Open Access

Calculation Method of Internetware Reliability Distribution

Jing Zhang[1,2,*], Hang Lei[1], Xuan Han[1] and Yunsheng Wang[1]

[1]School of Computer Science and Engineering, University of Electronic and Science Technology of China, Chengdu, Sichuan, 611731, P.R. China; [2]School of Computer, Panzhihau University, Panzhihua, Sichuan, 617000, P.R. China

Abstract: To plan Internetware reliability in advance can effectively save cost and guarantee the reliability of Internetware system. The conversion method of Markov chain of Internetware based on architecture and the reliability calculation method are studied; the reliability function based on structure is proposed; the characteristics of the improvement of Internetware reliability is analyzed; the cost function of Internetware reliability is studied; the uncertainty of dynamic distribution in a certain range and the distribution accuracy are improved; the pre-distribution is put forward; the effective planned distribution of Internetware reliability is realized by using optimization calculation algorithm. The experiments prove that the proposed method can effectively distribute Internetware reliability with high system reliability, low cost and short distribution calculation time.

Keywords: Algorithm, architecture, cost, distribution, Internetware, reliability, method.

1. INTRODUCTION

Internetware, an important software application mode in the internet environment, puts software entities in the form of software component to Internet node in a way of being open and discretionary. Every software entity can publish in some way in the open environment, and interconnects, intercommunicates, cooperates and forms coalition with other software entities through cross-network in one way or another to form internet application software system with some application functions. Adaptivity, cooperativity, reactivity, evolution and polymorphisms are the main characteristics of Internetware [1, 2].

According to the formation process of Internetware, its reliability is influenced by the plan, design, implementation and operation of the software, and so on. At present the study of software reliability mainly tests the related operation data of the existing software system to calculate system reliability. On the Internet, it is difficult to meet the demand for reaction and adaptation of Internetware and to improve evolution because of the complex testing environment, large amount of data and delay. So, to study reliability plan of the software in the design stage will effectively support the reliability analysis of Internetware system, and prevent the system from perfecting the design and development of the software because its reliability can not meet the demand after the system is made. This can greatly reduce development cost and shorten the cycle. To distribute and plan the reliability in advance is benefit for the effective construction of Internetware system.

There are mature models and methods of Internetware reliability study [3]. Document [4] has studied the test method of Internetware reliability; documents [5, 6] have proposed the calculation method of Internetware reliability based on Markov chain, and realized the reliability calculation based on the state; document [7] has studied the approximate prediction method of Internetware reliability based on structure; document [8] has put forward an approximate method of Internetware reliability and test time distribution; document [9] has proposed the distribution method of test resources in the development stage by using software growth model; document [10] has proposed the strategy model of reliability distribution, and made dynamic plan by using Lagrangian method, but it has not taken the demand for upper and lower limit of reliability in reality into consideration; document [11] has made reliability distribution by using inference of Bayesian network and conditional probability, and calculated reliability distribution through conditional failure probability, but in FTA, it is difficult to decide probability, the importance of structure and so on, and this is influenced by many factors; document [12] has put forward a dynamic approximation algorithm to distribute reliability, and measured the degree of approximation by using the ratio of the cost change to the reliability change, which has some limitations and the single determinant. In a word, because of different reliability calculation methods such as approximation method, or the method based on test data and parameter optimization, the result and the effect of distribution are different.

Internetware reliability calculation has two main methods, one based on path, and the other based on state. The method based on path has no high accuracy. It just makes the estimation of system reliability, and it is not suitable for the infinite path system. So it has its limitations. The method based on state is suitable for the infinite path system. In the open environment, it is loose coupling among components, the components have comparatively high independence, and the transfer among components is in accordance with

Markov characteristics, so it is better to be used in the open environment. But the traditional reliability calculation method based on state has the following shortcomings: 1) the expansion of state space. In the open environment, there may be thousands of components in a system. The traditional analysis method will lead to rapid expansion of state space and increase calculation complexity and even make calculation impossible. 2) Being unsuitable for software with complex program structure. In the open environment, the system structure of software is more and more diversified. The traditional analysis method has some limitations both in application scope and usage convenience. It is only suitable for classic structures, and it can not deal with complex structure systems such as parallel and call [4]. So, based on UML, case diagram and sequence diagram can be used to analyze and simplify structure [13, 14]. Then change it into state diagram which is in accordance with Markov characteristics. At last the system reliability can be calculated in the software design stage and then reliability distribution can be achieved.

