A Summary of Comparative Study of Software Reliability

Chong Peng, Yujie Meng, Liyun Lan, Lun Wang
School of Mechanical Engineering and Automation, Beihang University, Beijing, China
e-mail: pch@buaa.edu.cn, myj15873@163.com

***Abstract*—With the integration of informatization and industrialization, the application of software is getting more and more extensive and plays a powerful role in many facilities. At the same time, Software failures cause tremendous losses, thus ensuring the reliability of software becomes increasingly important. The basic conceptions of software reliability are put forward in this paper, and comparative analysis on the research status at home and abroad are studied. Meanwhile, the perspective of the further progress of software reliability is made.** *(Abstract)*

Keywords- software reliability; integration of informatization and industrialization; comparative analysis

I. INTRODUCTION

Nowadays, software plays an increasingly important role in more industries. With the modern industrial systems growing more complex, assurance of software reliability becomes more difficult. At present, though a large number of researches have been carried out and plenty of applications have been put into use, there is still a long way to go in the field of software reliability.

II. DEFINITION AND IMPORTANCE OF SOFTWARE RELIABILITY

A. Definition

IEEE Computer Society made a clear definition of software reliability in 1983, which was accepted as national standard by National Institute of Standards and Technology (NIST) in the United States. Later in the year 1989, China also accepted the definition as national standard. According to GB/T 11457-95-Software Engineering Terms, the definition of software reliability is as follows [1]:

- Software reliability is the probability of failure-free software operation for a specified period of time under a specified condition. This probability is a function of the input and usage of the system as well as the failure existed in software. The system input will determine whether an existing failure will be encountered.
- Software reliability is the ability that software performs the required functions during the prescriptive period under a specified situation.

B. Differences between Software Reliability and Hardware Reliability

A large percentage of hardware failure is due to equipment wear and material aging, while software will not change as time goes on, namely never wear.

The critical factor of hardware reliability is time, which can be affected by the process of design, manufacture, and service. Nevertheless, source code is the critical factor of software reliability. As for embedded software, the fault of the interface between hardware and software is a major factor resulting in failure [2].

C. Importance of Software Reliability

1) Software reliability is an essential condition to guarantee normal system operation

The effect of software is getting more and more influential as an increasing number of digital devices are putting into use. In the aerospace domain, the scale of source code in the airborne software reaches million lines. However, the sharp increase of scale and complexity in software also gives rise to the increase of failure number. One study shows, the codes written by professional software developers would have 6 faults every thousand lines [3]. Following this fault density, software with a million line codes would have as many as 6000 faults. What is worse, the density of fault increases geometrically as the scale of software grows. The increasing number of faults makes fault location more difficult and the repair cost rise dramatically. Besides, software failure can cause serious consequences. The most famous examples are: in 1962, MARINER I sent by the United States to Venus lost control 293 seconds after being launched. NASA owed this fault to the wrong code line in the Fortran language (missing a hyphen), causing the cost loss as high as 80 million dollars. The data from the famous safety agency SecurityFocus shows that, the most serious power outage in history occurred in the United States and parts of Canada on August 14th, 2003 was resulted from software failure. Actually, serious accidents caused by software failure are by no means only these two. These accidents teach us a lesson that software reliability must be taken into consideration before devices been put into use.

2) Software reliability becomes the bottleneck to improve system reliability

Software plays an increasing part in systems. For example, every time the fighter aircraft updates a new generation, the functions realized by software doubled [3]. Software reliability is directly related to system reliability.

Published by Atlantis Press, Paris, France.

Compared to hardware, software cannot make system recover via repairing or changing components but re-designing. Software reliability cannot be guaranteed by redundancy, and methods to verify its reliability are not like hardware which has a complete theoretical system. In general, ensuring software reliability is more difficult than that of hardware. Even the software system in NASA, its reliability is an order of magnitude lower than hardware. Hence, software reliability seriously affects the reliability of the whole system. In order to improve system reliability, software reliability must be paid great attention to.

III. Review on Software Reliability

Phase one: 1950-1967 Subject sprout period, software reliability did not attract attention.

