Summary of software reliability Research

Shizhuang YIN, Quan SHI, Yadong WANG and Cai CHEN

1 Shijiazhuang Campus of Army Engineering University, Shijiazhuang, Hebei Province, 050003
E-mail: 18525742489@163.com

Abstract. This paper summarizes the main contents of software reliability research. With the rapid development of information technology, software plays an increasingly important role. Along with its complexity increasing software faults are becoming more and more. In some key fields due to software failures may even lead to unacceptable consequences. Therefore, it is necessary to study software reliability. Through reading lots of literatures summarizing researches on domestic and foreign scholars this paper puts forward some main problems which need further discussion in software reliability research.

1. Introduction

With the wide application of computer and information processing, reliability problems of computer systems have been paid more attention to. Software has been closely integrated with our lives. It not only provides convenience for our daily life but also plays an important role in aerospace transportation military industry control and communication etc. Moreover, software reliability is required in these highly reliable and complex systems, otherwise any minor software failure may bring serious consequences. According to statistics, software faults account for 65% faults in computer systems. For this reason, America has proposed "making computers uninterrupted and repair continuously for only one minute in 200 years average time", and also put forward that "High reliability computer is one of main directions of computer development in 21th century" [1]. To achieve this goal, its primary problem is improving software quality. Meanwhile, during studying system reliability, the reliability of software system reliability is not considered, and its result is quite different from actual situation. Therefore improving software quality reducing software failure occurrence and development has become an urgent problem urgently researching how to diagnose software fault has very important significance.

2. Concept of software reliability

Software reliability refers to the ability of software to complete defined functions under specified conditions and specified time. Software reliability is one of the most important inherent characteristics of software. Compared with hardware reliability software reliability has following characteristics:

(1) software errors are mainly caused by design errors, while usage and maintenance have little impact on software reliability.
(2) software errors do not increase with longer usage hours.
(3) software repair is designed to eliminate errors by redesign.
(4) software reliability growth is independent of software usage time, and related to detecting and correcting errors.
(5) there is no aging or loss phenomenon during software life span.
(6) correct software does not change its correctness due to changes in usage environments.
(7) redundancy technology of same software cannot improve software reliability.
(8) using various library functions provided by software provided by high reliability standard parts—system software, it doesn't necessarily improve reliability of whole software.

The probability of software failure differs from traditional mode. Figure 1 shows six different types of equipment failure rates varying over time. Software failures vary depending on environment and users, and most failures occur due to human causes.

![Figure 1. Six basic forms of failure rate curve.](image)

3. Development and current situation of software reliability Research

The first paper on software reliability is a birth death process model proposed by Hudon 1967 [2] which derives Weibull distribution of fault intervals. After Hudon Jelinski et al studied software failure rate on 1972. They assumed that there was a Piecewise constant of fault which was proportional to residual error number. Most parameters estimation in above models adopts classical statistical method. However, lots of facts indicate that when the distribution deviates from the original assumption, the parameter estimation values obtained by classical statistics are often unreliable. Therefore, in 1977, J.W.Tukey proposed exploring data analysis method (EDA), which distinguishes classical statistical methods from few assumptions or assumptions at all, emphasizes on mathematical structure of problem, emphasizes directly studying data itself, understanding data features and structures, then analyzes how to obtain more information and then uses suitable estimates. From late 70s to early 80s, software reliability research began to focus on comparing and selecting software reliability models. Shortcomings in early work are lack of better fault data and comparative criteria needed.

Currently, software reliability research mainly includes following aspects.
(1) why software fails or fails.
(2) what specifications should be complied with in designing, developing and developing software.
(3) how to test whether or not it meets reliability requirements developed software. Among them include data collection processing software testing software reliability model establishment software reliability prediction etc.
(4) from the viewpoint of software engineering economics, considering the tradeoff between software reliability and software testing, maintenance and management expenses.
(5) how to evaluate their effectiveness and prediction ability on existing software reliability models.

Currently, the main achievements of software reliability research are:

3.1. Software reliability design
In software design process, in order to improve software reliability, some specialized technologies are often used to design reliability into software. There are 2 kinds of technologies which improve software reliability: one is avoiding faults namely faults and errors are not infiltrated into software during development; another is fault tolerant technology. Among them are recovery block technology and version programming technology. The fault tolerant techniques of software [3] are considered
from reliability optimization cost optimization etc. from others. Based on these techniques Belle et al. propose consistency recovery block acceptance vote design of self-checking program and fault tolerance technology considering failure correlation between versions [4] but currently these technologies are still theoretically studied and engineering practice still needs further discussion.

