Research on Fuzzy Extension Synthesis Metrics Algorithm for Software Quality

Jianli Dong
School of Computer Engineering, Huaihai Institute of Technology, Lianyungang 222005, China
dongjl1019@sina.com

Ningguo Shi
Lanzhou Resources and Environment Vocational and Technical College, Lanzhou 730021, China
sng@lzre.gs.edu.cn

***Abstract*—Based on the requirements and characteristics of software quality synthetical evaluation, the shortages and defects caused by original fuzzy synthesis evaluation algorithm and multilayer matter-element extension evaluation algorithm are systematically analyzed and discussed. These shortages are mainly resulted from the calculation of "taking large and taking small value" of the fuzzy comprehensive evaluation algorithm and the complex computing process of the multilayer matter-element extension evaluation algorithm, and also lead to the loss of a large amount of valid information and the multilayer transmission distortion of intermediate measurement data. Therefor, this paper extracts the excellent calculating structure from the original two algorithms and creates a new fuzzy extension synthesis evaluation method for software quality.**

***Index Terms*—fuzzy synthesis evaluation, software quality, matter-element model, extension evaluation algorithm, evaluation index system**

I. INTRODUCTION

Software quality is the basic guarantee and the lifeline for the development of the software industry. The fundamental goal of software engineering is to produce high-quality software products under the premise of controlling funds and pace. However, after decades of software engineering research, development and application software quality assurance has not fundamentally resolved. Even now, due to frequent accidents caused by software quality, especially such heavy casualties and damage in the military, economic, financial and national security applications, software quality situation remains worrying. Into the twenty-first century, the development of human society has entered the networking and the information age. The growing scale and high risk of investment, the complexity and difficulty of production are the basic characteristics of modern software industry development and now it is troubled by the bottleneck of the difficulty to ensure software quality. Decades of practice has proved that due to the particularity of software production, the problem of software quality assurance won't be solved completely overnight and requires developing constantly new techniques and methods to improve software quality gradually through long-term software engineering research and practice. Thus, software quality assurance is still the current difficulty and hot issue in the software engineering research and study[1-2].

Manuscript received June 10, 2010; revised August 20, 2010; accepted September 3, 2010.
This project is supported by Fund of Jiangsu University Natural Science Basic Research Project, Grant No. 08KJD520013.

II. THE STATUS OF SOFTWARE QUALITY METRICS

A. Metrics Methods and Standardization

In the process of the research and development of software engineering, a certain technology, methods and standards have also been gradually developed in software quality measurement and evaluation. Among them the most influential one is the software quality evaluation model put forward by Walters, MaCall, Boehm and others in 1978, as well as a series of software quality evaluation criteria formulated and promulgated by International Organization for Standardization in 1993, including ISO / IEC 14598, ISO / IEC 9126 [3-4].

The standardized software quality model puts forward various factors and the structure reflecting software products' quality, but how to determine a scientific and accurate method to achieve a quantitative measurement of software quality and the impartiality of evaluation is the key. Objectively speaking, up to now, due to the logical property, abstraction, complexity and large-scale of software products, there is still no a general algorithm of quantitative metrics and evaluation methods of software quality which is recognized and accepted by software engineering. Quantitative evaluation methods of software quality in current research and application, such as weighted average method, hierarchy analytic process, fuzzy synthesis evaluation method, matter-element model and extension evaluation, not only have their own advantages, but also have their own flaws. Using the basic calculation structures and ideas of fuzzy synthesis evaluation algorithm and matter-element extension evaluation algorithm, this paper will combine two calculation structures of the original two algorithms and

doi:10.4304/jsw.6.11.2099-2105

improves their respective flaws and shortcomings, creating a more scientific, reliable and practical new software quality fuzzy extension synthesis evaluation method[5-9].

The method, the quantitative way, not only can achieve a comprehensive evaluation of software quality, but also implement the sub-item evaluation of software quality characteristics, and the superiority and inferiority in software quality can be fully reflected from different levels and aspects.

B. Software Quality Evaluation Index System

In order to achieve a fuzzy extension synthesis evaluation method based on software quality, software quality evaluation index system suited to extension matter-element model and the fuzzy synthesis evaluation method and in accordance with standard metric model of software quality needs to be built.

