RELIABILITY TESTING METHODOLOGIES AND DISCRETE APPROACH FOR INTENSIVE COMPLEX SYSTEMS

[1]K. UMA MAHESWARI, [2]K.MALLIKARJUNA REDDY
[1]Asst. Professor, Srinivasa Ramanujan Institute of Technology, Ananthapuramu, A.P, India,
[2]Professor, Sri Krishna Devaraya University, Ananthapuramu, A.P, India.

ABSTRACT

The reliability of aircraft electrical components directly affects the combat performance of the aircraft. But the reliability is determined by the design, processing, environmental factors, accidents and other factors. Environmental testing is to prove the reliability and processing. The current reliability testing methods cannot precisely evaluate the comprehensive reliability of Software-Intensive Complex Systems. In order to solve the problem, it is necessary to research the method of carrying out the software-hardware interdependent reliability test. In this thesis, we do research on Software-hardware interdependent fault mode of Software-Intensive Complex Systems by collecting and studying the faults occurred in practical filed use. Then we design the software-hardware testing profile by adding software test profile to the reliability test profile, and put forward the idea about designing the test environment of software-hardware testing, which constitute a whole plan of software-hardware testing. And then a case study is demonstrated, which testifies the validation of the technique.

KEYWORDS: Design Optimization, Complex Systems, reliability, reliability test, Developmental Testing and Manufacturing Testing.

1. INTRODUCTION

Reliability testing is the cornerstone of a reliability engineering program. It provides the most detailed form of reliability data because the conditions under which the data are collected can be carefully controlled and monitored. Furthermore, reliability tests can be designed to uncover particular suspected failure modes and other problems. The type of reliability testing a product undergoes will change along different points of its life cycle, but the overriding goal is to ensure that data from all or most of the tests were generated under similar enough conditions so that an "apples to apples" comparison can be made of the product's reliability characteristics at different points in the product's life. It is for this reason that consistent and thorough reliability specifications and a standard definition of failure are up-front requirements to implementing reliability testing.

A properly designed series of tests, particularly during the product's earlier design stages, can generate data that would be useful in the implementation of a reliability growth tracking program. This will provide information that will be helpful in making management decisions regarding scheduling, development cost projections and so forth. This information will also be useful in planning the development cycle of future products.

Software-Intensive Complex Systems are those whose software plays a major role in the completion of required function or task, or in the development, operation, or the evolution of complex systems. The functions of complex software-intensive systems such as industrial control, avionics, satellite, telecommunications and other complex computer systems with hardware subsystems and software subsystems, can only be realized through the help of the embedded computer system featured by hardware and software together. With the extensive application of computer technology industries, complex software-intensive systems playing an increasingly important role in industrial products and people's daily lives are more closely related than ever.

This paper is about doing accelerating environmental testing and simulating main effects of environmental factors for aircraft electrical components. Without changing the actual damage mechanism of the premise, based on accelerated life reliability methods, we can determine the law of environmental factors effecting on aircraft performance and reliability of electrical components.

2. DEVELOPMENTAL TESTING

Developmental testing occurs during the early phases of the product's life cycle, usually from project inception to product design release. It is vital to be able to characterize the reliability of the product as it progresses through its initial design stages so that the reliability specifications will be met by the time the product is ready for release. With a multitude of design stages and changes that could affect the product's reliability, it is necessary to closely monitor how the product's reliability grows and changes as the product design matures. There are a number of different test types that can be run during this phase of a product's life cycle to provide useful reliability information:

Component-level Testing:

Although component-level testing can continue throughout the development phase of a product, it is most likely to occur very early in the process. This may be due to the unavailability of parts in the early stages of the development program. There may also be special interest in the performance of a specific component if it has been radically redesigned or if there is a separate or individual reliability specification for that component. In many cases, component-level testing is undertaken to begin characterizing a product's reliability even though full system-level test units are unavailable or prohibitively expensive. However, system-level reliability characterization can be achieved through component-level testing. This is possible if sufficient understanding exists to characterize the interaction of the components. If this is the case, the system-level reliability can be modeled based on the configuration of components and the result of component reliability testing.