3.2. Software reliability testing and management
By testing software reliability we find faults in software so as to eliminate faults continuously so as to improve software reliability. Traditional software testing methods include program-oriented static test and dynamic testing 2 categories. In 1993, Whittaker et al. studied software testing process and test case using Markov process. "best" is measured by reliability obtained after all tests fail to eliminate. This paper presents a comparatively economical method for evaluating software reliability after modification under certain ordinary environments [5]. With the development of object oriented programming (OOP) technology, object oriented software testing technology is developed correspondingly. Firstly, this technology is proposed by M.J.Harrold et al. Firstly, it finds out definition-reference relation between service inside and service, determines test case around definition-reference correlation. Aiming at most OOP tests currently focused on class testing document [6] proposes a systematic level OOP software testing method by establishing a state model to obtain software dynamic behavior. Software reliability management includes software testing management and distribution management. In software testing management firstly we examine the problem of schedule status management in test engineering secondly investigate optimal allocation problem of testing manpower on software module testing. Currently, it focuses on human resource allocation problem under certain reliability requirement (or making reliability maximum). On the other hand considering that optimal release time of software is related to parameters in software reliability model estimation of these parameters is usually not accurate due to the impact of software failure data collected.

3.3. Collection of software reliability data
Software reliability data collection is basis for estimating software reliability. Whether data collected is valid or not meets the requirement of model directly affects accuracy and reliability of software reliability evaluation. Generally collected software reliability data include 2 classes: failure count data and failure interval data. However due to a similar error in software may lead to many failures sometimes there exists correlation between several errors thus reducing data accuracy thus detailed plans and standards should be formulated when collecting software reliability data such as human resource allocation time allocation data form recording mode storage mode etc. Secondly, we can apply data for specific analysis and processing, such as data extraction, merging, correlation analysis etc. Moreover when applying a specific software reliability model it may occur that one kind of failure data may occur while another class is collected then conversion between invalid data is needed. Although some supporting tools have been developed automatically for software reliability data collection however it is limited greatly so how to automatically collect software reliability data accurately and efficiently is still a topic which needs further research and practice.

3.4. Software reliability prediction
Traditional software reliability prediction model is used to predict failure number of software in future time by fitting software failure data collected by large software. Considering that different periods software failure data affect software reliability prediction differently, we propose a dual exponential smoothing software reliability prediction model which gives a larger weight value for recent observed software failure data. Compared with traditional prediction model, this model is simple and accurate. Karunanithi et al. proposed using neural networks to solve this problem during IEEE software reliability engineering international conference in 1991 and 1992 respectively. During training artificial neural network stage using execution time as input while observing cumulative error number
as target output input an additional future time interval at completion of training and investigate artificial neural network estimate accuracy of cumulative error number.

Reliability usually uses risk functions and reliability functions to denote their general formulae as shown in formulae 1 and 2 respectively:

In risk function, T indicates failure time, P is probability function, \( z(t) \Delta t \) represents probability of failure within time interval \((t, t + \Delta t)\).

\[
 z(t)\Delta t = P\{t < T < t + \Delta t \mid T > t\}
\]

Reliability function \( R(t) \) gives probability of failure within time interval \((t, t + \Delta t)\)

\[
 R(t) = e^{- \int_{b}^{t} z(x) \, dx}
\]

3.5. Reliability problem of hard-software hybrid systems

Reliability problem of hard-software hybrid systems. Computer systems are composed of hardware and software. Hardware and software interact each other. Therefore, to evaluate computer system reliability, we must understand problems from the viewpoint of hard-software integrated system. Since the middle of 70s, foreign countries began to study reliability problems of integrated systems. 1983 ARMS meeting began to "composite hard / software reliability" as one of the conference topics. In the same year, "system reliability Seminar" was also held, and foreign research achievements have been applied in aviation fields. Since the middle of 80s, scientific workers in China have also invested this research and application. Several common reliability models of hard-software hybrid systems include: Markovian Markov model Thompson model Shooman model IBM model and Iyer-Tang model etc. Most of these models assume that systems contain hardware subsystem and software subsystem so when any subsystem fails it will cause system failure.

Domestic software reliability research began in 80s. It explored software reliability modeling software reliability allocation software reliability management etc. Researches on software fault avoidance techniques etc. were studied; etc. Software reliability modeling (proposed fuzzy software reliability model) software reliability model application selection software reliability testing software reliability measurement system is discussed in detail. Xu Renzuo, Li Deyin, etc. are researched on software reliability theory. In engineering practice, Mr. Yao Yiping has used his software reliability evaluation tool to evaluate software reliability of ACT verification machine. Mr Cai Kaiyuan also applies his fuzzy software reliability model to ACT verification machine [7]. CLIPS developed software reliability expert system using CLIPS, which provides strong support for selecting and comparing software reliability models [8].