This paper has the following parts: part 2 is Internetware reliability calculation method, part 3 is Internetware reliability plan, part 4 is experiment and part 5 are conclusions and future work.

2. INTERNETWARE SYSTEM RELIABILITY

2.1. Reliability Calculation

Internetware is composed of independent software entities, that is, components. Components are combined to form Internetware system with certain architecture. Internetware achieves some specific functions through components executing call (that is, component transfer). Component transfer on certain architecture reflects and embodies Internetware function and reliability. So, under certain architecture, system reliability is related not only to transition probability but also to component reliability.

Definition 1 Calculation model of Internetware reliability

Internetware component set is $R=\{r_1,r_2,...,r_m\}$, $m\in N^+$; transition probability of components i and j is $P=\{p_{ij}\}$, and $i,j\in[1,2,...,m]$; ST is a specific architecture, which is embodied by component transition. Internetware reliability model is $RE: f(ST,P,R)\to[0,1]$.

Transfer control among system components can be described by Markov chain. When executed, each component is corresponding to a Markov chain state. System reliability calculation [4, 5] is as follows:

$$RE=(-1)^{n+1}\frac{|(I-Q)_{n,1}|}{|(I-Q)|}r_n \tag{1}$$

In Formula (1), I is $n\times n$ characteristic matrix; $(I-Q)_{n,1}$ means deleting line n column 1 of matrix $(I-Q)$; $q_{ij}=r_i p_{ij}$; r_n means reliability of No. n component.

Transition probability embodies component connection (transfer direction) and connection measure (transfer ratio), and it also determines reliability calculation result. Software system structure can be obtained through system function design or automatic topology discovery. Transition probability can be obtained through system function design or later test. Usually transition probability numbers are: 1) determined by referring to operation statistics of the existing similar system software; 2) determined by related experts' experience in the industry and field; 3) obtained by test case and software operational profile.

2.2. The Formation of Markov Chain of Internetware

To obtain transition probability, in the early stage of software design, UML sequence diagram is transformed into Markov chain, each component is corresponding to an execution state, and this forms Markov chain state diagram, then transfer relation (or system structure) information and transition probability can be obtained, thus reliability can be calculated. To transform UML sequence diagram into Markov chain to form state diagram of Internetware components and connection, the same boundaries and nodes should be combined, and transition probability should meet the requirement of Markov chain to form normative component transfer diagram and finally obtain Markov chain.

State diagram before transformation $<V,E>$, $V=\{v_1,v_2,...,v_m\}$, $E=\{e\mid e_{ij}=(s_i,s_j,p_{ij})\},i,j\in[1,2,...,m]$, V,E mean the vertex set of components and the boundary set of component connection; p_{ij} means transition probability; the state diagram after transformation is $<V',E'>$.

In UML sequence diagram, the definition of transition probability of component in component transfer p_{ij} is as follows:

$$p_{ij}=\sum_{k=1}^{s}ps_k\times\left[\frac{|\operatorname{int}eract(c_i,c_j)|}{|\operatorname{int}eract(c_i,c_l)|_{l=1,2,...,N}}\right]_{c_i,c_j\in S_k} \tag{2}$$

S: the number of scenes; ps_k: execution probability of scene s_k;

N: the number of components;

$|\operatorname{int}eract(c_i,c_j)|$: the interaction number of components i and j in scene s_k.

The calculation of converting UML sequence diagram to Markov chain is as follows:

1). Initialization. Set initial value V,E, V',E' are empty.

If $v_i,v_j\in V$, $i,j\in[1,2,...,m]$, then $v_i\in V'$

2). $\exists$ e_{jf}, $f,j\in[1,2,...,m]$, if $v_j=v_f$, then delete any node of v_j,v_f. Suppose deleting v_j, then $\{e_{ik}\}$, $\{v_i\}$, $i,k\in[1,2,...,j-1,f,..,m]$.