System-level Testing:

Although the results of component-level tests can be used to characterize the reliability of the entire system, the ideal approach is to test the entire system, particularly if that is how the reliability is specified. That is, if the technical specifications call out a reliability goal for a specific system or configuration of components, that entire system or configuration should be tested to compare the actual performance with the stated goal. Although early system-level test units may be difficult to obtain, it is advisable to perform reliability tests at the system level as early in the development process as possible. At the very least, comprehensive system-level testing should be performed immediately prior to the product's release for manufacturing in order to verify design reliability. During such system-level reliability testing, the units under test should be from a homogeneous population and should be devoted solely to the specific reliability test. The results of the reliability test could be skewed or confounded by "piggybacking" other tests along with it and this practice should be avoided. A properly conducted system-level reliability test will be able to provide valuable engineering information above and beyond the raw reliability data.

Environmental and Accelerated Testing:

It may be necessary in some cases to institute a series of tests in which the system is tested at extreme environmental conditions or with other stress factors accelerated above the normal levels of use. It may be that the product would not normally fail within the time constraints of the test and, in order to get meaningful data within a reasonable time, the stress factors must be accelerated. In other cases, it may be necessary to simulate different operating environments based on where the product will be sold or operated. Regardless of the cause, tests like these should be designed, implemented and analyzed with care. Depending on the nature of the accelerating stress factors, it is easy to draw incorrect conclusions from the results of these tests. A good understanding of the proper accelerating stresses and the design limits of the product are necessary to be able to implement a meaningful accelerated reliability test. For example, one would not want to design an accelerated test that would overstress the product and introduce failure modes that would not normally be encountered in the field. Given that there have been a lot of incredible claims about the capability of accelerated testing and the improbably high acceleration factors that can supposedly be produced, care needs to be taken when setting up this type of reliability testing program.

Shipping Tests:

Although shipping tests do not necessarily qualify as reliability tests per se, shipping tests or simulations designed to test the impact on the product of shipping and handling should be a part of the reliability testing program. This is because the effects of shipping will often have an impact on the reliability of the product as experienced by the customer. As such, it may be useful to incorporate shipping tests alongside the normal reliability testing. For example, it may be a good idea to put the units of a final design release reliability test through a non-destructive shipping test prior to the actual reliability testing in order to better simulate actual use conditions.

3. TEST PLANNING FOR RELIABILITY TESTING

Reliability Testing is costly compared to other types of testing. So Proper planning and management is required while doing reliability testing. This includes testing process to be implemented, data for test environment, test schedule, test points, etc.To begin with reliability testing, tester has to keep following things,

- Establish reliability goals
- Develop operational profile
- Plan and execute tests
- Use test results to drive decisions

As we discussed earlier, there are three categories in which we can perform the Reliability Testing, -Modeling, Measurement and Improvement. The key parameters involved in Reliability Testing are:-

- Probability of failure-free operation
- Length of time of failure-free operation
- The environment in which it is executed

4. Modeling:

Software Modeling Technique can be divided into two subcategories:

1. Prediction Modeling

2. Estimation Modeling

Meaningful results can be obtained by applying suitable models. Assumptions and abstractions can be made to simplify the problems and no single model will suitable for all the situations.

The major differences of two models are:- Table:-1

Issues	**Prediction Models**	**Estimation Models**
Data Reference	It uses historical data	It uses current data from the software development.
When used inDevelopment Cycle	It will be usually created before the development or testing phases.	It will be usually used at the later stage of Software Development Life Cycle.
Time Frame	It will predict the reliability in the future.	It will predict the reliability either for the present time or in the future time.

5. Measurement:

Software reliability cannot be measured directly and hence, other related factors are considered in order to estimate the software reliability. The current practices of Software Reliability Measurement are divided into four categories: -

1. Product Metrics:-

Product metrics are the combination of 4 types of metrics:

Software size: - Line of Code (LOC) is an intuitive initial approach for measuring the size of the software. Only the source code is counted in this metric, and the comments and other non-executable statements will not be counted.

Function point Metric:- Function Pont Metric is the method for measuring the functionality of the Software Development. It will consider the count of inputs, outputs, master files, etc. It measures the functionality delivered to the user and is independent of the programming language.

