Component Criticality Approach Towards Minimizing the Risks of System Failure

Md. Mesbah-Ul-Awal, Saikat Das
Department Of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna-9203, Bangladesh
e-mail: mesbah_ul_awal@yahoo.com, saikat.bdg@gmail.com

***Abstract*— Component based approach mitigating the risk of system failure has been proposed by detecting of most critical components which's malfunction leads the system towards failure. Individual components have their own chances of occurring fault and these occurrences are silent most often as well as risky; the probability of system failure also becomes high in such phenomena resulting large amount of wretchedness. Protection of components from being faulty can be ensured at the early phase of any structure design or modeling, if the criticality is measured previously. Most reliability assessments of risk minimization have overlooked criticality consideration. However, Criticality is determined in a significant way by measuring each component's complexity and meaningful ranking of components based on random error occurrence. This approach obviously results an improved performance from previously studied cases on system failure risk minimization based on criticality ranking.**

Keywords-Component Criticality Analysis; Complexity Analysis; Severity Ranking; Risk Minimization; System Failure

I. INTRODUCTION

Any critical infrastructure needs to be designed not only for safety and efficiency but also for the risk minimization capability by measuring criticality so that it may maximize the survivability of the system during extreme events such as system failure, or any malicious events [1]. This paper explores component criticality analysis which is validated by automated system: Traffic Control System. Two modes of the system are analyzed here such as – design mode and execution mode. The criticality based approach facilitates the system designers by providing a simple means of estimating the sensitivity of different components according to the probability of being affected by errors. Some approaches may have error minimizing capability but it is much important and substantial to be aware of system failure during system design level and act so by partitioning the system structure into components, defining their risk measurement rules and thus ranking the critical components. This ranking approach allows the designers to prioritize components according to their criticality which will help to identify and mitigate the threats by ensuring improved system reliability. For any software application this would be a generalized approach having methodology consists of designing the system architecture and components such a way that they are capable of measuring complexity and determining severity to find criticality. The rate of complexity determines the inter-component and intra-component dependency and severity measurement classifies the components according to their impact on the system. Therefore, this modern methodology towards risk minimization has been demonstrated by a practical benchmark among critical components analyzing their various system impacts applied in both design and execution mode, discussed in this paper.

II. RELATED WORK

There are so many researches regarding risk minimization of system failure. Most of them assume that system component fails independently but in practice components failure not only affect other components but also vary according to their criticality level [2]. For the both back and front-end engineering it would an improved approach to minimize the risk of system failure if the component criticality evaluation is ensured during system design phase. In the field of military tactics the importance of critical components identification is well established [3]. There are other approaches to the development of safety-critical systems that may including compliance to DOD-178B for good development practices, use of commercial off-the-shelf software components for front end engineering and different forms of software testing to detect threats. But none of them are serious about the failure of component as well as the effect propagation [3].

The researchers in [4] played with transmission system ranked their components based on different criteria such as utilization, performance, importance to the entire system performance and capability etc. Depending on the objective of the ranking they used one or more criteria optionally to rank their additional components. Different components may have different performance indices, different utilizations and different impacts on the system performance when these elements are removed from service [4]. However, other performance criteria also could have been used which they didn't and complexity measurement was also not integrated, which would make the study more efficient thus establishing meaningful credential of reliability regarding corresponding systems.

Various researchers [5], [6] have used different reliability importance and criticality indices for ranking various components. But no technical work has been reported in the

978-0-7695-4941-5/13 $26.00 © 2013 IEEE
DOI 10.1109/ACCT.2013.11

literature for the measurement of components criticality which recommends the study is valid. Some [4] are conscious about component criticality and reliability but do not provide solution of component failure as well as system failure.

