Advances in Engineering Research, volume 139

2nd International Conference on Electrical Engineering and Automation (ICEEA 2018)

Research on Reliability Analysis of Component-Based Software

Wenna Ren and Baolong Guo*

Institute of Intelligent Control and Image Engineering of Xidian University, Xi'an 710071 China

*Corresponding author

***Abstract*—Component-based software development method has gradually become the mainstream software development method, with the rapid development of component technology, the software development method of complex software system with component design is more and more mature, but the software reliability analysis technology based on this technology is comparatively backward. At present, the component software reliability model does not consider the influence of component importance and complexity factors on software reliability, and presents the complexity analysis model of component software and the reliability evaluation model of important analysis model. In component-based software system, the granularity of component is usually less than the module, so based on the structure of component software, the complexity measure model with functional granularity is established, and the important measurement model is established according to the level, on the basis of which the state transition process between components is described as Markov process, This method calculates the reliability system-level reliability activity of software system by integrating component-level reliability evaluation. The feasibility of this method is validated by real time software, which is of great significance to the reliability evaluation of software development and the prediction of software reliability in the later stage.**

***Keywords*—component; reliability evaluation; markov process; state transfer**

I. INTRODUCTION

Component-based software development method has gradually become the mainstream software development method, so the reliability evaluation of component-type software system has been paid more attention [1-2]. From the perspective of software development technology, the emergence and rapid development of object-oriented technology, as well as the extensive application of web-based development, make the development mode of modular and component-based software accelerate [3]. It derives the Component-based software CBS (component-based Software) and development method, and the frequency and importance of each component is different, and it needs to evaluate the software reliability from the aspect of software architecture and software component [4]. So far, component-based software development technology is not mature, still lacks the effective method of reliability evaluation and prediction of component software system [5]. How to ensure the stable quality of software and evaluate the reliability of component-based software system is a long and complicated task.

The reliability evaluation methods of component software systems generally have three kinds of forms[6]: Path-based models, operational profiles-based models and State-based models. The path based method usually calculates the use quantity and probability of each component in the path of the test case after the system is constructed, and the reliability of the software is calculated[7-8]; Based on the description of the input domain of component software system, the reliability of different input domain and single component in different context is also inconsistent, the reliability of the system is calculated by calculating the probability of different sections and the migration probabilities between components, so as to evaluate the software reliability[9]; Based on the State method, the implementation process of component software system is regarded as the state transition process between components, and the Markov chain is constructed by using stochastic process theory, and the reliability of component system is calculated[10]. There are two kinds of stochastic process models at present, namely Markov process model[11] (Markov process MODEL,MPM) represented by J-M (Jelinski, Moranda) model and nonhomogeneous Poisson process model[12] (Nonhomogeneous Poission Model, NPM) represented by the G-O (Goel, Okumoto), also has a small range of use of neural network methods to measure the reliability of the software, the previous feed neural network[13] or support vector machine measurement method[14] is represented, but the application scope is small.

At present, the reliability evaluation model of the main software system [15] only considers the factor of component occupancy rate, has one-sidedness, and does not consider the importance of component complexity and importance to the reliability of software system. Component structure is more and more complex, some important components, although the use rate is not high, but its failure will have a significant impact on the software system [16]. This paper was based on the Markov stochastic process idea and considering component occupancy rate, failure rate, complexity and importance factors, the reliability evaluation model of two cascade of component and software system is established. The reliability index of Component software system is calculated by analyzing method. Finally, the model is validated in the software reliability analysis of a component, which can provide a good basis for software reliability growth, software quality improvement and component optimization in the software application process.

II. RELATED WORK

Reliability modeling representation, measurement calculation, prediction, assurance and evaluation are the basis of software reliability analysis. Reliability can be divided into two levels, first of all, the so-called component reliability (reliability of component). That is, the product is disassembled into a number of different parts or components, the reliability of these components are studied first, and then the whole system, the overall reliability of the product, that is, system reliability (reliability of systems). The method of component reliability analysis [17], in fact, statistical analysis, as for system reliability analysis, more complex, the method can be adopted more, can be distributed according to the importance of reliability, according to the complexity of the distribution of reliability, according to the technical level, task conditions, such as the overall distribution of reliability, or the relative failure rate distribution of reliability.