Complexity:- It is directly related to software reliability, so representing complexity is important. Complexity-oriented metric is a method of determining the complexity of a program's control structure, by simplifying the code into a graphical representation.

Test Coverage Metrics:- It is a way of estimating fault and reliability by performing the complete test of software products. Software reliability means it is the function of determining that the system has been completely verified and tested.

2. Project Management Metrics

Researchers have realized that good management can result in the better products. A good management can achieve higher reliability by using better development process, risk management process, configuration management process, etc.

3. Process Metrics

The quality of the product is directly related to the process. The process metrics can be used to estimate, monitor and improve the reliability and quality of software.

4. Fault and Failure Metrics

Fault and Failure Metrics are mainly used to check whether the system is completely failure-free. Both the types of faults found out during the testing process (i.e. before delivery) as well as the failure reported by users after delivery are collected, summarized and analyzed to achieve this goal.

6. Improvement:

Improvement completely depends upon the problems occurred in the application or system, or else the characteristics of the software. According to the complexity of the software module, the way of improvement will also differ. Two main constraints time and budget, which will limit the efforts are put into the software reliability improvement.

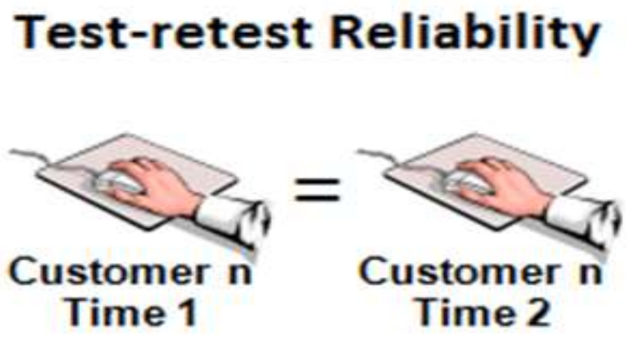
7. RELIABILITY TESTING APPROACH

Testing for reliability is about exercising an application so that failures are discovered and removed before the system is deployed.There are mainly three approaches used for Reliability Testing

- Test-Retest Reliability
- Parallel Forms Reliability
- Decision Consistency

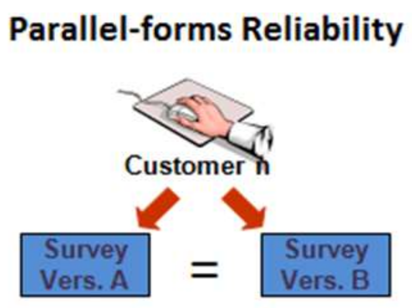
Below we tried to explain all these with an example.

Test-Retest Reliability



To estimate test-retest reliability, a single group of examinees will perform testing process only a few days or weeks apart. The time should be short enough so that the examinees skills in the area can be assessed. The relationship between the examinee's scores from two different administrations is estimated, through statistical correlation. This type of reliability demonstrates the extent to which a test is able to produce stable, consistent scores across time.

Parallel Forms Reliability



Many exams have multiple formats of question papers, these parallel forms of exam provide Security. Parallel forms reliability is estimated by administrating both forms of the exam to the same group of examinees. The examinees scores on the two test forms are correlated in order to determine how similarly the two test forms functions. This reliability estimate is a measure of how consistent examinees scores can be expected to across test forms.

8. FAULT MODEOF SOFTWARE-INTENSIVE COMPLEX SYSTEMS

The fault modes of Software-Intensive Complex Systems are hardware fault mode software fault mode and Software hardware interdependent fault mode. Hardware fault is stimulated by types and levels of stresses, however software fault is led to by a certain operational condition, for example, caused by the defected code, theoretically time dependent. However, software-hardware interdependent fault mode is related to both time and state, caused by unpredictable changes of operational conditions or environment. Those changes can directly act on hardware parts of systems, and as time goes by, the stresses infiltrate into inner unit and gradually affect components and wires, abnormal signals affecting software so that we see an abnormal performance of software, which in turn affects hardware and system leading to an overall failure or local failure. Therefore Software-hardware interdependent fault mode can be called 'software-hardware fault-based time/state'.