Reviewing literatures it is bit clear those researchers have understood the importance of component based analysis. So component detection and verification is the basic challenging part of any component based system analysis. System structure should have a simplified design so that components understandability is maximized and the criteria can be defined for component validation [7]. It would enhance the design mode of any system to track events or any error tracking if occurred. If the code structure has the facility to insert or delete traced code then it is advantageous for the system to minimize the risk of failure by detecting the erroneous component and deleting or replacing that code. Towards risk minimization approach criticality based analysis enhances system reliability by ranking components based on their impact on the system. The FMECA method of analysis covers the evaluation of failure modes, system-level effects, weighting criticality and likelihood of occurrence with impact's diagnosis of system failure. Though the authors in [8] have used FMECA++ approach but they could not differentiated it from the traditional FMECA method. All they have done can be covered by this FMECA for giving weight to the component. Moreover reviewing their cases it seems that they have no reason for risk priority number selection by which components can be ranked. Their used weight based on perceived criticality and likelihood of failure with the average detection feasibility score but not based on practical erroneous situations. It would be more efficient approach if the components can be ranked by studying the practical erroneous phenomena as the vulnerability significantly increasing the failure rate for advanced application components. There may be error in the code or data of the system structure referred by [9] which can lead the system to fail if the corresponding component cannot be identified. If identified then risk minimization operation can be applied and it would be the feasible consideration.

A scenario based approach for reliability analysis of software-based component [10] feels the importance of complexity in enhancing the criticality measurement of any system. Again, a few researchers [11] [12] are conscious of complexity measurement as a risk assessment methodology but not for dynamic environments and assuming a static value for complexity. One way of dealing with the problems of increased complexity in the design is component-based system development which promotes compositional development and component reuse [11]. However, the use of commercial-off-the-shelf components (COTS) in safety-critical system is highly unexplored. There are no obvious methods to incorporate knowledge about functions and resilience of COTS in safety assessments. By measuring complexity inter-component dependency and how much a component can have chance to be affected by errors is determined which results an improved measurement towards finding criticality. Author [11] claims that their approach is based on the system design model augmented with formal failure models. But their increased complexity in safety assessment is one sided approach because it does not deals with the matter of how a component behaves in its execution environment. However, in real time scenarios only structural complexity is not sufficient to represent several possible failure components.

Another Multi-layered Network Analysis [1] has shown a graph based approach to compute the criticality of each network component with respect to each infrastructure layer and rank-order. Their approach is appreciable but lack of complexity measurement, in real time scenario that does not satisfy the risk minimization approach sufficiently.

III. A Methodology To Minimizing The Risks Of System Failure

In computational systems complexity generally means as the measurement of time and space. But in the case of automated systems the complexity refers to the measurement of components belongs to the system which has chances to be affected by error or any unwanted malfunction. The components may be functions that posses certain properties or behavior to have control over the system. Complexity does not measure the impact of systems component failure in system functionality but it may show the rank of components according to their impact if any component experiences error. The measurement of impact over the system due to any component failure is termed as severity. It is the measurement of finding how much a system is affected when individual component has errors and results system failure. In this system methodology of measuring complexity and severity for any single component is proposed along with the risk minimization formula. The combination of these two measurements gives criticality of that single component. If a system takes longer time to process any problem and giving solution then it can be said that the system is more complex. Also a system may have two types of complexity-Structural and Behavioral which are studied as well. Complexity analysis does not measure the impact of components in system functionality; it accelerates defining the ranking criteria of erroneous component. If the erroneous behavior is increased then the complexity will also be increased. So, to minimize risk it is needed to identify the components that are suffering from error before the execution of the system.

Severity is the measurement of how much the system is affected when it is experiencing any error. In other words, when a component is affected by error then the measurement of the impact on the system is referred to as severity measurement. This observation is needed to determine the most critical components. Ideally, soft-error fault-tolerance properties have taken into account already at design time, selecting high-level source-code constructs, algorithms and data-type abstractions and representations as well to ensure integrative design methods properly. When severity measuring, fault tolerance of a system is typically determined afterwards, through testing. Fault injection can be done first as it is a testing approach similar to mutation testing. In this analysis error has been injected randomly around the usage role of the targeted variable. It is claimed

that, the usage role is a good determiner for soft-error impact, thus enabling improvements in terms of better test-case selection and prioritization. Different components comprising the system need a tight methodology also to evaluate their relative importance on the basis of the investigation of the critical components with improved dependability functionality. The dependability may be found by injecting errors in the system randomly and observing the impact on the system for this injection [14]. This random injection of error shows a fair result of system failure and this approach helps to implement a fair risk minimization way. In a summary, the methodology to analyze component criticality includes,

- Measurement of complexity
- Severity of failure
- Minimizing the system failure

The combined result of complexity measurement and severity of failure is then taken as a measure of criticality. This complexity and severity determines how critical a system component is. In the Fig. 1 given below it is shown that how this approach minimizes risks of system failure by calculating their criticality in an efficient way based on components.