A. *Markov Model*

Supposing that the parameter $T=\{0,1,2\cdots\}$ of random process $\{X_n, n\in T\}$ is a discrete time set, may take the whole composition of the State space as $I=\{i_1,i_2,i_3,...\}$, if for any integer $n\in T$ and any $i_0,i_1,...,i_{n+1}\in I$, the conditional probability satisfies , so called is a Markov chain, the upper formula is the mathematical expression of the no-validity of the Markov chain.

Assuming $P^{(n)}$ represents n-step transfer probability $p^{(n)}{}_{i,j}$ matrix, state space $I=\{1,2,...\}$, then the $P^{(n)}$ is called the N-step transfer probability matrix of the system state. If each row of the square p is a probability vector, it is called the matrix of probability. The N-Step transfer probability of Markov chain is completely determined by one step transfer probability. $p_{ij}^{(n)}=P\{X_{m+n}=j\,|\,X_m=i\},(i,j\in I,m\geq 0,n\geq 1)$ is the N-step transfer probability of Markov chain $\{X_n, n\in T\}$.

Suppose the vector v represents the probability vector, $V^{(0)}=\{p_1,p_2,...\}$ is the initial probability vector, $V^{(n)}=\{p_1(n),p_2(n),...\}$ is the absolute probability vector of n moment, the absolute probability $V^{(n)}$ has the nature of the N-Step transfer probability, we can determine the unique Markov chain by the initial probability vector and the one step transfer probability matrix.

B. *Reliability Measurement Parameters*

Introduce the software reliability measurement parameter [18], which is a quantitative evaluation of software reliability and a necessary parameter for evaluating software reliability.

Define 1: Failure rate

The failure rate is the probability that the T time has not yet failed, and the time after that time has failed. Generally written as λ, it is a function of time t, so it is written as λ (t), called the failure efficiency function, sometimes also called the fault rate function or the risk function.

$$\lambda(t)=\frac{f(t)}{R(t)}=-\frac{dR(t)}{R(t)dt}$$

Among them, $R(t)$ is the software reliability, that is, the software in the specified time no failure occurred probability, through $R(t)$ can be obtained $f(t)$, $f(t)$ is the failure probability density function.

The relationship between failure efficiency and reliability can be expressed as $R(t)=\exp\left\{-\int_0^t \lambda(t)dt\right\}=e^{-\int_0^t \lambda(t)dt}$,

When $\lambda(t)=\lambda$ is constant, $R(t)=e^{-\lambda t}$.

Define 2: Occupancy rate

γ_j represents the proportion of component J being executed in a task, it satisfying the below equation:

$$\gamma_j=\sum_{i=1}^{n} v_i p_{ij}$$

Where v_i represents the probability that the component I executes, p_{ij} represents the probability that the component executes to the component J.

C. *Component Reliability Model*

Failure efficiency is an important parameter of component reliability evaluation. There are two states of normal and failure in the component. In the process of component operation, the failure repair mechanism is introduced, and the component may be transferred from the normal state to the failure state, and after repair, it will return to normal condition according to certain repair probability. The transfer of state has randomness and no validity, that is, the future state is only related to the current state, regardless of the past state. The operation of the component can be regarded as a Markov stochastic process of 2 States (normal and failure states), assuming the state is 1, the failure state is 2, and two independent States constitute a simple Markov chain.

Then the member K has the state transition probability matrix: $G_K=\begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix}$.

Where the probability $g_{i,j}$ that the state I is transferred to the state J, $i,j\in\{1,2\}$, the one-step probability transfer matrix of the component state transition can be obtained by using the use case test results.

Assuming the Markov chain $\{V_k^{(n)}, n \in T\}$ of the component K, T is the time set, the initial state transition probability vector is $V_k^{(0)} = [v_1, v_2]$, where in v_1 and v_2 respectively represent the probability of the initial state being normal and abnormal, and the corresponding Markov chain is shown in Figure 1.