Phase two: 1968-1987 Subject formation period, Software Engineering was established and developed, mathematical models of software reliability began to emerge.

Phase three: 1988-now Subject developing period, Software Reliability Engineering was put forward, software reliability transits from theoretical research to engineering application. Increasingly importance has been attached to software reliability, our country promulgated GJB/Z 102-97 Software Reliability Security Design Criteria in the year 1997 as well [3].

IV. Development of Software Reliability

A. Abroad Development Status

1) Theoretical research

The first paper on software reliability is the Birth and Death Process raised by Hudon in 1967 [4], this model exported Weibull Distribution based on Mean Time between Failures (MTBF) [5].

From 1970s to early 1980s, the study of software reliability mainly concentrated in the comparison and selection of models, some famous models got used and improved. In this period, software reliability models had some characteristics such as correctness verification-oriented, using the stochastic modeling method, introducing statistical analysis technique to fault data, measuring software reliability by setting variables [6]. In 1972, Jelinski and Moranda proposed the famous Jelinski-Moranda model based on software failure rate, which appertains MTBF model and uses time dimension and maximum likelihood estimation [7-8]. Other models proposed later were mostly the improvements of J-M model by making the unreasonable hypothesis reasonable so that the model would be more realistic [9]. In 1975, Littlewood set software reliability model aimed at modular program [10], pointing that transmission and control among modules follow the Markov Process and it could be regarded a white-box model. In 1979, Goel and Okumoto introduced a software fault model using simple Nonhomogeneous Poisson Process (NHPP) [11], and made some progress to the previous models describing software failure process [6].

Since 1990s, rapid progress has been made on the study of software reliability, research includes software reliability design, reliability testing and management, collection of reliability data, reliability prognosis and reliability problems of hardware- software-hybrid systems [5,12].

Now, software reliability modeling is still a hotspot, e.g. reliability models based on neural network, reliability models based on Support Vector Machine (SVM) and other new kinds of models.

2) Engineering practice

Though the development status of engineering practice in software reliability is not so flourish as theoretical research, it has made some progress. On the website delphion one can get the patents authorized in the United States concerning software reliability, shown in the picture below:

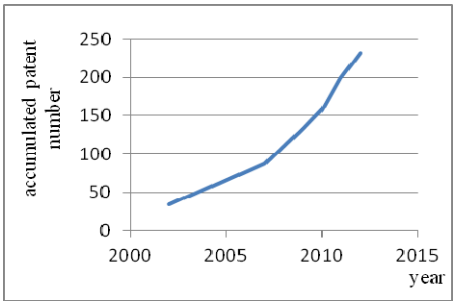

Figure 1. Growth trend of patent number

As is shown in Figure 1, the number of patents related to software reliability has a growing trend, that is to say theory has begun its march to practice.

Besides, more and more applications are made in production field about software reliability, especially in aerospace field and automobile industry. So far, reliability and safety management mechanism that faces the whole software life cycle systematically has been initially formed at abroad. Both NASA and ESA have published related assurance standard and instruction manual about software products. NASA requires that quantified risk analysis should be used to analyze the technical measures, reliability and safety of the mission-critical software, thus offering technical support for making decisions. As for the automobile industry, active methods are made to locate the problems of software quality, e.g. Motor Industry Software Reliability Association released the first edition of a series of recommended programming practices called MISRA-C: 2004 in the year 1998. 128 rules aimed at editing the C programing language more safely were included in this edition. From then on, these guiding lines have not only played an important role in automobile industry, but also permeated into nearly every application from aerospace to mining field.

Though the situation is optimistic, problems exist. The practice of software reliability engineering is not relatively independent from software engineering. There is no systematic approach and it brings considerable controversy in different software planning.

B. Domestic Development Status

1) Theoretical research

The study on software reliability established relatively late in China in the 1980s. Scholars like Xizi Huang,

Wangmei Chen, Kaiyuan Cai has made useful exploration to software reliability modeling and software reliability distribution and management. Breakthroughs has been made in software fault avoidance and tolerance technology, reliability assess tools, reliability test and reliability metrics [12,13]. In recent years, theoretical research on software reliability is flourishing, narrowing the gap with abroad.

