2011 International Conference on Computer and Software Modeling
IPCSIT vol.14 (2011) © (2011) IACSIT Press, Singapore

A Software Reliability Evaluation Method

Qiuying Li and Haifeng Li

School of Reliability and Systems Engineering, Beihang University, Beijing, China

Abstract. Software reliability evaluation plays a very important role in the development of software, but the traditional software evaluation method mostly focuses on evaluation by use of failure data which is gained only after testing or usage in the late phase of the software life cycle. Thus people hope to get every stage's information about the software's reliability which is taken as the reference or accordance to guide the software's design, analysis and testing and so on. A software reliability evaluation method is put forward in this paper, which focuses on lots of information correlative with reliability during the whole software life cycle. Finally an application is put forward to demonstrate the feasibility of this method.

Keywords: software reliability, reliability evaluation, reliability measurement, measurement factor

1. Introduction

Software reliability evaluation is playing an important role in software reliability engineering, which can give information taken as the reference or accordance to guide the software's design, analysis and testing and so on. Finally it will provide the quantitative estimation result for the issued software product.

In recent years, software reliability evaluation based on failure data has been deeply developed, as the main means of software reliability estimation, lots of software reliability growth models have been proposed[1-5]. But with the shortcoming of not very good evaluation quality, many new models and technique were proposed to effectively improve the reliability estimation performance, such as Neural-Network-based model presented by N.Karunanithi[6], chaos deduce model[7], Bayes networks model[8], fuzzy theory model[9] and so on. New technologies are also proposed, such as the failure data trend analysis and prediction quality improvement[10-11]. Based on the statistical theory, David et al.[12-14] proposed several software reliability assessment methods which established the sampling theory for software reliability evaluation. With the shortcoming of only applied in the late phase of the software life cycle, such as testing and maintenance process, its application is hindered. Whether it can be used in the early phase of software development becomes a hot and hard question. Li et al[15-17] provided the software reliability comprehensive evaluation method from the view of system theory. IEEE standard 982.1 stated that various factors related to the product, process and resource of software in the whole software life cycle will have great influences on software reliability[18]. Reference [19] presented the findings of empirical research from 13 companies participating in software development to identify the factors that may affect software reliability. Some software engineering experts want to use commonly used software metrics to predict software reliability in a direct way, thus whether these can give new ideas for software reliability evaluation?

2. Analysis of software reliability evaluation factors

Reference [19] presents the findings of empirical research from 13 companies participating in software development including AT&T, BellCore, Chrysler and MCI International to identify 32 factors that may affect software reliability. These factors are analyzed and ranked in terms of their impact on software

Corresponding author. Tel.: + 8601082339169; fax: +8601082313913.
E-mail address: li_qiuying@buaa.edu.cn.

reliability. Based on [4] and [17], this paper presents 30 ordinary factors as the objectives to be further studied, which is shown in Table 1, also these factors are classified into product factors, process factors and resource factors.

- Product factors: Product factors are software attributes or characteristics which have something to do with software size, document and structure. That is to say, it is a static parameter which can be collected from software design documents etc.
- Resource factors: Resources factors can be divided into three classes: human, reusable software component, software and hardware environment.
- Process factors: Process factors refer characteristics and behaviors in every stage of software development.

3. Evaluation method

Here the software reliability evaluation method refers well-known hardware reliability demonstration method[20], whose principle is to use factors' information to give an auditing for reliability, then according to grading scores to calculate the evaluation value. The steps are given as follows: ascertain factors used for auditing; list the detailed contents required auditing; design expert grading table, collect information of factors and choose software experts; organize experts to audit and grade according to information and contents; combine the experts' grading results to obtain the integrated evaluation results.

Thus software reliability evaluation is a process of auditing and demonstration, which not only presents evaluation results but also finds the latent defects in the process of software development and testing to guide and improve the process accordingly.

3.1. Ascertain factors for auditing

Factors listed in Table 1 can be regarded as objectives for auditing. Before evaluation, factors can be chosen as the auditing set according to the reality requirement. For example, in the early stage of development, factors affect software reliability can be chosen just involved in that stage.

3.2. Ascertain contents to be checked

Contents to be checked are those requirements for factors involved in software design, development and testing process and characteristics of software itself, e.g. software reliability design method as fault-tolerance, fault-avoidance and control of software complexity. Also the contents should be simple and accurate, distinctive and covering all characteristics. Some examples are shown in Table 2.

3.3. Design grading table for expert auditing

Based on hardware reliability qualitative demonstration method[20], the following four type grades are given as follows: excellent-the work underlying has been done very well and fulfilled the requirement completely even exceed the demand to some extent, which is equal to 90-100; good-the work underlying has been done well and fulfilled the requirement except some mistakes without affecting software reliability heavily and could be corrected easily, which is equal to 75-90; normal-the work underlying has been done and fulfilled the requirement primarily except a lot of serious mistakes are made affecting software reliability to some extent and ought to be corrected with lots of efforts, which is equal to 60-75; bad-the work underlying has been done badly and not fulfilled the requirement with a lot of serious mistakes affecting software reliability severely, which ought to be done over again to avoid more severe losses, whose score is below 60. Expert auditing grading table can be designed as Table 2.