Then the state transition probability vector for the current m oment component is: $V_k^{(1)} = V_k^{(0)} G_k$.

The current failure rate of the component k is defined as the 2nd element of probability vector $V_k^{(1)}$, that is: $\lambda_k = v_1 g_{12} + v_2 g_{22}$ .The state transition probability vector of t he component K at n time is: $V_k^{(n)} = V_k^{(0)} G_k^{(n)}$

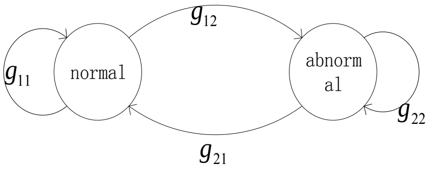

FIGURE I. THE MARKOV CHAIN DIAGRAM OF THE COMPONENT

By the nature of Markov chain, the state transition probability $V_k^{(n)}$ k will be constant convergent, that is, after a long time operation, the member will have a stable state transition probability. The failure rate of the predictable component k in the next n moment is: $\lambda_k^{(n)} = v_1 g_{12}^{(n)} + v_2 g_{22}^{(n)}$.

III. COMPONENT-BASED SOFTWARE RELIABILITY MODEL BASED ON MARKOV PROCESS

In the component reliability modeling, the influence of the component complexity and the importance degree on the reliability of the software system is considered synthetically. According to the Markov chain model to calculate the component failure efficiency, because the module number granularity is too big to objectively reflect the component complexity, the method of statistic instruction code number cannot reflect the structure characteristic of the component software, so the complexity analysis model considering the complexity and the structural of the component with the function body as scalar granularity is proposed. Because artificial assignment avoids the subjective factor of artificial assignment, the importance analysis model combining priority and exponential model is proposed. The traditional model only considers the occupancy rate and the failure efficiency factor, the description is not comprehensive and relatively unitary. In the software system reliability modeling, the software running process can be regarded as a stochastic process of state transition between different components, described as Markov process, the Markov chain of software system is established, and the reliability evaluation model of component level and system level is set up respectively to realize the reliability evaluation and prediction of component software system.

A. Component Reliability Modeling

In the complexity analysis metric, a complexity measurement model is proposed, which takes the function body as the scalar granularity, taking into account the complexity and structure of the component, β_j the complexity of the component j, and the complexity of the component is related to the number of function bodies contained in the component. The definition of complexity is:

$$\beta_j = \alpha_j \Big/ \sum_{i=1}^{n} \alpha_i$$

Where α_j is the number of function bodies that the component j contains.

In the critical degree analysis metric, in order to avoid the reason of artificial influence, the importance of the component should be proportional to the level of the component, that is, the hazard grade, and the important degree is non-linear relationship with the hazard grade, so the significance measurement model of the combination of component level and exponential model is proposed, as described below:

$$\mu_j = \exp\left(rank_j \Big/ \sum_{i=1}^{4} rank_i - 1 \right)$$

μ_j represents the importance coefficient of component j to the whole software system, $\mu_j \in (0,1]$, if the failure of the component will cause the software system to fail, then the μ_j value should be larger, conversely the smaller. The level of importance and degree of harm, the degree of harm level is not limited, the degree of harm from high to low for rank4 ~ rank1, randomly given the descending value (to the importance of measurement has no effect), assuming the hazard level is 3, 2, 1, 0. As shown in Table I. According to the above important degree model, the importance of various kinds of components can be calculated, and the damage degree of the component depends on whether the component is in the key position in the software architecture, and the analysis documents are available by software engineering. The importance of component allocation is shown in Table I.