3). If $\exists$ $e_{it},e_{jt}\in E$, $v_j,v_i\in V$, $v_j=v_i$,

If $e_{jt}\in E'$, then $p'_{jt}=p_{it}+p'_{jt}$;

If $\exists$ $e_{jt}\notin E'$, then $e_{jt}\in E'$, $e_{jt}=(v_j,v_t,p_{jt})$.

4). $\exists\ v_i \in V'$, start from v_i and form set of all boundaries E''

5). In E'', $e_{ij} = (v_i, v_j, p''_{ij})$, update to $p''_{ij} = p'_{ij} / \sum p'_{ij}$

6). In E'', $q_{ij} = p''_{ij} r_i$;

7). calculate reliability according to formula (1).

2.3. Reliability Calculation Function

Suppose the Internetware system is composed of n components, start component is s_1, reliability is r_1, and end component is s_t.

As in Fig. (**1**), suppose transfer state matrix P. The transfer from s_1 to s_t may go directly, or go from branch nodes, thus there would be K steps to achieve the goal. There are the following situations:

1). directly: $s_1 \rightarrow s_t$

2). go through transfer nodes: $s_1 \rightarrow s_i \rightarrow \ldots \rightarrow s_t$, or $s_1 \rightarrow s_{i+1} \rightarrow \ldots \rightarrow s_t$, or $s_1 \rightarrow s_n \rightarrow \ldots \rightarrow s_t$,

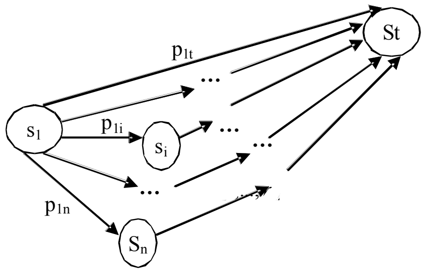

Fig. (1). State diagram.

The system reliability is:

$$R = R_{1t} + R_{12,t} + \ldots + R_{1n,t}$$

$$= R_{1t} + \sum_{i=2}^{n} R_{1i,t} = R_{1t} + \sum_{i=2}^{n} (r_1 \times p_{1i} \times r_i \times p_{it})$$

That is:

$$R = R_{1t} + \sum_{i=2}^{n} (r_1 \times p_{1i} \times r_i \times p_{it}) \qquad (3)$$

If there is return in s_i, as in Fig. (**2**), then obtain:

$R_{1i,t} = r_1 p_{1i} \sum_{k=0}^{\infty} (r_i p_{ii})^k r_i p_{it} = r_1 p_{1i} (I - Q)_{ii}{}^{-1} r_i p_{it}$, in which $Q_{ii} = r_i p_{ii}$.

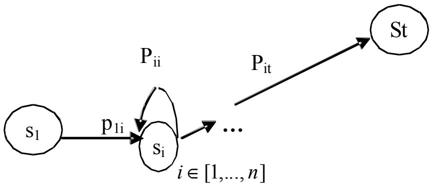

Fig. (2). State diagram in return.

$$R = R_{1t} + R_{12,t} + \ldots + R_{1n,t} = R_{1t} + \sum_{i=2}^{n} (r_1 p_{1i} (I - Q)_{ii}{}^{-1} r_i p_{it}),$$

in which I is unit matrix, matrix $Q = \{q_{ij}\}$, $q_{ij = r_i \times p_{ij}}$, that is:

$$R = R_{1t} + \sum_{i=2}^{n} (r_1 p_{1i} (I - Q)_{ii}{}^{-1} r_i p_{it}) \qquad (4)$$

So reliability can be calculated by using formula (4).

If there are the situations as in Figs. (**1**) and (**2**), see Fig. (**3**).

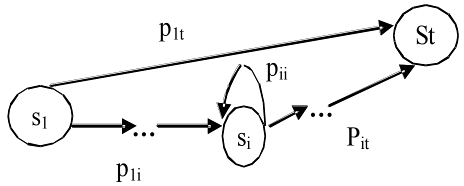

Fig. (3). State diagram under comprehensive situations.