Though domestic study on software reliability is booming, the disparities with abroad is obvious. The main gaps are as follows: a. Shortage on study powers; b. There is no powerful institution sparing no effort to support the study on software reliability; c. Software reliability modeling is the concentration while other aspects are very weak; d. High-level achievements which have international influence are rare [13].

2) Engineering practice

Yiping Yao used to use his own software reliability assessment tool to assess the software reliability of ACT verification aircraft. Kaiyuan Cai also applied his own fuzzy software reliability model to ACT [13]. In patent respect, patents related to software reliability are only eleven, which all applied after the year 2000. Nine of them were applied during 2010-2011. There is a long way from abroad since the number of patents is small and our engineering practice started late.

Since China Software Testing Center was found in 1990, each province has established its own software testing center gradually. Theses testing centers are used to test the quality of software, hardware and network security. During the development history of the past 20 years, China testing has formed a service system with vertical and horizontal integration, which covers the whole process of the project life cycle and makes significant efforts to guarantee software reliability. In the production process, quality of software is getting more and more valued.

Though progress has been made, domestic practice of software reliability still has a large blank. And just because of this, it has a huge potential.

V. EXISTING PROBLEMS AND FUTURE DIRECTIONS

A. Existing Problems

In spite of the breakthroughs made in the study on software reliability, many problems still exist [5,12,14,15]:

- Viewpoints, methods and tools

Now studies are mainly based on probability theory and mathematical statistics, which is not that proper. Software reliability theories and technologies need new mathematical tools, such as pattern recognition, artificial intelligence, petri net and so on. Besides, it requires nutrients from other branches of systematic science, especially high-level ones.

- Software reliability models

Hundreds of models established all have different extent of limitations. There is no recognized system for the classification of models and no universal analytical model.

- The application of software reliability models

The predicting outcomes are not consistent with each other when using different software reliability models. How to effectively put models into realistic software developing process is another problem.

- Data

Building software failure database to support software tests and collecting failure data automatically are problems needed to be solved.

- Generate software testing case automatically

Generating software testing case automatically in all kinds of software testing tools is waiting to be perfected.

- Hardware-software-hybrid system reliability

Software reliability framework can be made using mathematical methods similar to hardware. Failure Mode and Effects Analysis (FMEA) and Failure Mode Effects and Criticality Analysis (FMECA) perform well in hardware system, yet they are not enough for software system. The Fault Tree provides graphical and logical framework, which can offer a united modeling plan for hardware-software collaborative design. Modeling of hardware-software-hybrid reliability is one of the hotspots.

- Industrial practice

At present, software reliability engineering is not being widely used. The main reason is that the cost-effectiveness is unconspicuous. Many companies are not willing to put too much time and money into failure data collecting. Hence, reliability standard cannot be obtained, experience and lessons cannot be drawn from the former applications either. Usually the priority of reliability is lower than functionality and creativity in a product. When feel pressured for product release time, reliability is always the first property to be compressed.

- Software architecture

Fault isolation is the main consideration in designing software architecture. Lowering the dependency among different software blocks makes their reliability independent, so that they will not interact. New software architecture includes cross-platform technology, open-world software, service-oriented architecture and web application. Although there are some modeling methods to estimate the reliability of specific web systems, software reliability engineering technologies for general web systems and other service-oriented architectures need more research [16,17,18,19].

B. Future Directions

Software reliability modeling is becoming complete gradually. In order to make software reliability analysis and prognosis more accurate, people use correction, deviation rectification, weighted combination etc. to improve traditional models. Meanwhile, analyzing software reliability using artificial intelligence and simulation technique etc. are other notable trends.

Future directions of software reliability mainly include 5 aspects as follows:

- Software architecture

Accomplish software engineering based on components. Accomplish the reuse of software by taking advantage of the existing components.

- Software design

Accomplish software reliability design. Research is needed in the following stages: fault confinement, fault detection, diagnosis, reconfiguration, recovery, restart, repair and reintegration. Design for reliability techniques can be further pursued in four different areas: fault avoidance, fault detection, masking redundancy, and dynamic redundancy. Software design mainly considers cost-effectiveness, which calls for better reliability while no spare cost.