TABLE I. COMPONENT IMPORTANCE ALLOCATION TABLE

Number	*Harm degree*	*Hazard level*	*Important degree*
rank4	High	3	0.6065
rank3	middle	2	0.5134
rank2	low	1	0.4346
rank1	nothing	0	0.3679

Each member can be given the important index of each component according to the value specified in table 1. The state transition probability between the components is determined by the engineering analysis and the software Operation section. The software system composed of n components has the N state transition probability matrix:

$$p=\begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & \cdots & p_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ p_{n1} & p_{n2} & \cdots & p_{nn} \end{pmatrix}$$

Based on the definitions, the reliability model of software component is established:

$$R(\lambda_j,\beta_j,\gamma_j,\mu_j,t)=\exp\left\{-\lambda_j\beta_j\mu_j\sum_{i=1}^{n}v_i p_{ij}t\right\}=\exp\left\{-\lambda_j\beta_j\mu_j\gamma_j t\right\}$$

Among them, v_i represents the probability of executing component I; p_{ij} represents the probability that the component i executes to the component j; λ_j represents the loss efficiency of component j; β_j represents the complexity of the component j; μ_j represents the importance of component j; γ_j represents the occupancy rate of component K; T represents the widget run time.

Take t equals to 1 to get the reliability evaluation value of the current moment component j:

$$R(\lambda_j,\beta_j,\gamma_j,\mu_j,t)=\exp\left\{-\lambda_j\beta_j\mu_j\gamma_j\right\}$$

. Take t = k to obtain the reliability predictive value of J for future K Cycles:

$$R(\lambda_j,\beta_j,\gamma_j,\mu_j,k)=\exp\left\{-\lambda_j{}^{(k)}\gamma_j{}^{(k)}\beta_j\mu_j t\right\}$$

Among them,

$$\gamma_j{}^{(k)}=\sum_{i=1}^{n}v_i p_{ij}{}^{(k)},\quad \lambda_j{}^{(k)}=v_1 g_{12}{}^{(k)}+v_2 g_{22}{}^{(k)}$$

B. Reliability Modeling of Software System

It is assumed that the software system consists of n functions and relatively independent components of the structure, the running process of the software can be regarded as the process of executing the different components according to a certain probability, and the execution state of the component is relatively independent without any validity, and the running process of the software system can be regarded as an n-state Markov stochastic process. According to the concept of reliability, the reliability of software system is equal to the product reliability of each component, so the reliability model of software system is established:

$$R(\lambda_{1,2,\ldots n},\beta_{1,2,\ldots n},\mu_{1,2,\ldots n},t)=\prod_{j=1}^{n}\exp\left\{-\lambda_j\gamma_j\beta_j\mu_j t\right\}$$

Take T = 1 to obtain the reliability evaluation value of the software system at the current moment:

$$R(\lambda_{1,2,\ldots n},\beta_{1,2,\ldots n},\mu_{1,2,\ldots n},t)=\prod_{j=1}^{n}\exp\left\{-\lambda_j\gamma_j\beta_j\mu_j\right\}$$

, take T = k to obtain the reliability predictive value of the software system after K-cycles in the future:

$$R(\lambda_{1,2,\ldots n},\beta_{1,2,\ldots n},\mu_{1,2,\ldots n},k)=\prod_{j=1}^{n}\exp\left\{-\lambda_j{}^{(k)}\gamma_j{}^{(k)}\beta_j\mu_j\right\}.$$

Among them, $\lambda_j{}^{(k)}$ $\gamma_j{}^{(k)}$ will converge rapidly in several cycles, so software system reliability is only related to time.

IV. APPLICATION RESULTS AND ANALYSIS

A. Component Software System

The above model is applied to a large component software system, which is divided into 5 components according to the function and software structure, the number of function body is obtained from the software manual, the data of the experiment is collected, the analysis component contains the number of function bodies and the damage degree of each component, and the result is shown in Table II.

