

doi:10.21311/002.31.7.14

Quality Evaluation Model and Algorithm of Software System Based on Fuzzy Closeness Degree

Xinglong Liu, Aixin Wang, Lei Wang

Institute of information science and technology, Agricultural University of Hebei

Abstract

The development of software system is a complex system engineering work. The quality of a software system is controlled by multiple factors such as design requirements, industry application background, software system performance, the technological level of software development, markets, and the society. Therefore, we targeted at software system quality evaluation in this paper, and established a fuzzy closeness degree based model and algorithm to evaluate the quality of software system. Specifically speaking, we first extended the conventional ISO / IEC9126 quality model based software system quality evaluation index system into such a version that better applied itself to actual situations of enterprise software system development quality evaluation. Then, by using AHP, we calculated the weights of all evaluation indices involved here. After normalization treatment, we afforded the fuzzy distances and fuzzy closeness degree model, on whose basis the quality of software system was assessed. According to the results of empirical verification, the proposed algorithm and model in this paper are effective and feasible.

Key words: software system; software quality; evaluation index; evaluation model; fuzzy closeness degree

1. INTRODUCTION

The flourishing information technology and computer technology accelerate the rapid development of the software engineering technology. Software systems of various forms and functions are widely and irreplaceably used by growing enterprises and units in such fields as design / manufacturing, service industry, medical industry, education industry, e-commerce and engineering construction (Zhou et al., 2015; Sun et al., 2011). The increasingly enlarged size and more important functions of software systems highlight quality-related problems. Some problems can be serious enough to cause immeasurable and irreversible losses. In light of it, more and more attention has been drawn to software system quality studies (Shi et al., 2014; Gu and Chen, 2014; Gong, 2013). The development of software system is subject to demands of growth of different fields and industries. The quality of a software system is comprehensively controlled by multiple factors such as design requirements, industry application background, software system performance, the technological level of software development, markets, and the society. It is possible that quality problems happen at any of the requirement phase, the design phase and the maintenance phase. In this sense, software quality assessment is a complicated system process involving multi-level, multi-index and multi-stage decision-making analysis (Chen et al., 2013; Shen et al., 2016). At present, numerous studies have been undertaken on this, producing a reasonable number of research achievements as great contributors to the improvement of software system quality (Li et al., 2014; Chai et al., 2012; Su and Li, 2015; Yue, 2013). However, in consideration of the fuzzy uncertainties and application-varying characteristics of factors influencing software systems during their development and operation, the evaluation output of software system quality will be more

reliable with an effective processing of fuzzy uncertainties of evaluation indices. To this end, our analysis of software system quality evaluation was conducted on the foundation of the fuzzy system theory (Kaveh et al., 2013; Maisa et al., 2016), with the provision of a fuzzy closeness degree based software system quality evaluation model and the corresponding algorithm.

2. A MULTI-LEVEL EVALUATION INDEX SYSTEM OF SOFTWARE SYSTEM QUALITY

2.1 Index selection and system design principles

(1) Scientific principle. Indices should be selected in a way that reflects the characteristics of software system quality evaluation rationally and scientifically. They are required of the possession of explicit concepts and the ability to represent the clear essence of quality problems.

(2) Systematic principle. Indicators selected are expected to assess the quality of software system from the perspective of integrity and comprehensiveness and by means of extracting dominant influencing factors. This principle serves for the guarantee of the consistency and accuracy of evaluation output.

(3) The principle of measurability. Be it qualitative or quantitative, qualified indices should have good operability in terms of efficacious obtainment of measure values, which facilitates effective sequencing of evaluation results.

(4) The principle of practicality. Indicators should be selected in a way that conform to the specific implementation situations of software systems. It is not recommendable to select too many or too few indicators because evaluation output tends to be distorted or lack reliability in either cases.

(5) The principle of objectivity. Indices should not be selected subjectively, but according to the objective realities of software system development and its service. Moreover, qualified indices should be stable to a certain extent.

2.2 Contents and connotations of evaluation indices

Under the guidance of the above principles, we selected evaluation indices in two aspects: software system development and software system usage.

The evaluation from the perspective of software system development was done mainly by using the classic ISO/IEC 9126 quality model. The specific indicators and their connotations are listed in Table 1.

Table 1. Indices of the ISO / IEC 9126 quality model

Evaluation perspective	Index	Connotation
software system development U_1	functionality u_{11}	The ability to satisfy functional requirements, ranging from accuracy, adaptability, operability and security to functional compliance.
	reliability u_{12}	The ability to maintain performance, including maturity, fault tolerance, recoverability and reliable compliance.
	usability u_{13}	The ability for the software to be easily used, which covers comprehensibility, learnability, operability, attractiveness, and usable compliance.

	efficiency u_{14}	Referring to the applicability of the software in terms of temporal characteristics, resource utilization and efficiency compliance
	serviceability u_{15}	Referring to the writability of the software with respect to stability, writability, testability, analyticity and maintainable compliance
	transportability u_{16}	The ability for the software to be transplanted, including adaptability, instability, replacement, coexistence, transplantable compliance

The evaluation from the perspective of software system usage was done mainly based on the background of the software application field. The specific indicators and their connotations are listed in Table 2.