4. Software failure Law and failure Analysis

The main purpose of studying software reliability is to ensure and improve software reliability. Firstly, we should deeply understand software failure mechanism so that we can find correct research direction. Software failure arises from error which inevitably occurs during software development. Fault is generated when one or several software defects are activated during software running. If there isn't effective fault tolerance measures after software failure will lead to failure.

The model generated by its fault is shown in figure 2.
4.1. **Internal factors.**

With the expansion of software functions and complexity of computer control objects, software system scale becomes larger and larger, combined with software itself has complexity of path complexity and complexity of state complexity, so designing such complex systems inevitably leave some errors, resulting in software defects, specifically:

1. errors occurred during software design, preparation and debugging;
2. errors arising from software transplantation modifications;
3. approximation of algorithm leads to imprecise variable region;
4. deficiencies in data structures;
5. there exists specific points and critical values in procedure;
6. failure of privileged user operation causes system software errors or partial function failure.

4.2. **External factors.**

Usually software is tested in development environment but actual software operating environment is not exactly identical with software development environment so there exist hidden troubles of various faults. Moreover, due to complexity of software condition and complexity of path, it is almost impossible to test software in test process. So even if we do lots of tests on software, software defects cannot be completely detected or eliminated, software still may fail, generally there are several kinds:

1. hardware failures lead to software failures such as error data causing software faults;
2. external or internal electromagnetic interference causes loss or error of program data such as potential reversal due to space particles in registers;
3. when system environment changes, failure or function failure caused by program cannot change accordingly;
4. failures caused by operating errors caused software damage or error such as user deleting system files etc.

5. **Problems existed in software reliability Research**

Since software reliability research has been done many experts scholars at home and abroad have proposed many feasible methods but there are some problems in some extent:

1. the current research is mainly based on probability theory and mathematical statistics which is not always appropriate. Software reliability theory and technology need to be developed greatly,
requiring new mathematical tools such as pattern recognition artificial intelligence, PETRI network etc.

(2) Software reliability model problem. Although hundreds of software reliability models have been established at present, there are some limitations, so how to establish reasonable and practical software reliability model is still needed further research from problems such as rationality of software reliability model, whether practical application is simple and convenient, applicability scope is wide.

(3) Data issues. Software data collection is a difficult and tedious task. So far there has not been established a database for testing software reliability models proving their accuracy availability and differences between models and models. Therefore, software failure database establishment and automatic collection of software failure data are urgent problems urgently needed to be solved.

(4) Reliability problem of hard-software hybrid systems. Hard software faults are produced differently in computer systems because of incomplete reliability and reliability of software. Although some hard-software reliability models have been established at present, although these models are proposed for specific problems, they are different in focus, thus problems of model are also discussed.

6. Conclusions
Recently software plays more and more important role in product function and software reliability research gets more and more attention. However, current research mainly based on probability theory and mathematical statistics, hypothesis conditions are too harsh, adopting current emerging pattern recognition artificial intelligence techniques to make assumptions more reasonable, easier application scope will be more research direction in future. Meanwhile software reliability is a systematic project which needs to consider data collection processing software testing software reliability model establishment software reliability prediction etc. Moreover in order to improve reliability of equipment software reliability and hardware reliability should be combined together.

Reference
[1] Zheng Jun 1995 Development and problems of software reliability modeling Computer Science 6 78-4
[2] Yang Chaohong and Wang Huiyu 2018 Software reliability Research Information Communications 2018(08) 13-29
[3] Belli F, Jedrzejowicz P 1990 Fault-Tolerant Program and Their Reliability IEEE Trans.Relia 39(2) 184-92
[4] Liu Xicheng and Zhong Wanyi 1997 Software reliability Engineering Handbook (Beijing: electronic industry Press)
[5] Zheng Guoliang and Shao Weizhong 1997 Review of software Engineering Conference International academic developments 1997(10) 24-8
[6] Chang K H, Liao S, Seidman S B and Richard Champman Testing Object-Oriented Programs: from Formal Specification to Test Scenario Generation The Journal of System 2 84-92
[7] Cai Kaiyuan 1995 Software reliability Engineering Foundation (Beijing: Tsinghua University Press)
[8] Xu Renzuo 1996 Software reliability expert system (Beijing: Tsinghua University Press)