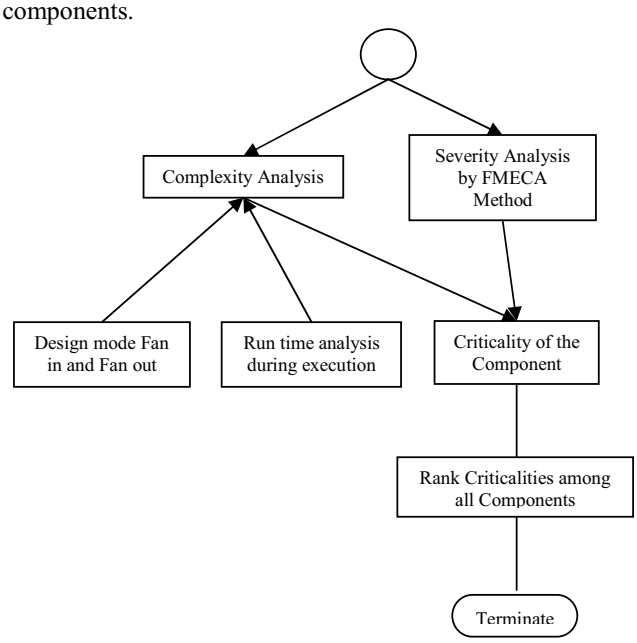


Figure 1. Approach towards detecting critical components by measuring relative criticality

A. Measuring Component Complexity

In the design mode fan-In and fan-Out of each component is calculated which refers the entering and leaving variables among the components. On execution mode the number of variables that are being exchanged among the components is calculated during a certain period of time. After that the average frequency of each component is calculated and added it with design mode fan-In fan-out of corresponding components.

1) Fan In and Fan Out:

To measure the relationships between files and between components the design mode Fan-in and Fan-out values are calculated. Here Fan-in and Fan-out used to measure their complexity of the static structure of the code. It is treated graphically and visualizes components as nodes and Fan-in is the number of links coming into a component and Fan-out is the number of arrows going out of a node. So, for design mode,

Fan-in (component) = number of components that call it,

Fan-out (component) = number of components this component calls.

If the value of Fan-in is high then it indicates a heavily called component where low Fan-out value indicates less dependency on other components. A high Fan-out indicates strongly coupled code which means more complex to execute and test. The measurement of these values gives decision of being a component to be complex.

2) Calculating Inter Component Variable Passing Rate(VPR):

The quantities of variables that are being exchanged among the components are calculated for a certain time interval and obtain the total variable passing amount of each component. If $COMP_{(i)}$ and $COMP_{(i+1)}$ be any two states of a component such as,

$$COMP_{(i)} \rightarrow COMP_{(i+1)} \quad (1)$$

Then the total amount of variable passed among each component is calculated for a certain time interval. In each time interval variable passing amount may vary because of different variable call in or out on that instant of time interval. If *VPR* is expressed in equation, it becomes,

$$VPR = \frac{Variable\ exchanged\ in\ COMP_{(i)} \rightarrow COMP_{(i+1)}}{observation\ time\ interval} \quad (2)$$

So this experiment is repeated for several time intervals and measured variable passing rate.

3) Average Of Variable Passing:

Previously the variable passing rate among the components is calculated based on several time intervals. Here's one thing should be mentioned that inter component variable passing is treated as the message in-out among the component. So getting various variable passing rates, then average them with respect to the total no of occurrence. Finally measuring the complexity of each component the fan in-fan out of design mode and the average value of variable in-out on execution mode is simply added.

