York, 36-51.

[10] Zheng, Z. and Lyu, M.R. (2010) Collaborative Reliability Prediction of Service-Oriented Systems. 2010 *ACM/IEEE* 32*nd International Conference on Software Engineering*, Cape Town, 2-8 May 2010, 35-44.

[11] Wang, W.L., Pan, D. and Chen, M.H. (2006) Architecture-Based Software Reliability Modeling. *Journal of Systems and Software*, **79**, 132-146. http://dx.doi.org/10.1016/j.jss.2005.09.004

[12] Wang, L., Bai, X. and Zhou, L. (2009) A Hierarchical Reliability Model of Service-Based Software System. 33*rd Annual IEEE International Computer Software and Applications Conference*, Seattle, Washington DC, 20-24 July 2009, 199-208.

[13] Tsai, W., Zhang, D., Chen, Y., Huang, H., Paul, R. and Liao, N. (2004) A Software Reliability Model for Web Services. 8*th IASTED International Conference on Software Engineering and Applications*, Cambridge, 8-11 November 2004.

[14] Ashish, S., Aggarwal, H. and Singla, A. (2014) Estimating Reliability of Service-Oriented Systems: A Rule-Based Approach. *International Journal of Innovative Computing, Information and Control. ICIC International*, **10**, 1111-1120.

[15] Gershteyn, Y. and Perman, L. (2003) Matlab: ANFIS Toolbox. http://csrit.gershteyn.net/courses/nn/Presentations/3-MatLab_ANFIS.pdf

[16] Arikan, S. (2012) Automatic Reliability Management in SOA-Based Critical Systems. *European Conference on Service-Oriented and Cloud Computing*, 1-6. http://dspace.icsy.de:12000/dspace/bitstream/123456789/367/1/reliability.pdf

[17] Becker, S. (2008) Coupled Model Transformations for QoS Enabled Component-Based Software Design. Ph.D. Thesis, University of Oldenburg, Oldenburg.

Appendix 1

Inference System: Sugeno

[System]
Name = 'SOARelANFIS2'
Type = 'sugeno'
Version = 2.0
NumInputs = 3
NumOutputs = 1
NumRules = 27
AndMethod = 'min'
OrMethod = 'max'
ImpMethod = 'prod'
AggMethod = 'sum'
DefuzzMethod = 'wtaver'

[Input1]
Name = 'AdhocMgmt'
Range = [0 1]
NumMFs = 3
MF1 = 'ARLow':'trimf', [0 2 4]
MF2 = 'ARMedium':'trimf', [2 4.5 7]
MF3 = 'ARHigh':'trimf', [5 8 10]