TABLE II. SOFTWARE STRUCTURE ANALYSIS TABLE

Component number	Function number	Hazard degree
1	10	High
2	21	Middle
3	30	Low
4	18	Middle
5	11	Nothing
Total	90	

The operation profile is a quantitative description of how the software system is used, is the concept of software quality management, the set of operations in the software system and the frequency of their occurrence, the Operation section will provide information about how the user will deploy the built products, it is a numerical description of how the software is used, and can be understood as the use probability of various usage methods. The software system running process is considered as Markov process, and the state transfer between the components is relatively independent. The state transition probability p can be obtained by software engineering analysis and Operation section:

$$p = \begin{bmatrix} 0.20 & 0.15 & 0.15 & 0.30 & 0.20 \\ 0.30 & 0.15 & 0.20 & 0.25 & 0.10 \\ 0.15 & 0.20 & 0.15 & 0.30 & 0.20 \\ 0.20 & 0.05 & 0.25 & 0.40 & 0.10 \\ 0.10 & 0.15 & 0.25 & 0.20 & 0.30 \end{bmatrix}$$

Extract 1000 use cases to test and record software case results,

TABLE III. TEST CASE RECORDS

Component number	Executions number	Failures number	Fixes number
1	118	12	12
2	226	18	16
3	396	28	28
4	172	14	12
5	288	20	20
Total	1200	92	88

B. Calculate the Relevant Indicators

According to the model and component software reliability analysis method, the related occupancy rate, failure rate, importance degree and complexity index are calculated and analyzed, then the software reliability is evaluated.

1) Occupancy

First, calculate the occupancy rate of each component. γ_j represents the proportion of component j executed in a task, the ratio of the number of components executed to the total number of executions in the test phase, which is calculated by Table 3 to calculate the occupancy rate of the components in the system at the current moment, that is $\gamma_j^{(0)} = \left(0.098, 0.188, 0.330, 0.143, 0.240\right)$. By the nature of the Markov chain, the occupancy rate matrix of the components can be predicted in the future N unit time: $\gamma^{(n)} = \gamma^{(0)} P^{(n)} = (0.1852, 0.1294, 0.2043, 0.3065, 0.1736)$.

This can predict the occupancy rate of each component after N unit time in the future.

2) Occupancy rate

Calculating the failure efficiency of each component. Supposing $\lambda = (\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5)$, to set the initial state transfer probability vector $V_k^{(0)}$ of the component K, after several iterations of the stable state transfer probability vector $V_k^{(n)}$, the loss efficiency vectors for evaluating the component failure efficiency at the present moment are: $\lambda^{(0)} = (0.1017, 0.0796, 0.0707, 0.0814, 0.0694)$. The failure efficiency vector of the component failure efficien-cy prediction for n time is:

$$\lambda^{(n)} = (0.1198, 0.0841, 0.0660, 0.0867, 0.0649)$$

3) Complexity and importance

The complexity of software systems can be defined by complexity: $\beta_j = \alpha_j \Big/ \sum_{i=1}^{n} \alpha_i = (0.1111, 0.2333, 0.3333, 0.2000, 0.1222)$.

The importance of the software system can be defined by the importance of:

$$\mu_j = \exp\left(rank_j \Big/ \sum_{i=1}^{4} rank_i - 1 \right) = (0.6065, 0.5134, 0.4346, 0.5134, 0.3679)$$

C. Evaluating Component Software System Reliability

According to the reliability evaluation model of component, we can know the reliability of each component at present time, take t=1, and evaluate it:

$$R(\lambda_j, \beta_j, \mu_j, t) = \exp\left\{ -\lambda_j \beta_j \mu_j \sum_{i=1}^{n} v_i p_{ij} t \right\} = \exp\left\{ -\lambda_j \beta_j \mu_j \gamma_j t \right\}$$

TABLE IV. COMPONENT RELIABILITY ASSESSMENT TABLE

No.	Occupancy	Failure rate	Complexity	importance	Reliability
1	0.1017	0.1017	0.1111	0.6065	0.9987
2	0.0796	0.0796	0.2333	0.5134	0.9988
3	0.0707	0.0707	0.3333	0.4346	0.9979
4	0.0814	0.0814	0.2000	0.5134	0.9974
5	0.0694	0.0694	0.1222	0.3679	0.9995

Take t = 1, the current reliability evaluation value of the software system is:

$$R(\lambda_{1,2,\ldots n}, \beta_{1,2,\ldots n}, \mu_{1,2,\ldots n}, t) = \prod_{j=1}^{n} \exp\left\{ -\lambda_j \gamma_j \beta_j \mu_j t \right\} = 0.9923$$

Compared with the traditional model and the software reliability of the model, the paper analyzes the accuracy and practicability of the model. The traditional model only considers the component occupancy rate, the new model considers the component occupancy, failure rate, complexity and importance factor synthetically. As shown in Fig. 2, fig. 3, the unit of the horizontal axis is a unit time, the reliability of the component under the two models changes over time, and the reliability prediction value of the software system is given, as shown in Figure IV.