Here, following the product-centric quality view, software quality metric model and the system of quality characteristics put forward by McCall and others are selected[4-5]. Furthermore, in the abstract sense this system consists of three levels in accordance with composition and affiliation constituting the three-class evaluation index system. The lower-class index is the refinement and decomposition of the upper-class index, as shown in Table 1. First class indexes (quality characteristics) abstracted and decomposed from relevant attributes of the entire software product quality are divided into product operation, product changes and product modifications, which numbered c_1, c_2 and c_3. The three indexes not only reflect different aspects constituting software quality attributes independently of each other, but also include and summarize all the factors and scope describing software quality. Second class indexes (sub-characteristics) are the refinement and decomposition of corresponding first class indexes, including 11 indexes, numbered c_{11}, c_{12}, …, c_{32}, c_{33}. Third class indexes(metric element), respectively, the refinement and decomposition of second class indexes, are atomic indexes directly used for measurement, a total of 41, numbered c_{111}, c_{112}, …, c_{332}, c_{333}. With regard to the weighted calculation of indexes at all levels AHP or expert evaluation method and other methods can be used to calculate level by level according to the impact that various indexes have on software quality or the upper-class index, and should be marked in the brackets after indexes. Thus, the entire software quality evaluation index system has been established.

III. MATTER-ELEMENT EXTENSION AND FUZZY EVALUATION METHODS OF SOFTWARE QUALITY

Fuzzy extension evaluation method with its novel theoretical system and measurement methods is the entirely new instrument for the realization of a synthesis evaluation of software quality. This method will now be discussed below[9-11].

A. Matter-Element Model of Software Quality Metrics

As shown in Table 1, once software quality evaluation index system is established, you can proceed with the establishment of matter-element model of quality characteristics (first class indexes) to meet the need of software quality extension evaluation method. In order to discuss conveniently and without loss of generality, set m quality characteristics (for example, in Table 1, m = 3), respectively, c_1, c_2, ..., c_m. According to the requirements of comprehensive evaluation of software quality, software quality evaluation criteria can be quantitatively divided into n-levels (such as excellent, good, qualified, unqualified) by an expert or through a statistical analysis method, and the value range of the evaluation index c_1, c_2, ..., c_m is also determined when software quality evaluation achieve a different level, thus matter element model of software quality evaluation (also known as "classical domain") is established as follows:

Table 1. software quality valuation indexes system

software	*first class index*	*second class index*	*third class index*
Software Products Quality	Product Operation c_1 (0.4)	correctness c_{11} (0.25)	traceability c_{111}(0.3) completeness c_{112}(0.35) consistency c_{113}(0.35)
		integrity c_{12} (0.2)	access control c_{121}(0.5) access audit c_{122}(0.5)
		Efficien-cy c_{13}(0.15)	execution efficiency c_{131}(0.6) storage efficiency c_{132} (0.4)
		usability c_{14} (0.2)	operability c_{141}(0.3) training c_{142}(0.2) communicativeness c143(0.2) i/o capacity c_{145}(0.15) i/o speed c_{146}(0.15)
		reliability c_{15} (0.2)	error-tolerance c_{151}(0.35) consistency c_{152}(0.3) accuracy c_{153}(0.2) simplicity c_{154}(0.15)
	product revision c_2 (0.3)	Maintainab-ility c_{21} (0.4)	consistency c_{211}(0.25) conciseness c_{212}(0.15) modularity c_{213}(0.25) self-descriptiveness c_{214}(0.2)
		flexibility c_{22} (0.25)	modularity c_{221}(0.3) consistency c_{222}(0.25) expandability c_{223}(0.25) self-descriptiveness c_{224}(0.2)
		testabilit yX_{23} (0.35)	simplicity c_{231}(0.2) modularity c_{232}(0.3) self-testability c_{233}(0.3) self-descriptiveness c_{234}(0.2)
	Product transition c_3 (0.3)	portability c_{31} (0.4)	modularity c_{311}(0.2) self-descriptiveness c_{312}(0.2) device-independency c_{313}(0.3) soft-independency c_{314}(0.3)
		reusability c_{32} (0.3)	generality c_{321}(0.15) modularity c_{322}(0.3) self-descriptiveness c_{323}(0.15) device-independency c_{324}(0.2) soft-independency c_{325}(0.2)
		interoperabi lity c_{33} (0.3)	modularity c_{331}(0.3) communication commonality c_{332}(0.35) data commonality c_{333}(0.35)

$$R_{0j} = \begin{bmatrix} N_{0j} & c_1 & V_{0j1} \\ & c_2 & V_{0j2} \\ & \cdots & \cdots \\ & c_m & V_{0jm} \end{bmatrix} = \begin{bmatrix} N_{0j} & c_1 & <a_{0j1}, b_{0j1}> \\ & c_2 & <a_{0j2}, b_{0j2}> \\ & \cdots & \cdots \\ & c_m & <a_{0jm}, b_{0jm}> \end{bmatrix}$$