3.4. Auditing and grading

- Hypothesis and notation:Suppose m is the number of factors and n is the number of experts. E_i is the ith considered factor, where $i = 1 \sim m$ .Let p_i be the number of contents under E_i , S_{ij} be the jth content of factor E_i , therefore $j = 1 \sim p_i$ .Suppose every expert has the same significant impact, then let weight for E_i is w_i and weight for every expert is 1/n.Let u_{ijk} be the score of S_{ij} given by the kth expert, therefore U_{ik} be the score of E_i given by the kth expert.
- Grading method:Grading score is an arbitrary real number in [0,1]. To express clearly, the grading result is expressed as a natural number in the interval [0,100].Let the highest software reliability is

100. Many factors will be used for evaluation and based on software itself and all factors' impact, every factor is assigned a highest score as the basis to integrated marks. Let A_i be the basic mark of E_i where $\sum_{i=1}^{m} A_i = 100$ .The simplest method of ascertaining basic score is in terms of weight as follows

$$A_i = w_i \times 100 \tag{1}$$

Take E_i for example, expert k gives a mark u_{ijk} after careful auditing for every content, then expert k gives a total mark for E_i , i.e. U_{ik} is：

$$U_{ik} = A_i \times \frac{\sum_{j=1}^{p_i} u_{ijk}}{p_i \times 100} \tag{2}$$

By Eq. 2, one can get the total mark U_i for E_i by all experts such that

$$U_i = \sum_{k=1}^{n} \frac{1}{n} U_{ik} \tag{3}$$

By Eq. 3, one can get the final grading score U_i for factor E_i .

3.5. Software reliability integrated evaluation result

The integrated evaluation result of software reliability R is as follows:

$$R = \sum_{i=1}^{m} U_i \tag{4}$$

04-grading method is used to give the weight of factor E_i .The principle of 04-grading method is given as follows:1)Expert grades separately without discussing between each other;2)When the significance of two factors compared, the comparing method is adopted and the following three steps can be used such that:i)Between two factors, the more important one gets the score of 4 and the other gets 0;ii)Between two factors, the relatively important one gets the score of 3 and the other gets 1;iii)If two factors are of the same importance, they all get 2;3) The two factors cannot both get 4 or cannot both get 0.Examples are given as Table 4.

4. Application

Here gives a simple application.

- Ascertain factors for auditing：For convenience, take 7 factors from Table 1 noted as A, B, C, D, E, F, G for example.
- Ascertain contents to be checked：The contents to be checked are shown as Table 2.
- Design grading table for expert auditing：The table is shown as Table 3.
- Auditing factors：Suppose seven software experts are invited to audit 7 factors and the weights are calculated as Table 4. E.g. expert A gives an auditing result as which is shown in Table 5.Take factor A (software complexity) for example, the score from expert A is computed as follows, then final result is shown as Table 6.

$$U_{A1} = 20 \times \frac{90+85+80+80+90}{5 \times 100} = 17.1 \tag{5}$$

- Result of integrated reliability evaluation： In terms of the results in Table 6, by Eq.4 ,then

$$R = \sum_{i=1}^{m} U_i = 88.3 \tag{6}$$

5. Conclusion

A software reliability evaluation method based on influencing factors was put forward. By means of experts grading and auditing, in terms of contents to be checked, the integrated evaluation result was achieved. From the practice of application, not only evaluation result can be obtained, but also guidance can

be put forward to improve the process of software development and testing and so on, which would play an important role in software engineering practice. More suitable factors, relationship between classification of factors and weights, contents to be considered should be further studied in future work.

Table 1 List of ordinary factors

Type	No	Factor Name	Type	No	Factor Name
Product factors	1	software complexity	Process factors	16	testing coverage
	2	percentage of reused code and modules		17	test case
	3	user's quality objective		18	fault detecting and removing process
	4	software programming languages		19	test efforts
	5	nature of defects and failures		20	documentation
	6	programmer's skill		21	design methodologies
Resource factors	7	development team size		22	software development design methodologies and technology
	8	programmer organization		23	software operation
	9	user's skill		24	development management
	10	testing tools		25	work standard
	11	testing environment		26	relationship of detailed design to requirements
	12	hardware resource		27	frequency of program specification and requirements change
	13	software development environment		28	difficulty of programming
	14	testing resource allocation		29	whole schedule
Process factors	15	testing methodologies		30	programming effort

Table 2 Contents of factors to be checked

Factor	Contents to be checked
software complexity	Does software have a good architecture system
	percentage of defects caused by source code size
	software McCabe's complexity
	software functionality
	code readability

Table 3 Grading table for factors' contents

Factor	Contents to be checked	Grading ranks				Grading results
		excellent	good	normal	bad	
software complexity	Does software have a good architecture system	√				90
	percentage of defects caused by source code size		√			80
	software McCabe's complexity		√			80
	software functionality		√			85
	code readability	√				90