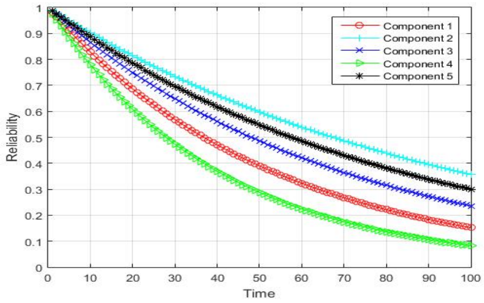

FIGURE II. COMPONENT RELIABILITY PREDICTION TRADITIONAL MODEL

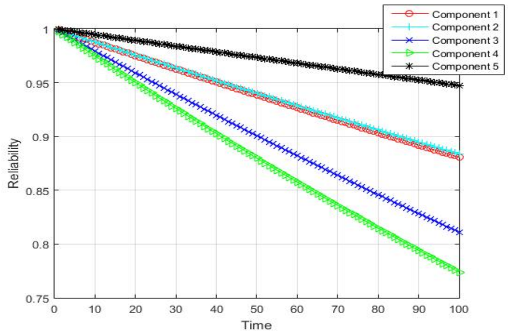

FIGURE III. COMPONENT RELIABILITY PREDICTION THIS PAPER MODEL

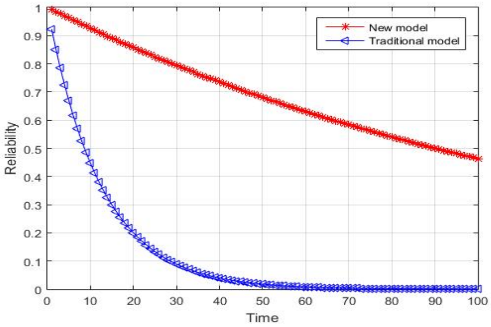

FIGURE IV. RELIABILITY PREDICTION OF SOFTWARE SYSTEM

It can be seen from Fig. II and Fig. IV that in the traditional model, the reliability of the component decreases exponentially with the growth of time, and occupancy rate is the main factor that affects the reliability of the component, the higher the component occupancy rate is, the greater the reliability is affected. Compared with component five, the higher the failure rate, the lower the reliability. From the trend of the general curve, the reliability of software system decreases exponentially with the increase of running time, and the later description is not very specific.

In this model, the reliability of the component is influenced by many factors, and it is shown from fig. III and Fig. IV that the components with higher occupancy rate and greater importance are sensitive to the loss efficiency. Considering the importance factor, the reliability of the component one increases, the contrast component three and component five in the two graphs of the curve, in the loss of efficiency, occupancy rate is not much, the higher the complexity of the lower the reliability, the lower the importance of reliability higher. With the increase of running time, the reliability of the software system decreases slowly, compared with the traditional model, the reliability performance description is more specific, the analysis result is more realistic, and the reliability of the component with higher occupancy rate and important degree is more easily affected, the overall reliability curve trend of software system is more reasonable.

After the reliability analysis of the software, this model can provide some suggestions for the later development and test of the software system, and the component with high occupancy rate will try to improve the reliability index and reduce the failure efficiency. For the failure of the components need to be repaired as soon as possible to avoid long-term operation to reduce the reliability of the software system, minimize the complexity of components; For the component with high importance, it is necessary to reduce the loss efficiency and improve the overall reliability of the software.