Where: j = 1,2, ..., n; R_{0j} means matter-element model of the j-class software quality, N_{0j} refers to the j-class software quality, $V_{0jk}=<a_{0jk}, b_{0jk}>$ (k=1,2,··· ,m) indicates the value range or interval of c_k, the k-level or class evaluation index, when software quality achieves the j level. Here, intervals can be open, closed or half-open-closed.

In addition, the matter-element model constituted by the allowed entire value range of each index (the value interval of each index) of software quality comprehensive evaluation can be expressed as (known as the "section domain"):

$$R_p = \begin{bmatrix} N_p & c_1 & V_{p1} \\ & c_2 & V_{p2} \\ & \cdots & \cdots \\ & c_m & V_{pm} \end{bmatrix} = \begin{bmatrix} N_p & c_1 & <a_{p1}, b_{p1}> \\ & c_2 & <a_{p2}, b_{p2}> \\ & \cdots & \cdots \\ & c_m & <a_{pm}, b_{pm}> \end{bmatrix},$$

Where R_p means section domain of matter-element model of software quality comprehensive evaluation, N_p refers to all the grades of software quality evaluation results. $V_{pk}=<a_{pk}, b_{pk}>$ indicates the value range of the index ck in N_p, $V_{0jk} \subset V_{pk}$, j=1, 2, …, n; k=1, 2, …, m.

For software to be evaluated all indicators of the actual data or analysis results got through the actual measurement can be expressed using the following matter-element model to be valuated :

$$R = \begin{bmatrix} N & c_1 & v_1 \\ & c_2 & v_2 \\ & \cdots & \cdots \\ & c_m & v_m \end{bmatrix}$$, where: k=1,2,…,m; N refers to the quality of software to be evaluated, v_k indicates the first k-index measurement of software to be evaluated.

B. Extension Synthesis Evaluation Algorithm

After the establishment of matter-element model of software quality comprehensive evaluation, it is necessary to evaluate software quality in accordance with evaluation levels. Therefore, the "correlation" between matter-element model to be evaluated and the classical domain of matter-element model needs to be calculated. In practice, different methods of calculation of the "correlation" should be selected according to characteristics of indexes. Here elementary correlation function of extenics can be used. Order:

$$\rho(v_k, V_{0jk}) = \left| v_k - \frac{a_{0jk} + b_{0jk}}{2} \right| - \frac{1}{2}(b_{0jk} - a_{0jk}) ,$$

where k=1, 2, …, m; j=1, 2, …, n.

$$\rho(v_k, V_{pk}) = \left| v_k - \frac{a_{pk} + b_{pk}}{2} \right| - \frac{1}{2}(b_{pk} - a_{pk}) ,$$

where k=1, 2, …, m.

It respectively indicates the correlation of point v_k and intervaV$_{0jk}$, V_{pkl}. For example, when $\rho(v_k, V_{pk}) \ge 0$, it shows that vk is within the range of V_{pk}, while $\rho(v_k, V_{pk}) \le 0$, it shows that vk is not within the range of V_{pk}. And various negative values express vk is in different locations within the range of V_{pk}. Order:

$$D(v_k, V_{pk}, V_{0jk}) = \begin{cases} \rho(v_k, V_{pk}) - \rho(v_k, V_{0jk}) + a_{0jk} - b_{0jk}, v_k \in V_{0jk} \\ \rho(v_k, V_{pk}) - \rho(v_k, V_{0jk}) \qquad , v_k \notin V_{0jk} \end{cases}$$

which indicates the position (distance) between the point v_k with the two intervals V_{0jk}, V_{pk}. Note, interval V_{pk} and V_{0jk} has not shared endpoints.