No.	Fault description	Fault mode
1	Failed to track the target at low temperatures	Temperature is too low resulting in a timing signal distortion and data transfer dislocation, impacting the realization of software and system function
2	Failed to debug after relocation	Electromechanical switch chip pulse pounding leads to data loss, so that the parameters are changed and the software function is not implemented
3	Processing crashed switching modes unit when the work	Work mode switching caused data bounds so that software crashes
4	Displaying Unit	Cable damage causes electromagnetic interference and bus flag is rewritten as well as software fault is reported
5	Antenna tilt stopper fractured	Software control limit margin is too small, resulting in damage to the hardware when the carrier aircraft is overload
6	Servo system crashed due to over current	Servo system software overload time setting is too small, causing the device to fail to work
7	Extended Range manner works when information processor crashed	Compiler software error caused the interruption of the output signal leading to software crashes

Analysis were implemented about the faults discovered in reliability testing and software testing and we found that the fault modes involved bounding , timing disorder at low temperatures and small-time control margins, most of which are resulted from software-hardware interdependent problems. The times when these faults occurs are dependent on the stringent timing and logic matching relationship between software operation and environments (temperature, humidity, power, vibration), therefore current reliability testing method is not in a position to expose such fault modes.

That’s why software –hardware reliability test should be carried out as for software-hardware interdependent fault modes, enhancing the efficiency of reliability test. That’s why software–hardware reliability test should be carried out as for software-hardware interdependent fault modes, enhancing the efficiency of reliability test.

9. COMPLEX SYSTEM RELIABILITY TEST BASED ON THE EMPIRICAL LIKELIHOOD

To infer the reliability lower confidence limit or the confidence limit of a CS using subsystem data, we can construct the following hypothesis tests: For a given t, test whether Ψ1,A(t) is not less than Ψ0 or not; that is,

$$(1)\quad H_0 : \Psi_{1,A}(t) = \Psi_0 \text{ vs } H_1 : \Psi_{1,A}(t) \neq \Psi_0;$$

$$(2)\quad H_0 : \Psi_{1,A}(t) \geq \Psi_0 \text{ vs } H_1 : \Psi_{1,A}(t) < \Psi_0.$$

We use empirical likelihood (EL), which is a nonparametric method introduced by Owen [9, 10, 13], to test the two hypotheses. Here, we first give a short introduction to empirical likelihood. The definitions of the empirical distribution function and the empirical accumulate function are given as follows.

Let $X_1, X_2, \ldots, X_n \in \mathbb{R}$ be independently identically distributed, then the empirical cumulative distribution function of X1, X2, …, Xn isfor −∞ < x < ∞.

$$F_n(x) = \frac{1}{n}\sum_{i=1}^{n} I_{(X_i \leq x)},$$

Let $X_1,\ X_2, \ldots, X_n \in \mathbb{R}$, be independent and with a common cumulative distribution F, the nonparametric likelihood of the F is

$$L(F) = \Pi_{i=1}^{n}(F(X_i) - F(X_i -)).$$

Define

$$R(F) = \frac{L(F)}{L(F_n)}.$$

Then the empirical likelihood ratio statistic is defined to be

$$\Re(\mu_0) = \max\left\{\Pi_{i=1}^{n} nw_i \middle| \sum_{i=1}^{n} w_i(X_i - \mu_0) = 0, w_i \geq 0, \sum_{i=1}^{n} w_i = 1\right\}.$$

The resulting empirical likelihood confidence region for the mean μ0 i

$$\{\mu | R(\mu_0) \geq r_0\} = \arg\max\left\{\sum_{i=1}^{n} w_i X_i \middle| \Pi_{i=1}^{n} nw_i \geq r_0, w_i \geq 0, \sum_{i=1}^{n} w_i = 1\right\}.$$

For a given hypothesis test, we can obtain a confidence region with the empirical likelihood method without assuming the specification of the data distribution family. In the following two subsections, we will derive the test statistics for two hypothesis tests for the reliability of a complex system using the empirical likelihood method.

10. CONCLUSION

It is meaningful that we make research into software hardware reliability test technique for software-intensity complex systems. Our project provides software-hardware reliability test, actually reflecting how system works in practical use. The case in this paper is universal and the techniques provided here can be used as norms when proper improvements are made.