V. CONCLUSION

With the development of software, software has been growing rapidly in two aspects of scale and complexity, component-based software design and component reuse have become important methods of software development. In this paper, a new model is proposed based on the traditional component software reliability evaluation model, which not only considers the influence of component occupancy rate factor on component reliability, but also considers the complexity and importance factors of the component. Based on the structure of component software, a complexity measurement model based on the particle size of function body is established, and a combination level and exponential model are established to establish the importance measurement model. On this basis, the process of state transition between components is described as Markov process, and the reliability of software system is evaluated by integrating component-level reliability assessment and system-level reliability activities. This model can be used to analyze and forecast the reliability of the software development lifecycle, which is of great significance to the reliability evaluation of software development and the prediction of software reliability in the later stage.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China under Grants No. 61571346.

REFERENCES

[1] Upadhyay N, Deshpande B M, Agarwal V P. “Reliability modelling and analysis of component-based software system: A graph theoretic systems approach”, International Journal of Reliability and Safety, 2013, 7(2), 174-199.

[2] Xv Xuefeng, Feng Shuli, Shen Shilu. “Research on Component-based software development method and technology [J]”, Journal of Equipment Command and Technical College, 2007(05): 94-98.

[3] Derbel, L. L. Jilani and A. Mili, "Computing attributes of software architectures a static method and its validation", 2015 International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE), Barcelona, 2015, pp. 55-66.

[4] S. S. Gokhale, W. E. Wong, J. R. Horgan and K. S. Trivedi, "An analytical approach to architecture-based software performance and reliability prediction", Performance Evaluation, vol. 2004, 58, no. 4, pp. 391-12.

[5] Zhang Ce, Cui Gang, Liu Hongwei. "Process technology of component software reliability [J]", Journal of Computer Science, 2014, 37(12):2586-2612.

[6] Geng Ji, Nie Peng, Qin Zhiguang. "The present situation and research of software reliability model [J]", Journal of university of electronic science and technology, 2013, 42(4) : 565-570.

[7] C. J. Hsu and C. Y. Huang "An adaptive reliability analysis using path testing for complex component-based software systems", vol. 60, no. 1, pp. 158-170, 2011.

[8] Zhang Wei, Zhang Weiqun. "Research on improved path-based component software reliability model [J]", Computer Science, 2011, 38(02): 148-151.

[9] Liang Shenjie, Zhang Zhihua. "Study on the influence of operation profile on reliability evaluation of software dependability [J]", Computer Engineering and Applications, 2006(14): 102-104+140.

[10] Yue Hai, Zhou Xinlei, Yuan Kun. "Software reliability testing method based on System state analysis [J]", Aerospace control, 2010, 28(05):76-79.

[11] Liu Yanfu. "Research on reliability of software system based on Markov analysis method [D]", Changchun University of Technology, 2005: 96-98.

[12] Du Xianfeng. "Software reliability growth model based on non-homogeneous Poisson process [J]", Journal of Henan University of Technology (natural Science Edition), 2008(03): 328-332.

[13] Yu Huafeng. "Neural network application in the software reliability prediction research [J]", Computer simulation, 2011, 28(04):203-207.

[14] Nie Peng, "A SVM reliability evaluation model for component-based software systems", 2013 2nd International Symposium on Instrumentation and Measurement, Sensor Network and Automation (IMSNA), 2013, pp. 704-708.

[15] Guo Yong, Ma Peijun, Su Xiaohong. "Research status of reliability evaluation technology for component-type systems [J]", Intelligent intelligence and its application, 2013, 3(02):20-24.

[16] W. Wang, D. Pan and M. Chen "Architecture-based software reliability modeling", Journal of Systems and Software, vol. 2016, 79, no. 1, pp. 132-146

[17] Awad Ali, Dayang N.A. Jawawi, Mohd Adham Isa, Ashraf Osman Ibrahim, "Deriving behavioural models of component-based software systems from requirements specifications", Computing Control Networking Electronics and Embedded Systems Engineering (ICCNEEE) 2015 International Conference on, pp. 260-265.

[18] Peng H, Coit D W, Feng Q. "Component reliability criticality or importance measures for systems with degrading components", IEEE Transactions on Reliability, 2016, 61(1): 4-12.