Order: $K_j(v_k) = \dfrac{\rho(v_k, V_{0jk})}{D(v_k, V_{pk}, V_{0jk})}$, where: k=1, 2, …, m; j=1, 2, …, n. It means the correlation between c_k, the k-evaluation index of matter-element to be evaluated with the j-level evaluation results, $-\infty < K_j(v_k) < +\infty$. If $K_j(v_k) \ge 0$, it means that vk belongs toV_{0jk} and the larger $K_j(v_k)$ is, the more properties of V_{0jk} vk has; If $K_j(v_k) \le 0$, it means that v_k does not belong to V_{0jk} and the smaller $K_j(v_k)$ is, the farther vk is from the interval V_{0jk}.

Thereby the correlation matrix between various evaluation indexes of software to be evaluated and various evaluation levels can be calculated: $K=[K_j(v_k)_{m\times n}]$. According to the correlation matrix $K=[K_j(v_k)_{m\times n}]$, calculated as follows:

$$K^*(v_k) = K_{i_0}(v_k) = \max_{1 \le j \le n} K_j(v_k), \text{ k=1, 2, …, m.}$$

Then $K_{i_0}(v_k)$ refers to the i-level of evaluation of the k evaluation index and comprehensive evaluation of software quality can be decided by $K_{i_0}(v_k)$ As follows:

If $\alpha_i (\sum_{i=1}^{m} \alpha_i = 1)$ is the weight coefficient of software quality evaluation index, then the correlation between software to be evaluated and the first j-level evaluation results are:

$$K_j(R) = \sum_{i=1}^{m} \alpha_i K_j(v_{i)}),$$ Where: j = 1, 2, ... , n. then calculated as follows: $K_{j_0}(R) = \max_{1 \le j \le n} K_j(R)$. Thus j_0, the comprehensive evaluation rating result of software quality being assessed, can be available.

C. Fuzzy Synthesis Evaluation Algorithm

In order to facilitate the analysis, the single-level and multi-level fuzzy synthesis evaluation algorithm model are firstly listed below[8]. (Note: In the following discussion of fuzzy evaluation algorithm, the variable x has the same meaning as the quality evaluation index c of extension evaluation method. They only retain their own names).

1) Single-level Synthesis Evaluation Algorithm

a) Determining factor set of the object being evaluated:

$X=\{x_1, x_2, \ldots, x_n\}$.

b) Offering evaluation set (comments set):

$Y=\{y_1, y_2, \ldots, y_m\}$.

c) Building a fuzzy mapping from X to Y:

$\underset{\sim}{f}: X \rightarrow F(Y),\ x_1 \rightarrow r_{i1}/y_1 + r_{i2}/y_2 + \ldots + r_{im}/y_m$,

where: $0 \le r_{ij} \le 1$, i =1, 2, …, n; j=1, 2, …, m.

d) Obtaining the fuzzy evaluation matrix:

$\underset{\sim}{R} = (r_{ij})_{n\times m}$.

e) Determining the weight distribution of each factor:

$\underset{\sim}{A} = (a_1, a_2, \ldots, a_n)$, where: a≥0, $\sum_{i=1}^{n} a_i = 1$.

f) Synthesis evaluating: synthesis calculation applying fuzzy matrix:

$B = \underset{\sim}{A} \cdot \underset{\sim}{R} = (b_1, b_2, \ldots, b_m)$, where:

$$b_j = \bigvee_{i=1}^{n} (a_i \wedge r_{ij}),\ j = 1, 2, \ldots, m; \qquad (1)$$

And to seek the result of $\max\{b_1, b_2, \ldots, b_m\} = b_{j_0}$, $j_0 \in \{1, 2, \ldots, m\}$.

Evaluation conclusion: The review of the object being evaluated is b_{j_0}.

2) Multi-level Synthesis Evaluation Algorithml

Multi-level synthesis evaluation algorithm is mainly used for synthesis evaluation of complex event (system). The_principle is to decompose and refine evaluation factors from the higher to the lower level based on relevant attributes to form factors' multi-level affiliation, and to use single-level synthesis evaluation algorithm to judge from the lowest to the high level, until the highest level (objects being evaluated) achieve the conclusion. The specific algorithm model is as follows:

a) Dividing factor set

The factor set $X=\{x_1, x_2, \ldots, x_n\}$ can be divided into s sub-sets according to their elemental attributes, recorded as: $X_1, X_2, \ldots, X_s$. where $X_i = \{x_{i_1}, x_{i_2}, \ldots, x_{i_{q(i)}}\}$, $X_i \cap X_j = \phi$ (i≠j; I, j=1, 2, …, s), $\bigcup_{i=1}^{s} X_i = X$, $\sum_{i=1}^{s} i_{q(i)} = n$;