Table 4 Statistical results by 04-grading method

Factor	A	B	C	D	E	F	G	Grading score	Weights
A	--	4	4	3	3	2	1	17	0.20
B	0	--	3	2	2	0	0	7	0.08
C	0	1	--	1	2	0	0	4	0.05
D	1	2	3	--	2	1	0	9	0.11
E	1	2	2	2	--	1	0	8	0.10
F	2	4	4	3	3	--	1	17	0.20
G	3	4	4	4	4	3	--	22	0.26
Amount	--	--	--	--	--	--	--	84	1

Table 5 Grading table of expert A

Factors	Contents to be checked	Grading ranks				Grading results
		excellent	good	normal	bad	
software complexity	Does software have a good architecture system	√				90
	percentage of defects caused by source code size		√			80
	software McCabe's complexity		√			80
	software functionality		√			85
	code readability	√				90

Table 6. Grading results of all experts

Factor	Expert 1	Expert 2	Expert 3	Expert 4	Expert 5	Expert 6	Expert 7	Integrated grading score	Basic score
A	17.1	17.8	18.8	18.1	18.7	17.6	15.9	17.7	20
B	6.5	6	6.3	6.8	7.1	5.7	6.7	6.4	8
C	4.3	3.9	4.1	4.4	3.8	4.1	4.3	4.1	5
D	9.2	9.9	9.8	9.4	10.1	9.2	10	9.7	11
E	9	9.1	8.8	8.6	9.5	9	8.8	9	10
F	16.4	18.2	18.9	17.3	19.1	18.4	17.4	18	20
G	21.5	23.6	23.2	24.6	22.7	23.2	24.8	23.4	26
Amount	--							88.3	100

6. References

[1] R. Z. Xu, M. Xie and R. J. Zheng. *Software Reliability Model and Application*. Beijing: Tsinghua University Press, 1994(in Chinese).

[2] S. Yamada Osaki. Software reliability growth modeling: Models and applications. *IEEE Trans. Software Engineering* SE-11(12), 1431-1437, 1985.

[3] J. D. Musa, A. Iannino and K. Okumoto. *Software Reliability: Measurement, Prediction, Application*. Mc-Graw-Hill, New York, 1987.

[4] H. Pham. *Software Reliability*. Springer-Verlag, Singapore, 2000.

[5] S. Yamada. *Software reliability models In Stochastic Models in Reliability and Maintenance*. Springer-Verlag, Berlin, 2002, 253-280.

[6] N. Karunanithi, D.Whitley and Y. K. Malaiya. Prediction of Reliability Using Connectionist Models. *IEEE Transactions on Software Engineering*, 18(7), July 1992, 563-574.

[7] F. Z. Zou. *Software reliability theoretical analysis and computation*. Ph.D thesis of Wuhan University of Hydraulic and Electric Engineering, 1999(in Chinese).

[8] C. G. Bai.*Software reliability research based on Bayesnetworks*. Ph.D thesis of Zhejiang Univers,1999(in Chinese).

[9] J. H. Guo,X.Z.Yang and H.W.Liu.Software Reliability Nonlinear Modeling and Its Fuzzy Evaluation.*4th Wseas Int. Conf. on Non-linear analysis, non-linear systems and chaos,* Sofia, Bulgaria, October 27-29, 2005,49-54.

[10] B. Littlewood. *Forecasting Software Reliability.* CSR Technical Report,1989.

[11] S. Brocklehurst, P. Y Chan and B. Littlewood. Recalibrating Software Reliability Models. *IEEE Trans.on Software Engineering*, SE-16(4), 458-470, Apr.1990.

[12] L. P. David, A. John, S. P. Kwan. Evaluation of Safety-critical Software. *Communication of ACM*, 1990(6).

[13] D. L. Parnas, G.J.K. Asmis, J. Madey. Assessment of Safety-critical Software in Nuclear power plants. *Nuclear safety*, 1991(2).

[14] W. H. Howden. Good enough versus High Assurance Software Testing and Analysis Methods. *In Proc. of IEEE HASE*, 1998.

[15] H. F. Li, M. Y. Lu, Z. X. Wang and Z. Li. Framework for software reliability comprehensive evaluation based on grey system theory, *Journal of Beijing University of aeronautics and astronautics*,34(11), 2008(in Chinese).

[16] B. Liu, M.Y. Lu and L. RUAN. Early Software Reliability Prediction: an Approach Based on Fuzzy Neural Network *Journal of Beijing University of aeronautics and astronautics*,2001,27(2) (in Chinese).

[17] T. J. Wang and M. Li. A Fuzzy Comprehensive Evaluation Model for Software Reliability. *Computer engineering and applications*, 2002, 38(20) (in Chinese).

[18] IEEE Std 982.1-1988, IEEE Standard Dictionary of Measures to Produce Reliable Software, IEEE, 1988.

[19] X. M. Zhang, Hoang Pham. An analysis of factors affecting software reliability.*The journal of systems and software*, 2000, 50,43-56.

[20] *RMS qualitative demonstration method*. National defence technology report, 2004(in Chinese).