B. Measuring Severity Of The Components

A single soft error in a particular component could have a greater effect than multiple soft errors in another or a set of components. For this reason, the effects of soft errors in the whole system need to be analyzed by injecting transient

faults (which will create soft errors if activated) into each component. These results are merged with the component's complexities to obtain a better measure of their impact on the system if it is affected by soft errors. The severity of failure of the components and exchanged variables is determined by the impacts of the system.

1) Injecting Errors Externally:

To find the severity of a component it is decided to inject external error to view the performance of the system and thus can easily identify the component for which a severe system failure occurs. By injecting error on different components separately it can easily overviewed the system whether severe failure occur or not for that erroneous component. To inject error in the simulator there used a table where all variables for that component are showed with real time value which is editable. Editing the value of particular variable and then observe whether the system failure is occurred or not and then prioritized the component.

2) Failure Modes & Effects Criticality Analysis (FMECA):

Failure mode, effects, and criticality analysis (FMECA) is a bottom-up, inductive analytical method which may be performed at either the functional or piece-part level. FMECA extends FMEA by including a criticality analysis, which is used to chart the probability of failure modes against the severity of their consequences. The result highlights failure modes with relatively high probability and severity of consequences, allowing remedial effort to be directed where it will produce the greatest value. After that during the system definition phase of analysis, system has been partitioned into several modules fulfilling component criteria and they are defined such as Traffic module, Vehicle module, Alarm module, Car crash control module.

Before detailed analysis takes place some assumptions are defined to categorize the result of severity. According to the assumption the components are set weight. Failure Mode Identification is an important issue to find out failure mode and failure mode ratio. Failure can be identified by if there is any untimely operation occurs, loss of output, erroneous or invalid output. FMECA usually involves very large data sets; a unique identifier is assigned to each item and to each failure mode of each item. If any component experiences error system level effects may include:

- System failure
- Degraded operation
- System status failure
- No immediate effect

Now, based on the consequences of effect components are assigned severity classification for each failure mode of each unique component. Here is a small set of classification of severity in TABLE I,

TABLE I. SEVERITY CATEGORIES ACCORDING TO IMPACT

Severity Level	Rank	Impact on the system
Catastrophic	10	Rapid car crash Occurs
Serious	9	Multiple car crash on same instances
Extreme	8	Single car crash on individual instances
Major	7	Predicting clash between two cars, one car does not stop
Significant	6	Traffic light delay is changed or no signal
Moderate	5	Traffic signal is not obeyed
Low	4	Frequently cars come not on random speed
Negligible	3	Speed breaker does not invoked
Very minor	2	Not reducing speed after getting alarm
No Effect	1	No alarm for high speed car

Eventually, a weighted value of each component is given with the help of prioritized category, referred severity.

C. Measuring Criticality

Complexity measurement at the system modeling level and Severity combining gross is termed as criticality. After getting the complexity and severity of each component from the analysis, criticality of corresponding component is found by their product. Therefore,

$$Criticality = (Complexity \times Severity) \quad (3)$$

D. Risk Minimization of System Failure

The proposed risk minimization approach based experimental analysis has been shown in Fig. 2 below.