Table 2. Evaluation indices involving software system usage

Evaluation perspective	Index	connotation
Software system usage U_2	Development cost u_{21}	Costs generated in the design phase
	Degree of satisfaction u_{22}	The overall user evaluation of the software system
	Development period u_{23}	The development time needed from demand analysis to complete application
	accuracy u_{24}	The degree to which the design results satisfy actual requirements
	Risk controllability u_{25}	The ability to defend against the attack of viruses, Trojans and plug-ins

2.3 Establishment of multi-level index system

According to the above discussions, we established a multi-level index system to evaluate software system quality, as shown in Figure 1.

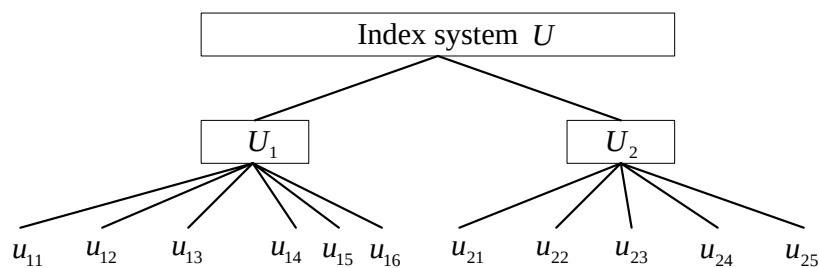


Figure 1. The multi-level quality assessment index system

3. ESTABLISHMENT OF THE FUZZY CLOSENESS DEGREE BASED SOFTWARE SYSTEM QUALITY EVALUATION MODEL

3.1 Determination of the index set

The index set required for software system quality evaluation is divided into two levels, the tier-one index set is

$$U = \{U_1, U_2\} \quad (1)$$

The tier-two index set is

$$\begin{cases} U_1 = \{u_{11}, u_{12}, u_{13}, u_{14}, u_{15}, u_{16}\} \\ U_2 = \{u_{21}, u_{22}, u_{23}, u_{24}, u_{25}\} \end{cases} \quad (2)$$

3.2 Determination of the index weight

The AHP method (Mohammad, et al., 2015) is characterized by simplicity, practicability and reliable calculation results, which satisfies our requirements and is accordingly used in this paper to determine index weight. Assuming that there were n indices to be determined, we obtained the AHP-based comparison matrix A by pairwise comparison.

$$A = [a_{ij}]_{n \times n} \quad (3)$$

Where a_{ij} was the comparison magnitude between index i and index j .

The weight of the index i was

$$w_i = \frac{\sum_{j=1}^n a_{ij}}{\sum_{i=1}^n \sum_{j=1}^n a_{ij}} \quad (4)$$

The weight sequence W was obtained as

$$W = (w_1, w_2, \dots, w_n) \quad (5)$$

In order to determine the validity of index weight calculated before, consistency check became a necessity, i.e.

$$AW = \lambda_{\max} * W \quad (6)$$

$$CI = \frac{\lambda_{\max} - n}{n - 1} \quad (7)$$

$$CR = \frac{CI}{RI} \quad (8)$$

Where the value of RI was obtainable by look-up table. If $CR < 0.1$, the comparison matrix A had good consistency; otherwise A needed modifications to meet requirements.

3.3 Index normalization

According to the description of the said index system, there are two types of indices, the benefit index and the cost index. Therefore, normalization processing is necessary. We assume that there are m software system schemes and that the j -related magnitude of scheme i is v_{ij} . If j is in essence a benefit index, the normalized value r_{ij} will be

$$r_{ij} = \frac{v_{ij} - \min_{1 \leq i \leq m} (v_{ij})}{\max_{1 \leq i \leq m} (v_{ij}) - \min_{1 \leq i \leq m} (v_{ij})} \quad (9)$$

If j is a cost index, the normalized value r_{ij} will be

$$r_{ij} = \frac{\max_{1 \leq i \leq m} (v_{ij}) - v_{ij}}{\max_{1 \leq i \leq m} (v_{ij}) - \min_{1 \leq i \leq m} (v_{ij})} \quad (10)$$

The normalized r_{ij} falls into the interval of $[0, 1]$. All the indices are subject to a uniform evaluation standard, thus ensuring the credibility of the assessment results.