b) Calculating the synthesis evaluation result of X_i

Let evaluation set, $Y = \{y_1, y_2, \ldots, y_m\}$, the concentrated weight distribution for each factor of X_i is $\underset{\sim}{A}_i = (a_{i_1}, a_{i_2}, \ldots, a_{i_{q(i)}})$, where $\sum_{k=1}^{q(i)} a_{i_k} = 1$, ($a_{i_k} \ge 0$), and single-level fuzzy evaluation matrix for X_i is $\underset{\sim}{R}_i$. Thus the single synthesis assessment result of X_i can be drawn: $B_i = \underset{\sim}{A}_i \cdot \underset{\sim}{R}_i = (b_{i1}, b_{i2}, \ldots, b_{im})$ (i =1, 2, …, s), Where:

$$b_{ij} = \bigvee_{i=1}^{q(i)} (a_{i_k} \wedge r_{i_k j}),\ (j=1, 2, \ldots, m) \qquad (2)$$

c) Single-level synthesis evaluation

Using again each X_i as factor to form a set $\{X_1, X_2, \ldots, X_s\}$, and use B_i (i=1, 2, …, s) as the row vector of fuzzy evaluation matrix $\underset{\sim}{R}^*$ of single-level synthesis evaluation of higher-level, that is $\underset{\sim}{R}^* = (B_1, B_2, \ldots, B_s)^T = (b_{ij})_{s\times m}$, it is just the fuzzy evaluation matrix of the factor set $\{X_1, X_2, \ldots, X_s\}$. Taking the weight distribution of the factor set $\{X_1, X_2, \ldots, X_s\}$ as $\underset{\sim}{A}^* = (a_1, a_2, \ldots, a_s)$, so the synthesis evaluation results can be obtained:

$$B^* = \underset{\sim}{A}^* \cdot \underset{\sim}{R}^* = (b_1, b_2, \ldots, b_m), \text{ where:}$$

$$b_j = \bigvee_{i=1}^{s} (a_i \wedge r_{ij}),\ (j = 1, 2, \ldots, m) \qquad (3)$$

d) Repeat b) and c) steps, until obtaining fuzzy synthesis evaluation value of first-class index.

IV. THE DEFECTS ANALYSIS OF THE FUZZY AND EXTENSION EVALUATION ALGORITHM

In the process of the practical and separate application of above-mentioned matter-element extension valuation algorithm and fuzzy synthesis evaluation algorithm, we found the following shortcomings [12-13]:

A. Defects of Matter-Element Extension Algorithm

When using the extension algorithm of matter-element model for the implementation of the quantitative measurement and evaluation of software quality, the key is to build a variety of matter-element models of software quality evaluation. That is, classical domain, section domain and matter-element model to be evaluated. However due to the logic of software products and the complex relationships among many abstract quality attributes, software quality comprehensive evaluation index system generally consists of three or more levels. When matter-element model extension algorithm is used to implement the evaluation of multi-level complex quality index system, the difficulties directly encountered are: How is a multi-level matter-element model created? What is the calculation process of multi-level extension evaluation? How should the weight distribution matched with the process, the selection of correlation function, the division of evaluation classes and the determination of value range be considered and quantified? With the increasing levels of evaluation index system, these problems seem to be extremely complex and difficult. If they can't be quantified and handled, they will inevitably lead to a larger deviation of software quality evaluation result.

B. Defects of Fuzzy Synthesis Evaluation Algorithm

1) Loss of a large amount of valid information

The fuzzy matrix synthesis calculation is used for the comprehensive evaluation. That is, the operation defined by (1) formula in single-level model and (2), (3) formula in multi-level model. From the "comprehensive" point of view "selecting large and small value" is counter-productive of calculation result that a lot of valid information is lost. A certain aspect of quality and poor quality instead of the overall situation will inevitably lead to the bias and mistakes of comprehensive evaluation results. Especially in multi-level comprehensive evaluation of complex systems, a large amount of multi-level valid information is lost. More importantly, the ills of "a point on behalf of surface" caused by such calculation will be passed from the lowest level to the highest level, sometimes resulting in serious inaccuracy of comprehensive evaluation results and failure to achieve the goals of comprehensive evaluation. This defect also exists in extension evaluation algorithm.