$R = r_1 p_{1t} + r_1 p_{1i} (I - Q)_{ii}{}^{-1} r_i p_{it}$, obtain:

$$R = r_1 p_{1t} + r_1 p_{1i} r_i p_{it} / (1 - r_i p_{ii}) \qquad (5)$$

For convenience, abbreviated to $A_{1t} = r_1 p_{1t}$, $B_{1i} = r_1 p_{1i}$, $C_{1t} = r_i p_{it}$, $D_{ii} = r_i p_{ii}$, then obtain:

$$R = A_{1t} + B_{1i} C_{1t} / (1 - D_{ii}) \qquad (6)$$

C_{1t} is the reliability from s_1 to s_t, and D_{ii} is the reliability from s_i to s_i.

So, the degree of components influencing system reliability, that is, the sensitivity, is:

$$\partial R / \partial r_i = r_1 p_{ii} p_{it} / (1 - r_i p_{ii})^2 \qquad (7)$$

According to formula (7), the importance of components reliability in Internetware system reliability can be evaluated.

Because $p_{1i} + p_{1t} = 1$, $p_{ii} + p_{it} = 1$, then:

$$\partial R / \partial r_i = r_1 p_{ii} (1 - p_{ii}) / (1 - r_i p_{ii})^2$$

Change formula (5) to obtain:

$$r_i = (R - r_1 p_{1t}) / (p_{1i} p_{it} r_1 - p_{1t} p_{ii} r_1 + R p_{ii}) \qquad (8)$$

Obtain pre-distribution reliability of Internetware component:

$$r_i = (R - r_1 p_{1t}) / ((1 - p_{1t}) p_{it} r_1 + (R - r_1 p_{1t})(1 - p_{it})) \qquad (9)$$

In order to calculate easily, suppose r1=1, that is, the start component is reliable which can operate successfully and guarantee the transfer of component execution state, thus formula (9) becomes:

$$r_i = (R - p_{1t}) / ((1 - p_{1t}) p_{it} + (R - p_{1t})(1 - p_{it})) \qquad (10)$$

$$p_{ii} = 1 - p_{it} = 1 / (1 / (1 - p_{it})) = 1 / \sum_{k=0}^{\infty} (p_{it})^k = 1 / (I - P)_{ii}{}^{-1}$$

That is:

$$p_{ii}=1/(I-P)_{ii}{}^{-1} \tag{11}$$

$$p_{1t}=1-p_{1i}=1-p_{1i}/((1-p_{ii})/(1-p_{ii}))$$

$$=1-p_{1i}\sum_{k=0}^{\infty}(p_{ii})^k/\sum_{k=0}^{\infty}(p_{ii})^k \ =1-\sum_{k=0}^{\infty}(p_{1i}p_{ii})^k/\sum_{k=0}^{\infty}(p_{ii})^k$$

That is:

$$p_{1t}=1-\ (I-P)_{1i}{}^{-1}/(I-P)_{ii}{}^{-1} \tag{12}$$

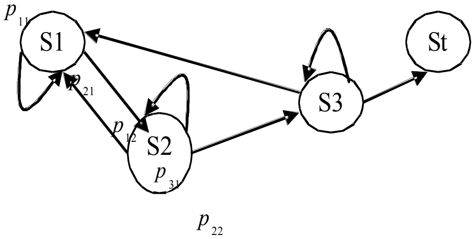

Fig. (4). System state diagram.

As in Fig. (**4**), components S1, S2 and St form an Internetware system. Suppose component reliabilities are respectively r1=0.997559, r2=0.999939**,** r3=0.999989, and transition probabilities P_{11}=0.8, p_{12}=0.2, p_{21}=0.4, p_{22}=0.4, p_{23}=0.2, p_{31}=0.4, p_{33}=0.4, p_{3t}=0.2

Transition probability matrix is:

Q=[0.8*r1 0.2*r1 0; 0.4*r2 0.4*r2 0.2*r2; 0.4*r3 0 0.4*r3].