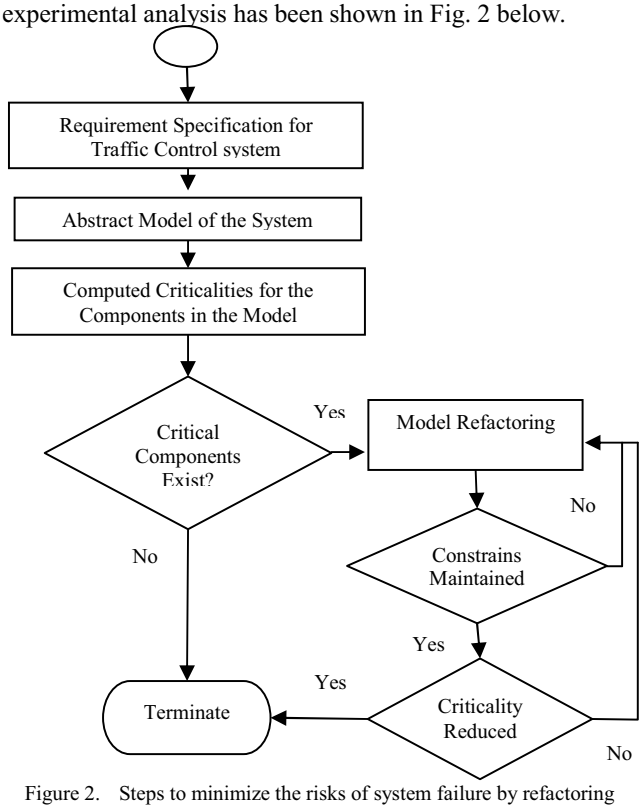


Figure 2. Steps to minimize the risks of system failure by refactoring

From criticality measurement system designers can make decision that whether there is any error in the system or not where to change to fix the problem. Criticality helps in these cases of predicting erroneous components. As much as the product of severity rank and complexity will increase, the risk of system failure is increased. This helps the designers to predict the malicious nodes of the system where criticality is to minimize as well as changing the malicious affecting codes. This would be a structural approach which will redesign the system architecture according to the components criticality level. As a result inter-component dependency in other terms, Fan-in and Fan-out would be changed which will reduce execution time variable exchange rate as well as reduce the total criticality. So, for the improved redesigning paradigm Refactoring is a suitable approach.

IV. EXPERIMENTAL ANALYSIS

To implement this thesis a simulator of an automated traffic system is designed. It is an automated system where the analysis is accomplished. In accordance to this evaluation various components also designed to fulfill the system based on relevant criteria, such as -Traffic Module(TM), Alarm Module (AM), Vehicle Module (VM), Car Crash Control Module (CCCM). It is a safety critical system where Traffic Module(TM) controls traffic lights to run the vehicles in a systematic order. When each vehicle comes to the traffic point it checks whether it is GREEN or RED or YELLOW signals. RED signal forces to be stopped at traffic point for 25 seconds and the YELLOW allows all vehicles to get prepared to stop or start within 15 seconds. If any vehicle acquires over speed closed to the traffic junction such that the system may feel risk of any accident then the Alarm Module generates an alert for that vehicle and the speed of that vehicle is lessen. Each vehicle generates random speed and run by this speed. As this system referred as a safety critical system there is another component here that always monitors each car with all other cars if there is any clash or not. It always checks the minimum distance between cars to avoid car crash as well as system failure. When the simulator is executed it has an interface looks like Fig. 3

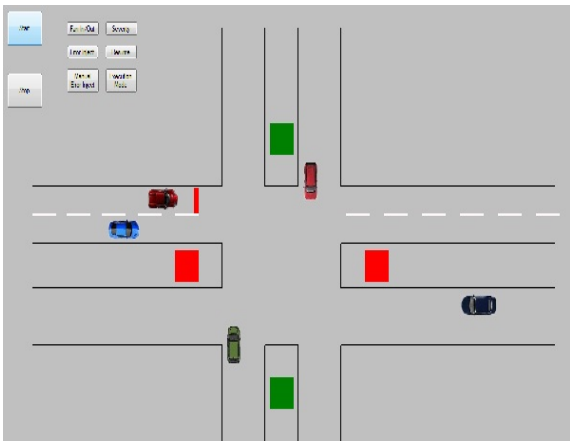
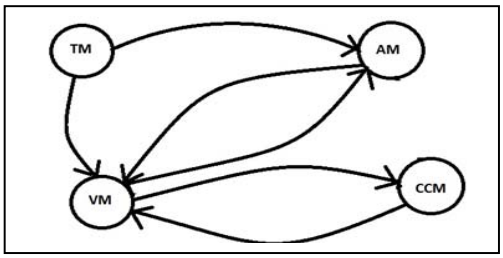


Figure 3. Interface of Designed Simulator

A. Complexity Analysis

According to the methodology design mode evaluation is performed considering Fan-in and Fan-out. But the efficient evaluation criterion is ensured when variable passing rate is determined in execution mode with respect to the execution time. These methodologies are applied in the Traffic Control System explained above. The four individual components have four individual behaviors to control the traffic system automatically. The dependency graph based on complexity of components is shown below,



Here,
TM= Traffic Component
AM= Alarm Component
VM= Vehicles component
CCM= Car Crash Control Component

Figure 4. Component dependency based on variable passing

The variable passing rate (*VPR*) of each component has been calculated according to (2). Here the experiment has been accomplished considering *observation time interval* is 5 minutes and reported accordingly.