2) The oversimplification and absolutes of the qualitative evaluation conclusion

From the qualitative method of final comprehensive evaluation conclusions, the conclusive comment, y_k for the object being judged is based on the formula $\max\{b_1, b_2, \ldots, b_m\} = b_{k_0}$ ($k_0 \in \{1, 2, \ldots, m\}$). This method itself is simply to take a larger value to evaluate. In fact, it is a certain point of quality and poor quality that takes the place of the overall situation and not a comprehensive measure evaluation. The shortcoming also exists in extension evaluation algorithms.

3) The Difficulty to Realize the Comparative Evaluation Between Similar Things

The single-level and multi-level comprehensive evaluation, as mentioned above *2)*, are only simple and absolute evaluations for the quality of individual things. It is very difficult to carry out the comparison evaluation between similar things, because such horizontal comprehensive comparison evaluation isn't involved in the comprehensive evaluation algorithm. However, in the actual comprehensive evaluation, more evaluations are comparative between similar things. Such as advanced workers, excellent products, scientific and technological achievements and so on. Therefore, this defect severely limits the application scope of comprehensive evaluation and reduces the evaluation practical value.

V. THE CREATION OF FUZZY EXTENSION SYNTHETICAL METRICS ALGORITHM

1) Determining the index system and evaluation set of the object being evaluated

Determine the evaluation index system (as shown in Table 1) of software quality to be evaluated and evaluation set Y based on the standard model of software quality measurement[12, 14].

2) Determining the weights of evaluation index at all levels

The weight of each evaluation index is achieved through AHP or Delphi Method or other mathematical statistical methods. It shouldn't be freely given subjectively. As shown in Table 1, the weight is offered through the three cycles of statistics of Delphi method.

3) Establishing matter-element model of first class indexes

Under the guidance of experts in the field of software evaluation, establish the first class (ie the highest level index) matter-element model of classical domain and section domain. The first class matter-element model to be evaluated will be calculated and got directly through fuzzy comprehensive evaluation method.

4) Establishing matter-element model of third class indexes

Build third class indexes'(metric element) classical domain matter-element model.

5) Changing fuzzy matrix synthesis operation to the general matrix multiplication operation

In order to effectively implement integrated measure in the comprehensive evaluation algorithm and prevent the loss of a large amount of valid information, fuzzy matrix synthesis operations defined by the formula $b_j = \bigvee_{i=1}^{s} (a_i \wedge r_{ij})$ in the comprehensive evaluation algorithm model can be changed into the ordinary matrix operations defined by the formula $b_j = \sum_{i=1}^{n} (a_I \cdot r_{ij})$ ($j = 1, 2, \ldots, m$). In essence, the calculation of "taking large and taking small value" is changed to "matrix multiplication" operation identified by (1), (2), (3) in the single-level and multi-level comprehensive evaluation algorithm model.

6) Determining the fuzzy membership function of third class indexes

After determining the classical domain matter-element model of third class (metric element), the measurement value of each of third class indexes can be obtained adopting metric element measurement tools and formulas offered by RADC (Rome Air Development Center) [15]. These metric element measure values are from the most objective, accurate and basic atomic attribute values of software being evaluated, but not as fuzzy membership degree values of third class indexes for evaluation grades.

Therefore, according to the distribution of metric element measure values of software quality and evaluation experience, the membership function of metric element for different levels of evaluation can be regarded as normal distribution function:

$$\mu(x) = e^{-(\frac{x-m}{c})^2} \quad (4)$$

In the formula m and c are constants. The value range for variable x is restricted to the classical domain range of matter-element model of third class indexes, and with μ (x) fuzzy membership degree value for third class indexes at classical domain evaluation levels can be determined. Constants m and c: As μ (x) = 1, taking the maximum value of membership degree. Therefore, m can be used as the mid-point value (average) of the classical domain value range, ie $m=(x_l+x_r)/2$. x_l, x_r are the left and the right end points of the classical domain interval of third class indexes. Furthermore, deduced by the formula μ (x), when u (x) is at the critical point of classical domain value range of matter-element model of two adjacent evaluation levels, two membership degree values for two adjacent reviews should be the same, approximately equal to 0.5. When $e^{-(\frac{x_r - x_l}{2c})^2} = 0.5$, we can find the value of c in different classical domain range.

So far, the values of constants m and c of fuzzy membership degree function μ (x) in different range of third class indexes classical domains have been determined. Through RADC method the specific measure value for each of third class indexes can be obtained and then fuzzy membership degree value of third class indexes for different evaluation levels can be determined by the formula (4). Thus fuzzy membership matrix of third class indexes is available.