According to documents [4, 5] and formula (1), system reliability expression is:

R=0.2*(-r1*r2)/(25*((4*r1)/5 + (2*r2)/5 + (2*r3)/5 - (6*r1*r2)/25 - (8*r1*r3)/25 - (4*r2*r3)/25 + (14*r1*r2*r3)/ 125 - 1)) (13)

Put each component reliability value into the formula (13), obtain system reliability RE1 = 0.900021269205992.

According to Fig. (**4**) and formula (6), i=1,r_1p_{1t}=0,B_{1i}=1, suppose C_{1t}= $C_{1\text{-}t}$ is the reliability from s_1 to s_t, that is, $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow s_t$; suppose D_{ii}= $D_{1\text{-}1}$ is the reliability from state 1 to state 1, that is, $s_1 \rightarrow s_1$, $s_1 \rightarrow s_2 \rightarrow s_1$, $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow s_1$, thus obtain:

$C_{1\text{-}t}$= r1*0.2*r2*0.2*r3*0.2/((1-r3*0.4)*(1-r2*0.4));

$D_{1\text{-}1}$=r1*0.8+r1*0.2*r2*0.4/(1-r2*0.4)+r1*0.2*r2*0.2*r3*0.4/((1-r2*0.4)*(1-r3*0.4));

According to formula (6), obtain system reliability RE= C_{1t} /(1- D_{ii}), that is:

R=-(r1*r2*r3)/(125*((2*r2)/5 - 1)*((2*r3)/5 - 1)*((4*r1)/ 5 - (2*r1*r2)/(25*((2*r2)/5 - 1)) + (2*r1*r2*r3)/(125*((2* r2)/5 - 1)*((2*r3)/5 - 1)) - 1)) (14)

Put each component reliability value into the formula (14), obtain RE2= 0.900011368972042. Compare the calculation results, absolute error is 9.900233949977633e-06, and relative error is 1.099999998745776e-05.

Using Monte Carlo sampling method, the mean value of the formula (13) is 0.900021275210536, and the variance is 2.399116264688769e-08. The mean value of the formula (14) is 0.899748034921805, and the variance is approximately zero.

So, the reliability function is effective.

3. INTERNETWARE COMPONENT RELIABILITY DISTRIBUTION

3.1. Reliability Cost Function

Software reliability is directly related to development cost which is influenced by many factors such as software complexity, development tool, development experience and developers. Reliability cost function has the following significant characteristics:

1). The value of the function is positive;

2). The function is non-decreasing;

3). The higher the reliability, the higher the reliability cost.

The present reliability cost functions are mainly powerful number model, polynomial model, Lagrangian model, logarithm model, and index model. According to classic Mettas cost function, and because Internetware should connect and call on cross-network to achieve interaction among components, set a interaction ratio parameter α_i. Cost function is as follows:

$$C(R_i)=e^{\alpha_i(1-c_i)\frac{R_i-R_{i,\min}}{R_{i,\max}-R_i}} \tag{15}$$

In which c_i is the complexity of component i, R_i is the reliability of component i, $R_{i,\min}$ is the minimum reliability in all components, and $R_{i,\max}$ is the maximum reliability in all components. Complexity reference c_i can be tested by using cyclomatic complexity, but this will not be discussed in this paper.

In initial stage, because of developers' degree of familiarity with software and the preparation, earlier stage cost is very high, Use LOG function.

That is:

$$f(r_i)=LOG(\sum C(r_i)) \tag{16}$$

3.2. Reliability Distribution

Reliability distribution demands that system reliability be the highest and that cost be the lowest [15, 16]. According to component reliability and cost calculation formula, obtain the following system reliability distribution plan model:

$$Min \qquad f(r_i)$$

$$s.t. \qquad R(r_i) >= R_0\ ,\ 1 > r_i > 0 \tag{17}$$

The above model can be calculated by using optimnization algorithm [17, 18] to obtain component reliability distribution. In order to improve efficiency, use formulas (10) - (12) to calculate and obtain component pre-distribution value and use it as reference value to obtain calculation result faster and more accurately.

Table 1. Reliability Distribution.