- Reliability testing

Bring software testing and software reliability together, so that the reliability can be accurately measured.

- Metrics for reliability prediction

We are supposed to better collect and use metrics via various tools. At the moment, metrics and data collection process is one-way and open-loop so that they cannot provide feedbacks. In the future, we expect close-loop process which can provide information to predictable reliable software.

- Reliability for specific software applications

In some specific domains such as the service industry, software plays such an increasingly significant role that its reliability requires insurance. Service-oriented design is also applied to Software Engineering. Moreover, open system approach is another trend in software applications.

VI. CONCLUSION

In recent years, study on software reliability has made considerable progress. However, the gap between abroad and domestic is not small, especially engineering practice. There is still a long way to go both domestic and abroad. So, opportunities and challenges coexist, this area calls for further developments and improvements.

ACKNOWLEDGMENT

This material is based upon work supported by the National Science and Technology Major Project “High-Grade CNC Machine Tools and Basic Manufacturing Equipments” (Grant No. 2011ZX04016-021) and the Special Items Fund of Beijing Municipal Commission of Education.

REFERENCES

[1] GB/T 11457-95 Software Engineering Terms.

[2] Yichen Wang. Test and Fault Diagnosis of NCS Software. Plant Maintenance Engineering 2005,2:36-37.

[3] Minyan Lu. Software Reliability Engineering. National Defense Industry Press, 2011.

[4] Hudon G R. Program Errors as a Birth and Death Process. Report SP-3011, SsntaMonica, CA:System Development Corporation, 1967.

[5] Yun Liu, Wei Zhao. Research and Progress in Software Reliability. Microcomputer Development, 2003(2):1-15.

[6] Bangqing Qiu. Model Study and Overview of Software Reliability Abroad. Quality and Reliability, 1994(1): 14-18.

[7] Z. Jelinski, P. Moranda. Software Reliability Research: Statistical Computer Performance Evaluation. N.Y. and London: Academic Press, 1972: 465-484.

[8] Yong Cao. Software Reliability Model Based on Fractal and Mathematic Mechanization of Program Correctness Proof. University of Electronic Science and Technology of China, Doctoral Dissertation, 2010.

[9] Ming Han. Software Reliability Models. Beijing University of Technology M.E.Dissertation, 2007.

[10] B. Littlewood. A Reliability model for systems with Markov structure. Applied Statistics, 1975, 24: 172–177.

[11] A.L. Goel, K. Okumoto. Time-Dependent Error Detection Rate Model for Software and other Performance Measures. IEEE Transactions on Reliability, 1979, 28(3): 206-211.

[12] Feng Yin. The Development Situation and Prospect of Hot Technique Problems of Software Engineering. Journal of Changsha University, 2006, 20(5) : 45-49.

[13] Kaiyuan Cai. Software Reliability: A Personal View.Systems Engineering and Electronics, 1993, (4) : 47-54.

[14] Michael R. Lyu. Software Reliability Engineering: A Roadmap. FOSE '07 2007 Future of Software Engineering, IEEE Computer Society Washington, DC, USA. 2007: 153-170

[15] Yanyan Zheng, Wei Guo, Renzuo Xu. Overview of Software Reliability Engineering.Computer Science, 2009, 36(2) : 20-25.

[16] Bishop J, HorspoolN. Cross-Platform Development: Software That Lasts. IEEE Computer Society, 2006, 39(10):26-35.

[17] Baresi L, Nitto E, Ghezzi C. Toward Open-World Software: Issues and Challenges. IEEE Computer Society, 2006, 39(10):36-43.

[18] Margaria T, Steffen B . Service Engineering: Linking Business and IT.IEEE Computer Society, 2006, 39(10):45-55.

[19] Wenli Wang, Meihui Tang. User-Oriented Reliability Modeling for a Web System. Proceedings of the 14th International Symposium on Software Reliability Engineering (ISSRE’03).Denver, Colorado, November 2003:1-12.