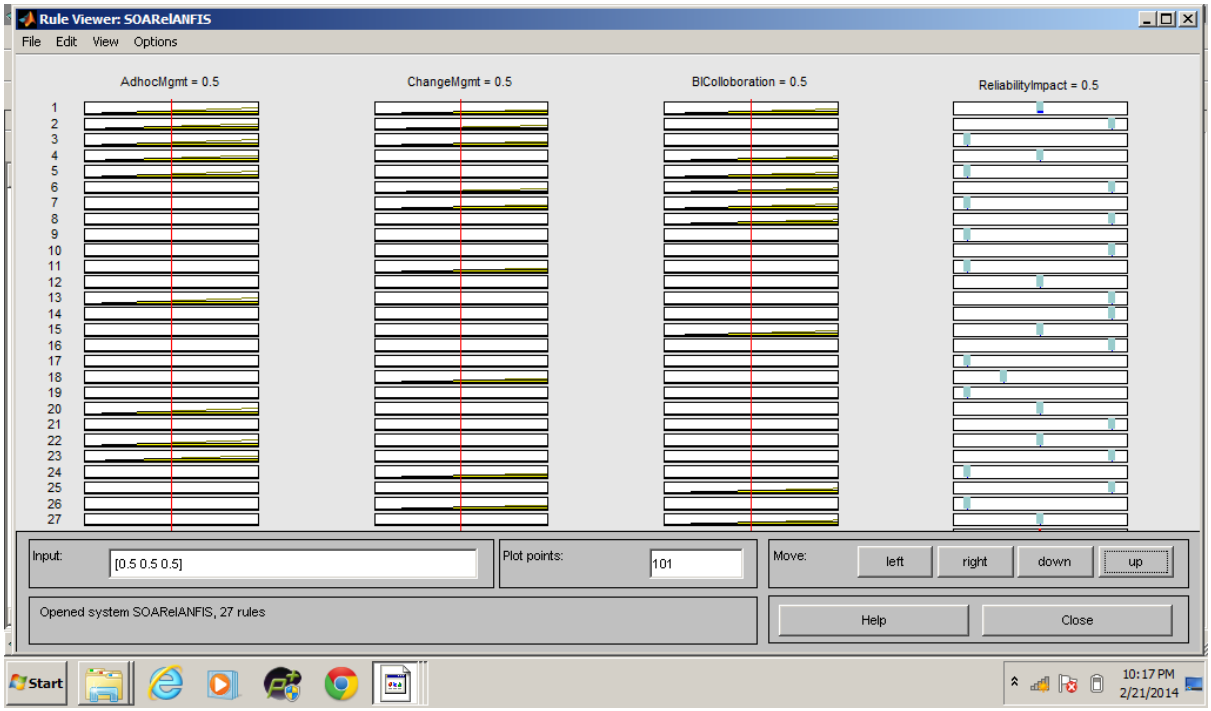
[Input2]
Name = 'ChangeMgmt'
Range = [0 1]
NumMFs = 3
MF1 = 'CMLow':'trimf', [0 3 4.5]
MF2 = 'CMMedium':'trimf', [3 5 7]
MF3 = 'CMHigh':'trimf', [5.5 7.5 10]

[Input3]
Name = 'BIColloboration'
Range = [0 1]
NumMFs = 3
MF1 = 'BILow':'trimf', [0 2.5 5]
MF2 = 'BIMedium':'trimf', [4 6 7.5]
MF3 = 'BIHigh':'trimf', [6 8.5 10]

[Output]
Name = 'ReliabilityImpact'
Range = [0 1]
NumMFs = 5
MF1 = 'RI_VLow':'constant', [0]
MF2 = 'RI_Low':'constant', [0.5]
MF3 = 'RI_Meduim':'constant', [1]
MF4 = 'RI_High':'constant', [0]
MF5 = 'RI_VHigh':'constant', [0.25]