No.	*Pre-distribution Value: (r1,r2,r3)*	*Module Distribution Reliability*	*System Reliability*	*Cost*	*Operation Time (Second)*
1	*(0.5 0.5 0.5)*	*(0.998005536576151 0.998960330340881 0.998960867524147)*	*0.900023744807483*	*1.018754019698208e+02*	*8.289827999999998*
2	*(1 1 1)*	*(0.998015359044075 0.998929411172867 0.998965859413147)*	*0.900024559364200*	*1.023813850544096e+02*	*16.600520285714286*
3	*(0.978260869565220 0.925925925925926 0.892857142857143)*	*(0.997991818532200 0.998987740839202 0.999005415400648)*	*0.900044805839254*	*1.011830181935573e+02*	*7.674584285714286*

4. ILLUSTRATION

According to Fig. (**4**), obtain reliability distribution model:

$$Min \quad \log\left(\sum_{i=1}^{n} \alpha_i \times e^{c_i \frac{R_i}{1-R_i}}\right)$$

$$s.t. \quad R(r_i) >= R_0 \,, 1 > r_i > 0.5 \qquad (18)$$

In which calculate R(ri) by using formula (14), and set R_0=0.9 to mean the lowest value required by system reliability.

Use genetic algorithm, set some parameters: population: 200, the biggest iterative number: 500, crossover probability :0.97, mutational rate: 0.01, and mobility: 0.2.

Use formulas (10)-(12) to obtain respectively reliability pre-distribution values of components is:

[r1, r2, r3]=[0.978260869565220, 0.925925925925926, 0.892857142857143].

Use Matlab simulation software, reliability distribution and corresponding system reliability, and cost are in Table **1**.

In Table **1**, for the scheme designed in this paper (that is, scheme 3), the cost is decreased by 0.68%, and 1.17%, compared with scheme 1 and scheme 2; the system reliability is the highest; the operation time is the shortest, which has been shortened by 7.42% and 53.7%. So it achieves the distribution purpose and effect.

The importance of reliability is analyzed by using formula (7).

$$(\partial R / \partial r_1, \partial R / \partial r_2, \partial R / \partial r_3) = \ (36.7231, 12.3188, 4.2017).$$

The influence on system reliability are respectively 68.97%, 23.137%, 7.893%. So system reliability is mainly influenced by component 1 and component 2.

5. CONCLUSIONS AND FUTURE WORK

Internetware reliability plan distribution involves many factors. Doing reliability plan distribution in design stage is benefit to system development and construction, decreases later modification cost and guarantees Internetware system reliability. The method to change Internetware into Markov state chain according to architecture in design stage has been studied; the reliability computation and the reliability function generation method have been analyzed. Combined with Internetware characteristics, the cost function of improving reliability has been studied. According to pre-distribution, using genetic algorithm to make dynamic plan, reliability has been distributed effectively with low cost, short distribution time and improved system reliability. Later automatic extraction method of Internetware architecture and automatic acquisition method of transition probability will be studied to improve the generation efficiency of reliability function. The accuracy of reliability cost function will be studied to improve the accuracy and comprehensive effect of Internetware reliability distribution.

CONFLICT OF INTEREST

The authors confirm that this article content has no conflict of interest.

ACKNOWLEDGEMENTS

This work is sponsored by the Foundation Project (613153) of natural science of Hainan province, and the Foundation Project(14ZA0341) of Sichuan Education Department. Authors gratefully thank the anonymous reviewers for their valuable comments on this manuscript.

ABOUT THE AUTHORS

First Author ZHANG Jing is currently a doctor candidate in School of Computer Science and Engineering, UESTC and a professor at the Computer School, Panzhihua University. His research interests include software reliability and computer network reliability

Second Author LEI Hang, received the Ph.D. degree from UESTC in 1997. He is currently a professor with UESTC. His research interests include software reliability testing and evaluation, hardware-software co-design of embedded systems.

Third Author HAN Xuan is currently a doctor candidate in School of Computer Science and Engineering, UESTC. His research interests include software engineering, software reliability analysis techniques and software test case generation.

Fourth Author WANG Yun-sheng is a Ph.D. Candidate in School of Computer Science and Engineering of UESTC and the programs manager in China Electronic Technology Avionics Co., LTD. His research interests include software engineering and avionics software reliability.