7) Implementing fuzzy evaluation calculation from the lower to the higher level

After the fuzzy membership matrix of third class indexes is determined, in accordance with fuzzy multi-level comprehensive evaluation method implement fuzzy comprehensive evaluation level by level from metric element (third class indexes) to quality characteristics (first class indexes) until work out the quality evaluation value of first class indexes (quality characteristics) in the software quality evaluation index system, thus matter-element model to be evaluated of software quality can be available. In other words, the specific measure values of quality characteristics are calculated level by level from the bottom to the top through fuzzy comprehensive evaluation method.

8) Implementing extension algorithm evaluation

When matter-element model to be evaluated of first class indexes is established, one can implement software quality comprehensive evaluation using matter-element model extension evaluation method from first class indexes of the evaluation system for a start. Up to now, fuzzy extension synthesis evaluation method of software quality has been created.

9) The indication of software quality evaluation results

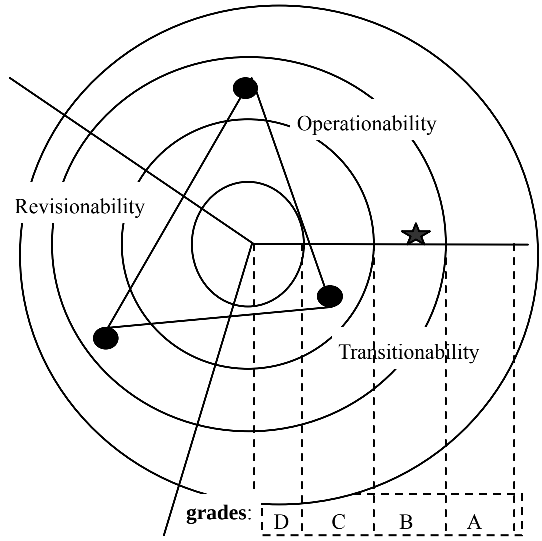

Figure 1. Kiviat figure of evaluation result

The reresentaion and analysis of the final comprehensive evaluation conclusion of software quality being evaluated is an important task in the later eriod of software quality evaluation. Generally speaking, software quality evaluation result is very important for the demand-side and development side because the correct and effective denotation and comprehensive analysis of the evaluation conclusion can help the parties have a clear understanding of the quality and oor quality of software development in various aspects, which is critical to strengthen software management and to improve software development capabilities, the quality of software development and market competitiveness. At present, software quality evaluation result of quantitative measurement take many forms, such as ercentile figures, text descriptions of excellent, good, qualified and unqualified. As shown in Figure 1, the radar chart (Kiviat) is used here to represent it graphically. (solid dots reresent evaluation grades of quality characteristics, the solid five-pointed star represent comprehensive evaluation grades of software quality). It offers relevant assessment staff an intuitive and effective method to examine comprehensive evaluation result of software quality and the quality level of each quality characteristic.

VI. CONCLUSION

It should be noted that when simply using matter-element model and extension evaluation method to implement quantitative measurement and evaluation of software quality, the key is to build various matter-element models of extension, namely classical domain, section domain and matter-element model to be evaluated. However, with the increasing levels of quality evaluation index system and numbers of indexes, when creating extension matter-element model and calculating level by level and moving evaluation results, the weight distribution, the identification of correlation function calculation match, the division of evaluation grading as well as the determination of indexes' value range, which are matching to such calculation rocess, must be taken into consideration overall and quantified. It seems to be particularly complex and cumbersome. But the fuzzy comprehensive evaluation method has the same computing capacity and evaluation effectiveness, in addition, its transmission of calculation level by level and moving is simpler and easier. Therefore, in this aer extension evaluation method and fuzzy evaluation method are merged into a new fuzzy extension evaluation method. It draws the advantages of two algorithms and abandons the shortcomings, after all, it is a good way for software quality comprehensive evaluation[16][17].

ACKNOWLEDGMENT

This project is supported by Fund of Jiangsu University Natural Science Basic Research Project, Grant No. 08KJD520013. Hereon, we should give thanks to the friends and colleagues who help and support our work. We also specially thank the leaders of the school of computer engineering.