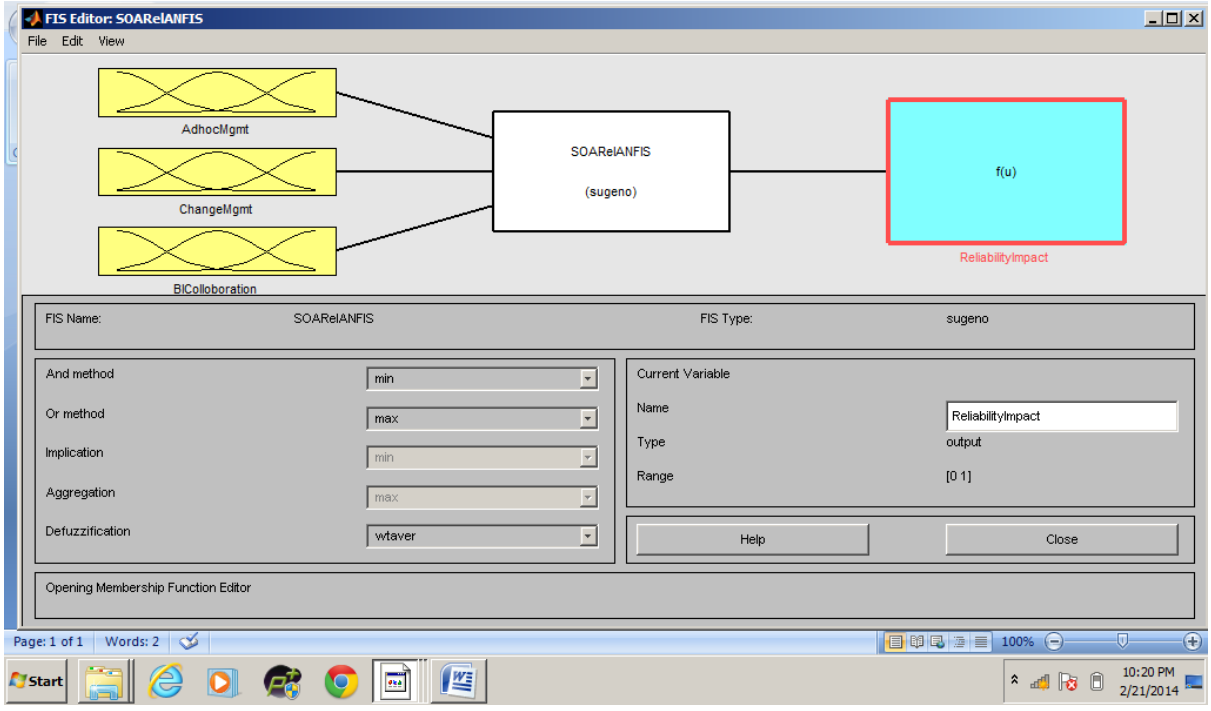
Appendix 2

Rule Viewer



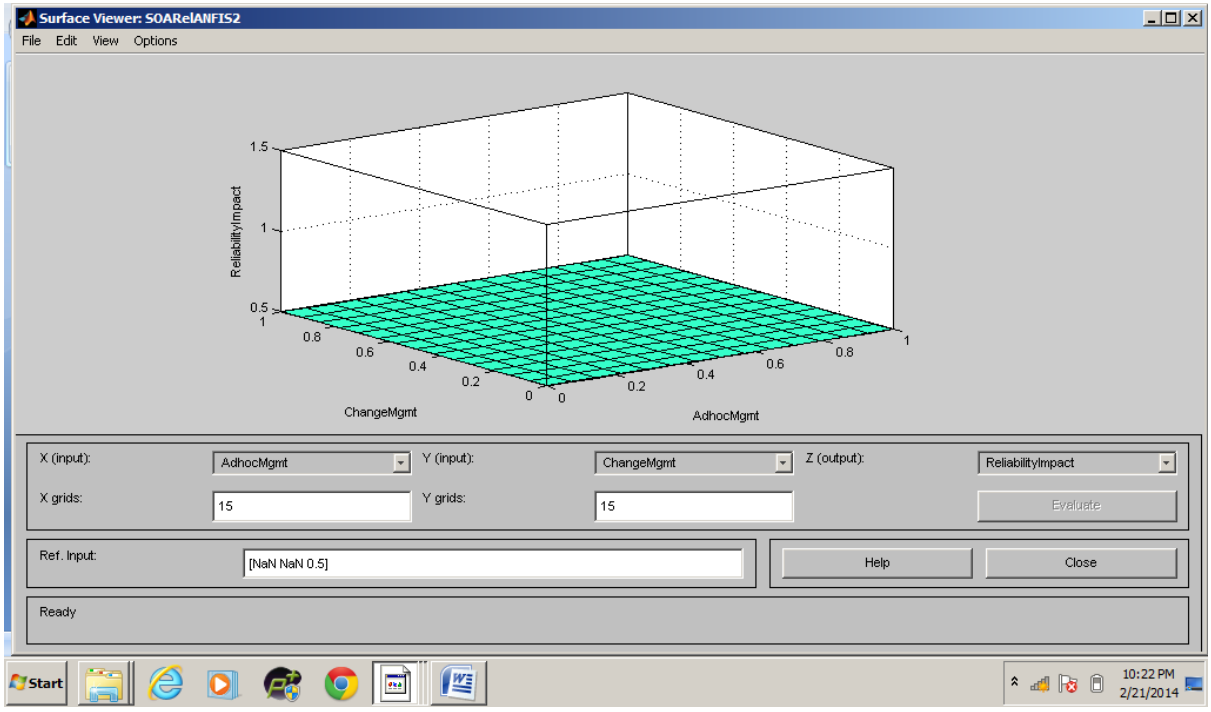
Appendix 3

FIS Editor



Appendix 4

Surface viewer



Appendix 5

Rule Editor