REFERENCES

[1] H. Mei, G. Huang, and T., "Internetware: A software paradigm for Internet computing", *IEEE Softw.,* vol. 45, no. 6, pp.26-31, 2012.

[2] W. T. Tsai, Z. Jin, and X. Y. Bai, "Inrnetware Computing:Issues and Perspective", *Int. J. Softw. Inform.*, vol. 3, no. 4, pp. 415-438, 2009.

[3] K. Tyagi, A. Sharma, "Reliability of component based systems - A critical survey", *WSEAS Trans. Comput.*, vol. 2, no. 11, pp. 45-54, 2012.

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

[5] R. C. Cheung, "A User-Oriented software reliability model", *IEEE Trans. Softw. Eng.*, vol. 6, no. 2, pp. 118 - 125, 1980.

[6] J. Lo, C. Y. Huang, I. Y. Chen, S. Y. Kuo, and M. R. Lyu, "Reliability assessment and sensitivity analysis of software reliability growth modeling based on software module structure", *Comput. Softw. Appl.*, vol. 76, no. 1, pp. 3-13, 2005.

[7] S. S. Gokhale, K. S. Trivedi, *"Reliability Prediction and Sensitivity Analysis Based on Software Architecture*", in Proceedings of the 13th International Symposium on Software Reliability Engineering(ISSRE'02), Nov 12-15, 2002,Annapolis, MD, USA, pp.64-78.

[8] R. Pietrantuono, S. Russo, K. S. Trivedi, "Software Reliability and Testing Time Allocation: An Architecture-Based Approach", *IEEE Trans. Softw. Eng.,* vol. 36, no. 3, pp. 323-336, 2010.

[9] M.R. Lyu, S.Rangarajan, A.P. A. van Moorsel, "Optimal allocation of test resources for software reliability growth modeling in software development", *IEEE Trans. Reliab.*, vol. 51, no. 2, pp. 183-192, 2002.

[10] J. H. Lo, S. Y. Kuo, M. R. Lyu, and C.Y. Huang, "*Optimal Resource Allocation and Reliability Analysis for Component-Based Software Applications*", in Proceedings of the 26 th Annual International Computer Software and Applications Conference (COMPSAC'02), 26-29 August, 2002, Oxford, England, pp.7-12.

[11] W. X. Qian, X. W. Yin and L. Y. Xie, "System Reliability Allocation Based on Bayesian Network", *Appl. Math. Inform. Sci.*, vol.6, no. 3, pp. 681-687, 2012.

[12] H. Guan, T. M. Wang, and W. R. Chen. "Exploring Architecture-Based Software Reliability Allocation Using a Dynamic Programming Algorithm", in Proceedings of the Second Symposium International Computer Science and Computational Technology (ISCSCT'09), Huangshan, P.R. China, 26-28, Dec. 2009, pp. 106-109.

[13] J. Yan, J. Wang, H. W. Chen, "Deriving Software Markov Chain Usage Model from UML Models", *J. Softw.,* vol. 16, no. 8, pp. 1386-1394, 2005.

[14] L. Y. Yin, "Study on software reliability analysis model automatically transform based on UML", M.S Thesis, Chongqing University, April 2012.

[15] K. K. Aggarwal, J. S. Gupta, "On minimizing the cost of reliable systems," *IEEE Trans. Reliab.*, vol. 24, no. 3, pp. 205-209, 1975.

[16] A. O. C. Elegbede, C. Chu, K. H. Adjallah, and F. Yalaoui, "Reliability allocation through cost minimization", *IEEE Trans. Reliabil.,* vol. 52, no. 1, pp. 106-111, 2003.

[17] H. Rathod, M. Parmar, "Study of genetic approach in estimating reliability of component based software", *Indian J. Res.*, vol. 1, no. 11, pp. 17-19, 2012.

[18] S. H. Aljahdali, M. E. El-Telbany, "Genetic algorithms for optimizing ensemble of models in software reliability prediction", *ICGST-AIML J.,* vol. 8, no. 1, pp. 5-13, 2008.

Received: September 22, 2014 Revised: November 30, 2014 Accepted: December 02, 2014