REFERENCE

[1] YANG FuQing, “Thinking on the Development of Software Engineering Technology,” Journal of Software(in Chinese), 2005, 16(1), pp.1-7.
[2] BEN Keron, “Outlined Of Software Quality Assurance Technologies Research”, Journal of Naval University of Engineering(in Chinese), 2002, 14(4), pp.1-6.
[3] YIN Ping, “The Analyze of Software Quality Evaluation Based on ISO,” Software Engineering and Standardization (in Chinese), 2005, No.12, pp.37-41.
[4] XU Guanghui, JIN Fenglin, DING Li, The method and practice of software engineering (in Chinese), Beijing: China Machine Press, 2004, pp.186-191.
[5] LIANG Chengcai, TANG Wei, XIAO Liwen, “A Quantitative Assessing Method for Software Quality,” Computer Engineering (in Chinese), 2003, Vol.29(14), pp.95-97.
[6] CAI Xiyao, “On Software Quality Metrics,” Computer Engineering & Science (in Chinese), 1996, No.1, pp.1-6.
[7] Stephen H.Kan, Metrics and Models in Software Quality Engineering, Boston Addison-Wesley, 2003.
[8] WANG Fan. Fuzzy Math and Engineering Science (in Chinese), Harbin: Harbin shipping engineering college Press, 1998.
[9] YANG Chunyan, CAI Wen, Extension Engineering(in Chinese). Beijing: Science Press, 2007.
[10] CAI Wen, “Introduction of Extenics,” System Engineering Theory & Practice (in Chinese), 1998, No.1, pp.76-84.
[11] YANG Chunyan, CAI Wen. “Study on Extension Engineering,” Engineering Science (in Chinese), 2000.12, Vol.12, No.112, pp.90-96.
[12] Dong Jianli, Ningguo Shi, “An Improved Fuzzy Synthesis Evaluation Algorithm for Software Quality,” IEEE Computer Society, ICIII 2009, Vol.2, pp.565-569.
[13] XIAO Zhi, ZHONG Bo, LI Youhui, “A Synthetic Evaluation Matter-element Model and Extension Evaluating Method for Business Competitive Capacity”, Operations Research and Management Science, 2002.4, Vol.11, No.2, pp70-75.
[14] Dong Jianli, Ningguo Shi, “Multilayer Matter-Element Extension Synthesis Evaluation of Software Quality,” IEEE, ICBECS 2010, pp.693-697.
[15] AD-A153 990, “SPECIFICATION OF SOFTWARE QUALITY ATRIBUTES Software Quality Evaluation Guidebook”, RADC-TR-85-37, Vol.Ⅱ-Ⅲ, Final Technical Report, 1985.
[16] Yang Aimin, Zhang Wenxiang, “Based on Quantification Software Quality Assessment Method,” JOURNAL OF SOFTWARE, DECEMBER 2009, VOL. 4, NO. 10, pp.1110-1117.
[17] Witold Pedrycz and Giancarlo Succi, “ Genetic granular classifiers in Modeling software quality,” Journal of Systems and Software, 2005, Vol.76, No.3, pp.277-285.

Jianli DONG was born in Shanxi province, China, in 1957. He got his Bachelor of Mathematics Science in Northwest Normal University, Lanzhou, Gansu province, China, in 1988 and got his Master of Software Engineering in Beijing University of Aeronautics and Astronautics, Beijing, China, in 1995. He is now a professor at the School of Computer Engineering in HuaiHai Institute Technology, Lianyungang, China. He has published 40 papers, and completed many reseach projects, and won 6 times scientific and technological progress awards from the provance and military.

Mr. DONG current research interests include software engineering, integrated software engineering enviroment, software architecture, engineering database system, object-oriented technology.

Ningguo SHI was born in Linxia of Gansu province of China on 11, 1963. He graduated from Xi’an Mining Institute majoying in mine survey in Bechelor Degree in 7, 1985. Now he is a president and a professor in Lanzhou Resources & Environment Voc-Tech College. Pro. Shi is one of first leading talents of Gansu province. Also he is an expert who is entitled to speical grants from the National Council of China, an evaluation expert of Ministry of Education for assessing the talent cultivating work in higher vocational colleges. He gets outstanding acdemic achievements: the first prize in Gansu Science and Technology Progress Award for one time; the Provincial Teaching Achievement prize for one time; the first prize in Science and Technology Progress Award of many provincial deparrtments for two times.

Mr. SHI current research interests focuses on constructing and developing an excellent course — Mine Survey, and computer technology and application, software engineering, database system, information technology, etc.