Similarly variable exchange among the components for 15 minutes also calculated. If the *VPR* measurement repeated for consecutive n times for the same time interval k and after that the average is calculated, it contributes a meaningful measurement of finding complexity of execution mode of each component. This may be expressed in equation as follows:

$$\text{Avg. } VPR = \frac{\sum_{i=1}^{n} VPR}{n} \quad (4)$$

For each k=5 minutes and repeated n=4 times, here is the analysis result that is shown in the Table II.

TABLE II. TABLE FOR COMPONENT COMPLEXITY FOR K=5 AND N=4

Components	VPR for 1st 5 minutes	VPR for 2nd 5 minutes	VPR for 3rd 5 minutes	VPR for 4th 5 minutes	Average VPR	Complexity
Traffic Module(TM)	80	80	88	80	82	0.00027
Alarm Module(AM)	371810	400886	240093	258644	317858.25	1.0595
Vehicle Module(VM)	30040	23247	23782	25120	25547.25	0.0851
Car Crash Control Module(CCM)	908	690	838	782	804.5	0.00268

Expressing the analyzed data in a graph the comparison can be visualized of each component during certain execution period of time k. The graph representation in Fig. 5 helps prioritize components according to their reliability quality. Comparison of variable passing rate of all modules with respect to different time period,

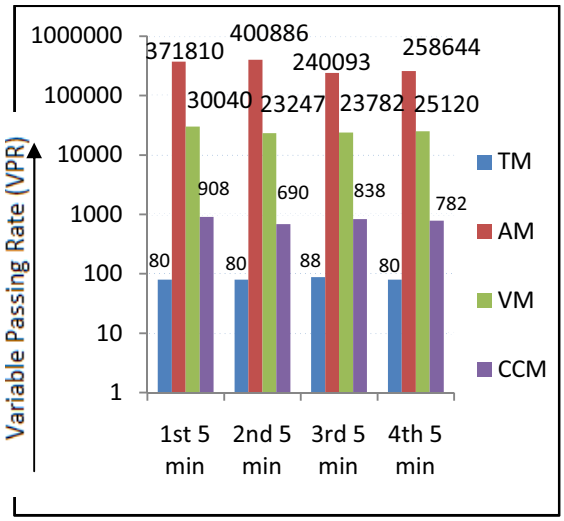


Figure 5. Comparison of all Components according to *VPR*

The complexity is now obtained by adding fan-In fan-out and *VPR*. The comparison of different components is shown in graphically to the next page on Fig. 6,

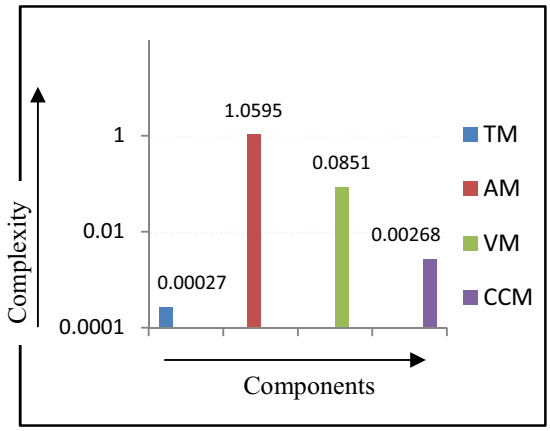


Figure 6. Comparison of Complexity among all Components

Therefore, it is a simple way of deciding and detecting the most critical component.

B. *Severity Analysis*

For the measurement of severity the Failure Mode and Effect Criticality Analysis (FMECA) method is used here. The measurement of severity determines that how much the system is affected when any error is occurred. A single soft error in a particular component can have the effect larger than the effect caused by multiple soft errors in any component. Here errors have been injected to each component to observe the impact due to error. After that these results are merged with complexities. This results a better measurement of their impact on the system. Here, those impacts are ignored which caused permanent damage to the system and the impacts by which the system experiences chaos are counted. Some errors do not let the system to be executed as a result the system becomes unusable. So that such types of error are ignored.