3.4 Fuzzy distance and fuzzy closeness degree

In this paper, the Euclidean distance was used as the calculation model of the distance between evaluation schemes and the optimal scheme. In consideration of the generality and the j -related magnitude of scheme I $r_{ij} = [r_{ij}^a, r_{ij}^b]$, the fuzzy distance D_{ij} between the scheme I and the optimal scheme is

$$D_{ij} = \sqrt{\frac{|\max_{1 \leq i \leq m} (r_{ij}) - r_{ij}^a|^2 + |\max_{1 \leq i \leq m} (r_{ij}) - r_{ij}^b|^2}{2}} \quad (11)$$

The corresponding fuzzy closeness degree ψ_{ij} is

$$\psi_{ij} = 1 - D_{ij} \quad (12)$$

3.5 Comprehensive evaluation model

If there are m quality evaluation schemes, the scheme set C is expressed as

$$C = \{C_1, C_2, \dots, C_m\} \quad (13)$$

As said before, there are two tiers of index set required for software system quality evaluation. If the weight of a tier-one index j is w_j^1 , then the comprehensive fuzzy closeness degree ψ_{ij}^2 is

$$\psi_{ij}^1 = \sum_{j=1}^{n_1} (w_j^1 * \psi_{ij}) = \sum_{j=1}^{n_1} (w_j^1 * (1 - D_{ij})) \quad (14)$$

If the weight of a tier-two index j is w_j^2 , the comprehensive fuzzy closeness degree ψ_i^2 is

$$\psi_i^2 = \sum_{j=1}^{n_2} (w_j^2 * \psi_{ij}^1) \quad (15)$$

3.6 Realization of the model and the algorithm

The optimal software system is calculated according to the value of ψ_i^2 . The optimal scheme k should satisfy

$$\psi_k^2 = \max(\psi_1^2, \psi_2^2, \dots, \psi_m^2) \quad (16)$$

As a summary, we listed the steps of realizing fuzzy closeness degree based model and algorithm below.

Step 1: Based on the principle of index selection, the software system quality evaluation is selected;

Step 2: Establish a software system quality assessment system to generate an evaluation index set;

Step 3: Appropriate weight assignment method is used to determine the weight of evaluation indices;

Step 4: Obtain evaluation schemes and form the evaluation scheme set.

Step 5: Obtain the magnitude of evaluation indices, and normalize evaluation indices of different types;

Step 6: Obtain the fuzzy distance and fuzzy closeness degree of the evaluated schemes;

Step 7: Obtain the optimal evaluation scheme based on fuzzy closeness degree.

4. CASE STUDY

The model and algorithm in this paper were validated and further explained with application to the evaluation of the service software system quality of a local medical and health system. Four high-end medical units participated in the evaluation, whose magnitudes of different evaluation indices were acquired and listed in Table 3 through survey and statistic work.

Table 3. Magnitudes of evaluation indices

Index	weight	Evaluation target			
		A	B	C	D
functionality u_{11}	0.259	0.90	0.95	0.90	0.95
reliability u_{12}	0.238	0.85-0.90	0.90-0.95	0.95-1.00	0.95-1.00
usability u_{13}	0.123	0.90-0.95	0.80-0.85	0.85-0.90	0.90-0.95
efficiency u_{14}	0.091	0.80	0.90	0.90	0.85
serviceability u_{15}	0.223	0.85	0.90	0.95	0.85
transportability u_{16}	0.066	0.85-0.90	0.95-1.00	0.85-0.90	0.95-1.00
Development cost u_{21}	0.054	100	120	120	150
Degree of satisfaction u_{22}	0.272	0.85	0.95	0.90	0.90
Development period u_{23}	0.171	180	150	100	90
accuracy u_{24}	0.407	0.80	0.90	0.90	0.95
Risk controllability u_{25}	0.095	0.80-0.85	0.90-0.95	0.95-1.00	0.80-0.85

The comparison matrices of indices under the respective principle UI and principle U2 and corresponding index weights were acquired and listed in Table 3.

$$A_1 = \begin{bmatrix} 1 & 1 & 2 & 3 & 1 & 5 \\ 1 & 1 & 3 & 2 & 1 & 3 \\ 1/2 & 1/3 & 1 & 2 & 1/2 & 2 \\ 1/3 & 1/2 & 1/2 & 1 & 1/3 & 2 \\ 1 & 1 & 2 & 3 & 1 & 2 \\ 1/5 & 1/3 & 1/2 & 1/2 & 1/2 & 1 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 & 1/5 & 1/3 & 1/7 & 1/2 \\ 5 & 1 & 2 & 1/2 & 3 \\ 3 & 1/2 & 1 & 1/2 & 2 \\ 7 & 2 & 2 & 1 & 4 \\ 2 & 1/3 & 1/2 & 1/4 & 1 \end{bmatrix},$$

We normalized evaluation indices and obtained their fuzzy closeness degrees, as shown in Table 4.