11. REFERENCES

1. Moskowitz, F. and Mclean, J. B., Some reliability aspects of system design, *IRE Transactions on Reliability and Quality Control*, v-8, 1956, 7-35.
2. Tillman, F. A. and Ditchwater, J. M., Integer programming formulation of constrained reliability problems, Management Science, v-13, 1967, 887-899.
3. Sharma, J. and Venkateshwara, K. V., A direct method for maximizing the system reliability, IEEE Transactions on Reliability, v-20, 1971, 256-259.
4. Misra, K. B. and Ljubojevic, M. D., Optimal reliability design of a system : a new look, IEEE Transactions on Reliability, v-22, 1973, 255-258.
5. Luus, R., Optimization of system reliability by a new nonlinear integer programming procedure, IEEE
6. Transactions on Reliability, v -24, 1975, 14-16.
7. Agarwal, K. K., Gupta, J. S. and Misra, K. B., A new heuristic criterion for solving a redundancy optimization problem, IEEE Transactions on Reliability, v -24, 1975, 8687.
8. Tillman, F. A., Hwang, C. L., Fan, L. T. and Balbale,S.A., Systems reliability subject to multiple nonlinear constraints, IEEE Transactions on Reliability, v -17, 1968, 153-157.
9. Tillman, F. A., Hwang, C. L. and Kuo, W., Determining component reliability and redundancy for optimal system reliability, *IEEE Transactions on Reliability*, v -26, 1977, 162-165.
10. Nakagawa, Y. and Nakashima, K., A heuristic method for determining reliability allocation, *IEEE Transactions on Reliability*, v -26, 1977, 156-161.Inagaki, T., Inoue, K. and Akashi, H., Interactive optimization of system reliability under multiple objectives, *IEEE Transactions on Reliability*, v -27, 1978, 264-267.
11. Cateanu, V. M., Popentiu, F. and Gheorgiu, M., Areliability optimization computer algorithm for complex systems, *Microelectronics Reliability*, v-26, 1986, 10191023.
12. Hikita, M., Nakagawa, Y., Nakashima, K. and Narihisa,H., Reliability optimization of systems by a surrogate constraints algorithm, *IEEE Transactions on Reliability*, v 41, 1992, 473-480.
13. Gen, M., Ida, K., Tsujimura, Y. and Kim, C. E., Largescale 0-1 fuzzy goal programming and its applications to reliability optimization problem, *Computers and Industrial Engineering*, v-24, 1993, 539-549.
14. Hwang, C. L., Lee, H. B., Tillman, F. A. and Lie, C. H.,Nonlinear integer goal programming applied to optimal system reliability, *IEEE Transactions on Reliability*, v -33, 1984, 431-438.
15. Coit, D. W. and Smith, A. E., Reliability optimization ofseries-parallel systems using a genetic algorithm, *IEEE Transactions on Reliability*, 1996, in press.
16. Painton, L. and Campbell, J., Genetic algorithms in optimization of system reliability, *IEEE Transactions on Reliability*, v-44, 1995, 172-178.
17. Kumar, A. and Malik, K., Voting mechanisms in distributed systems, *IEEE Transactions on Reliability*, v-40, 1991, 593-600.
18. Pisinger, D., A minimal algorithm for the multiple-choice knapsack problem, DIKU, University of Copenhagen, Denmark, Report 94/25, 1994.
19. Ceria, S., Cornuejols, G. and Dawande, M., Combinatory cuts, *Proceedings of the 4th International Conference on Integer Programming and Combinatorial Optimization*, May 29-31, 1995, Copenhagen, Denmark.
20. Brusco, M. J. and Jacobs, L. W., A simulated annealing approach to the cyclic staff-scheduling problem, *Naval Research Logistics*, v-40, 1993, 69-84.
21. Kirkpatrick, S., Gelatt, C. D. and Vecchi, M. P.,Optimization by simulated annealing, *Science*, v-220, 1983, 671-680.
22. Glover, F., Tabu search, *ORSA Journal on Computing*, v-1, 1989, 190-206.
23. Glover, F., Tabu search - Part II, *ORSA Journal on Computing*, v-2, 1989, 4-32.
   Eglese, R. W., Simulated annealing : A tool for operational research, *European Journal of Operational Research*, v-46, 1990, 271-281.