Based on the impact on the system components are weighted according to their severity. Weight table of the components according to their severity after error injection shown to the below Table III

TABLE III. TABLE FOR BENCHEDMARKED WEIGHTS OF DIFFERENT COMPONENTS BASED ON SEVERITY

Components	Weight or Rank
Traffic Module(TM)	**5**
Alarm Module(AM)	**8**
Vehicle Module(VM)	**7**
Car Crash Control Module(CCM)	**10**

C. *Criticality Analysis*

In this analysis the criticality is determined by measuring the product of complexity and severity of each component given in (3). For any increase in complexity there is a high probability that not only the severity will be increased but also total risks of system failure will be increased. Hence, criticality is taken as the product of overall complexity and severity. The experimental result is shown below,

TABLE IV. TABLE FOR EXPERIMENTAL CRITICALITY OF COMPONENTS

Components	Criticality
Traffic Module(TM)	**0.00135**
Alarm Module(AM)	**8.476**
Vehicle Module(VM)	**0.5957**
Car Crash Control Module(CCM)	**0.0268**

V. CONCLUSIONS

This research is motivated by the fact of measuring component criticality of a system and establishing a general methodology that has ability to minimize the risks of a system failure. Several relevant papers are analyzed for gathering knowledge to accomplish this research. This analysis covers significant sorts of risks minimization tasks that prevents system components from occurring failure. Though error injection is a tough task, but it is handled here

logically without breaking system consistency. How much severe damage may consequent if any system fails due to critical components have already discussed and given an improved way to get rid of disaster of system failure. Thus the criticality analysis represented in this paper will have great benefit towards minimizing risks of system failure. There is an open scope to extend this research by considering propagation of failure in computing criticality and merging the consequences with existing measurements. Though this part remains uncalculated here, a logical prediction of failure propagation is covered based on severity which leaded this research to decide most critical components successfully. Another matter regarding time and space complexity of the system structure should have in consideration when working on refactoring methodology. So, maintaining minimum threshold level of time and space complexity risks minimization approach can be accomplished in future.

ACKNOWLEDGMENT

All the gratitude goes to the Almighty for his kind compassion. The Authors express their indebtedness to Dr. Muhammad Sheikh Sadi for his kind supervision and excellent guidance. His suggestions, guidelines about the theme and motto of the research are the reasons, the study is accomplished. Also, special thank goes to Dr. K.M Azharul Hasan, Professor and Head Dept of Computer Science and Engineering, KUET for his encouragement and outstanding help providing valuable potentials. Lastly, Authors are grateful to their family for their firm support.

REFERENCES

[1] Abdel-Rahim, P. Oman, B.K. Johnson, R.A. Sadiq, "Assessing surface transportation network component criticality: A multi-layer graph-based approach", Proceedings of the 2007 IEEE, Intelligent Transportation Systems Conference Seattle, WA, USA, Sept. 30 - Oct. 3, 2007 vol-1, pp.1000-1003, 2007.

[2] Atef Mohamed and Mohammad Zulkernine, "Failure type-aware reliability assessment with component failure dependency", School of Computing, Queen's University, Kingston Ontario, Canada K7L 3N6, IEEE Computer society : Secure Software Integration and Reliability Improvement (SSIRI), vol-1, pp.98-105, 2010.

[3] Hassan Reza, Steve Buettner, Varun Krishna "A method to test Component Off-The-Shelf (COTS) used in safety critical systems", School of Aerospace Sciences, University of North Dakota Grand Forks, 58202 USA. Information Technology: New Generations, pp.189-194, 2008.

[4] Hamoud, G.A, "Assessment of transmission system component criticality in the de-regulated electricity market", Hydro One Inc., Toronto, ON .Probabilistic Methods Applied to Power Systems, 2008. PMAPS '08. Proceedings of the 10th International Conference vol-1,pp.1-8, May 2008.

[5] Ramirez-Marqurez, J. E and Coit, "Composite importance measures for multi-state systems with multi-state components", IEEE Transaction on Reliability, vol.54, pp. 517-529. Sept 2005.