Table 4. The fuzzy closeness degree of evaluation indices

Index	Evaluation target			
	A	B	C	D
functionality u_{11}	0.950	1.000	0.950	1.000
reliability u_{12}	0.873	0.921	0.965	0.965
usability u_{13}	0.965	0.823	0.921	0.965
efficiency u_{14}	0.900	1.000	1.000	0.950
serviceability u_{15}	0.900	0.950	1.000	0.900
transportability u_{16}	0.873	0.965	0.873	0.965
Development cost u_{21}	1.000	0.833	0.833	0.667
Degree of satisfaction u_{22}	0.900	1.000	0.950	0.950
Development period u_{24}	0.500	0.600	0.900	1.000
accuracy u_{24}	0.900	0.950	0.950	1.000
Risk controllability u_{25}	0.823	0.921	0.965	0.823

Considering the weight of the criterion layer $w_{u1}=0.60$, $w_{u2}=0.40$, according to the aforementioned computation model, we obtained the comprehensive fuzzy closeness degrees $\psi_A^2=0.879$, $\psi_B^2=0.925$, $\psi_C^2=0.951$, $\psi_D^2=0.955$, Based on their values, the software systems used by unit C and unit D were proved to be the best ones that benefits the work to be done.

5. CONCLUSION

We extend the conventional ISO/IEC9126 quality model based software system quality evaluation index system into an adaptive version. Through normalization treatment, we calculate the weights of all evaluation indices involved here. Then, we provide the fuzzy distances and fuzzy closeness degree model, on whose basis the quality of software system was assessed. This novel method and algorithm feature simplicity, practicability and good reliability in evaluating complex software systems. According to the results of empirical verification, the proposed algorithm and model in this paper are effective and feasible

6. REFERENCES

- Chai H., Yan J., Li Q. (2012). Research on fuzzy evaluation model about confidence levels of software quality. Computer Engineering and Design, 33, 2,607-611.
- Che G., Bai X., Liu Y. (2013).Survey on Health Management of Service-Based Software System. Journal of Frontiers of Computer Science and Technology, 7, 7, 577-591.
- Gong L. (2013).A model of software quality function deployment based on FAHP. Manufacturing Automati, 35, 1, 50-54.
- Gu Q., Chen D. (2014).Validation and simulation of software system evolution rules using software networks. Scientia Sinica (Informationis), 44, 1, 20-36.

- Khalili-Damghani K., Sadi-Nezhad S. (2013). A decision support system for fuzzy multi-objective multi-period sustainable project selection. *Computers & Industrial Engineering*, 64, 4, 1045-1060. Doi: [10.1016/j.cie.2013.01.016](https://doi.org/10.1016/j.cie.2013.01.016).
- Isaai M.T., Kanani A., Tootoonchi M. (2011). Intelligent timetable evaluation using fuzzy AHP. *Expert Systems with Applications*, 38, 4, 3718-3723. Doi: [10.1016/j.eswa.2010.09.030](https://doi.org/10.1016/j.eswa.2010.09.030).
- Li F., Wang F., Cong Q. (2014). Research Software Failure Analysis Methods and Software Quality Evaluation Model. *SOFTWARE*. 35, 1, 143+145.
- Mendonça Silva M., de Gusmão A.P.H., Poleto T. (2014). A multidimensional approach to information security risk management using FMEA and fuzzy theory. *International Journal of Information Management*, 34, 6, 733-740.
- Shen G., Huang Z., Xie B. (2016). Survey on Software Trustworthiness Evaluation: Standards, Models and Tools. *Journal of Software*, 27, 4, 955-968.
- Shi L., Huang K., Li M. (2014). Establishing the System of Software Confidence Evaluation Based on the Software Quality Model. *Journal of Huaibei Normal University (Natural Science)*, 35, 4, 57-63.
- Su Z., Li J. (2015). Evaluation Model on Quality of Software Based on Fuzzy Matter Element. *Computer & Digital Engineering*, 43, 11, 1909-1913+1941.
- Sun S., Xia C., Wang L. (2011). Survey of the measurement of software structures based on complex networks. *CAAI Transactions on Intelligent Systems*, 6, 3, 208-212.
- Yue F., Su Z., Lu Y. (2013). Comprehensive evaluation of software quality based on fuzzy soft sets. *Systems Engineering and Electronics*, 35, 7, 246-248.
- Zhou Z., Xia J., Wang Y. (2015). Research on quality management of software system. *Journal of Naval University of Engineering(Comprehensive Edition)*, 12, 4, 35-37.