[6] Hamoud, G., lee, L., Tonegouzzo J., and Watt G. "Assessment of component criticality in customer delivery systems", Probabilistic Methods Applied to Power Systems, 2004 International Conference Iowa State University, Ames, Iowa, USA, vol-10, pp.819-825.Sep 2004.

[7] Jerry Gao, Ming-Chih Shih "A component testability model for verification and measurement", Proceedings of the 29th Annual International Computer Software and Applications Conference.vol-1, pp.211-218, 2005.

[8] Baybutt, M.; Nanduri, S.; Kalgren, P.W.; Bodden, D.S "Seeded fault testing and in-situ analysis of critical electronic components in EMA power circuitry.", Aerospace Conference, 2008 IEEE .vol-1, pp.1-12, 2008.

[9] Adam Piotrowski, Dariusz Makowski, Grzegorz Jabło´nski, Andrzej Napieralski, "The automatic implementation of software implemented hardware fault tolerance algorithms as a radiation-induced soft errors mitigation technique.", IEEE Nuclear Science Symposium Conference Record, vol-1, pp.841-846, 2008.

[10] Yacoub, S.M.; Cukic, B.; Ammar, H.H , "Scenario-based reliability analysis of component-based software.", Software Reliability Engineering, 1999. Proceedings. 10th International Symposium, vol-4, pp.22-31, 1999.

[11] Elmqvist, J.; Nadjm-Tehrani, S., "Tool support for incremental failure mode and effects analysis of component-based systems.", Design, Automation and Test in Europe, 2008. Vol-1, pp.921-927, 2008.

[12] Ping Gao, Su Wu, "Improved criticality analysis of railway locomotive components by simulation",Tsinghua University Zhonghua Cheng, PhD, Tsinghua University, Reliability and Maintainability Symposium , vol-1, pp.494-498, 2007.

[13] V. Cortellessa and V. Grassi, "A modeling approach to analyze the impact of error propagation on reliability of component-based systems", Proceedings of the 10-th International Symposium on Component Based Software Engineering (CBSE'07), LNCS4608, pp. 140-156, November 2007.

[14] W. Abdelmoez, D.M. Nassar, M. Shereshevsky, N. Gradetsky, R. Gunnalan, H.H. Ammar, B. Yu, A. Mili, "Error propagation in software architectures", Proceedings of the 10-th IEEE International Symposium on Software Metrics(METRICS'04), Morgantown, WV, USA, pp. 384-393, Sep 2004.

[15] A. Avizienis, J.C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing", IEEE Transactions on Dependable and Secure computing, vol. 1, pp. 11-33, January 2004.

[16] S. Chetan, A. Ranganathan, and R. Campbell, "Towards fault tolerance pervasive computing", Proceedings of the IEEE Technology and Society Magazine, vol. 24, no. 1, pp. 38-44, 2005.

[17] R. S. Freedman, "Testability of software components". IEEE Transactions on Software Engineering, 17(6), 553-563, June 1991.

[18] Everett, W., "Software component reliability analysis", Proc. of the 1999 IEEE Symposium on Application-Specific Systems and Software Engineering & Technology, ASSET'99, Richardson, Texas, March 24-27, 1999, pp204-211, March 1999.

[19] C.L. Blackmon, and M.L. Yin., "A design tool for large scale fault tolerant software systems", Proceedings of the Reliability and Maintainability Annual Symposium, pp. 256-260, January 2004.

[20] A. G. Mohamed, S. Chad, T. N. Vijaykumar, and P. Irith, "Transient-fault recovery for chip multiprocessors", IEEE Micro, vol. 23, pp. 76, 2003.

[21] B. Littlewood and L. Strigini, "Software reliability and dependability: a roadmap", Proceedings of the 22-nd IEEE International Conference on Software Engineering on the Future of Software Engineering (ICSE'00), 2000. pp. 175-188, Limerick, Ireland, 2000

[22] Chidamber, S.R., Kemerer, C.F., "A metrics suite for object oriented design," IEEE Transactions on Software Engineering, vol. 20, pp. 476-493, 1994.