A New Approach of Software Quality Metrics: Fuzzy Extension Algorithm and Its Application

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

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

Wenbin Hu
School of Computer Engineering, Huaihai Institute of Technology, Lianyangang, 222005, P.R.China.
hwb1008@163.com

Abstract—According to the needs and characteristics of software quality comprehensive assessment, the author systematically analyzes the drawback of the loss of a large amount of effective information caused by the calculation in the fuzzy synthesis evaluation algorithm as well as the defects of distortion among the intermediate data transformation in the processing of multi-level extensive evaluation algorithm. For that reason, this paper combines the two original algorithms and improves their shortcomings. Finally, an innovative software quality fuzzy extension synthesis evaluation method is created. Practice shows that adopting this new method to implement software quality evaluation not only has the advantages of accurate, reliable and practical but also the characteristics of easy to achieve computable and standardized.

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 of the development of the software industry. The fundamental goal of software engineering is to produce high-quality software products in 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, accidents caused by software quality occurred frequently, especially such heavy casualties and damage in the military, economic, financial and national security applications, the situation of software quality remains worrying all the same. Into the twenty-first century, the development of human society has entered the network and information age. The growing scale and high risk of investment, the complexity and difficulty of production has been the basic characteristics of modern software production and difficult to ensure software quality is the bottleneck that troubled the development of modern software industry. Decades of practice proved that the problems of software quality assurance would not be solved completely overnight due to the special nature of software products, it 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 difficult and hot issue of the software engineering currently [1, 2].

II. THE STATUS OF SOFTWARE QUALITY METRICS

A. Some Problems of Existing Metrics Methods

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, McCall, 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].

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 field. The current quantitative metrics methods of software quality, such as weighted average method, hierarchy analytic process described in Ref.[4, 5], fuzzy synthesis evaluation method described in Ref.[6, 7]. Furthermore, Chinese expert Wen CAI and his colleagues proposed the matter-element extension evaluation method which is discussed in Ref.[8, 9], etc., only have their own advantages, but also have their own defects which is described in Ref.[10, 11, 12] in detail. So, more accurate, reliable and practical metrics methods have been researched for software quality [13].

The paper will extract the basic calculation structures and novel ideas of the original fuzzy and matter-element extension evaluation algorithm, and improve their respective defects, and finally create a more reliable and...
A practical new software quality fuzzy extension synthesis evaluation method. Based on the application results, it can be seen that the new approach can achieve better evaluation for software quality.

B. Software Quality Evaluation Index System.

In order to achieve fuzzy extension synthesis evaluation method for software quality, we need to establish a kind of software quality evaluation index system which must be adapt to extension matter-element model and fuzzy synthesis evaluation as well as the software quality standard measurement model.

Here, following the product-centric quality view, the software quality measurement model and the quality characteristics system put forward by McCall et al. are adopted in this paper [4, 5]. Furthermore, this system is abstracted to three levels to consist of three-class evaluation index system according to composition and affiliation constituting. At the same time, the lower-class index is the refinement and decomposition of the upper-class index, as shown in Table 1. The first-class index (quality characteristics) is the abstract and decomposition of relevant attributes of entire software product quality, which can be divided into product operation, product changes and product modifications and numbered by c1, c2 and c3, respectively.

The three indexes not only reflect the different aspects of software quality attributes independently but also include and summarize all the factors and scope describing software quality. The second-class indexes (sub-characteristics) are the refinement and decomposition of corresponding first-class indexes, which include 11 indexes and are numbered by c11, c12, …, c33, respectively.

The third-class indexes (metric element) is the refinement and decomposition of corresponding second-class indexes, which are atomic indexes that can be directly used for measurement and include 41 indexes that can be numbered by c111, c112, …, c332, c333, respectively. With regard to the calculation of indicators weight at all levels, it should be calculated step by step using the methods such as AHP (Analytic hierarchy process) or rotational expert evaluation method according to the impact of various indexes affecting on software quality or the upper-class index, which should be marked in the brackets after indexes. Thus, the entire software quality evaluation index system is completed.

III. FUZZY AND MATTER-ELEMENT EXTENSION EVALUATION ALGORITHM INTRODUCTION

For creating a new fuzzy extension evaluation method, original matter-element extension valuation algorithm and original fuzzy synthesis algorithm is firstly introduced below. And then their advantages and defects will be analyzed and discussed. Final a new fuzzy extension synthesis valuation algorithm is created [9-11].

<table>
<thead>
<tr>
<th>Software Product Operation</th>
<th>Software Product Quality</th>
<th>Third Class Index</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correctness</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c11 (0.25)</td>
<td>Traceability</td>
<td>c11 (0.3)</td>
</tr>
<tr>
<td>Correctness</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c12 (0.2)</td>
<td>Complementness</td>
<td>c12 (0.35)</td>
</tr>
<tr>
<td>Correctness</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c13 (0.2)</td>
<td>Consistency</td>
<td>c13 (0.35)</td>
</tr>
<tr>
<td>Efficiency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c21 (0.15)</td>
<td>Execution efficiency</td>
<td>c21 (0.6)</td>
</tr>
<tr>
<td>Efficiency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c22 (0.2)</td>
<td>Storage efficiency</td>
<td>c22 (0.4)</td>
</tr>
<tr>
<td>Efficiency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c23 (0.2)</td>
<td>Accessibility</td>
<td>c23 (0.25)</td>
</tr>
<tr>
<td>Efficiency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c24 (0.2)</td>
<td>Consistency</td>
<td>c24 (0.15)</td>
</tr>
<tr>
<td>Efficiency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c25 (0.2)</td>
<td>Modularity</td>
<td>c25 (0.25)</td>
</tr>
<tr>
<td>Efficiency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c26 (0.2)</td>
<td>Expandability</td>
<td>c26 (0.25)</td>
</tr>
<tr>
<td>Efficiency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c27 (0.2)</td>
<td>Self-descriptiveness</td>
<td>c27 (0.2)</td>
</tr>
<tr>
<td>Efficiency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c28 (0.2)</td>
<td>Simplicity</td>
<td>c28 (0.2)</td>
</tr>
<tr>
<td>Efficiency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c29 (0.2)</td>
<td>Self-testability</td>
<td>c29 (0.2)</td>
</tr>
<tr>
<td>Efficiency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c30 (0.2)</td>
<td>Soft-independency</td>
<td>c30 (0.2)</td>
</tr>
<tr>
<td>Efficiency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c31 (0.2)</td>
<td>Communication</td>
<td>c31 (0.3)</td>
</tr>
<tr>
<td>Efficiency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c32 (0.2)</td>
<td>Information commonality</td>
<td>c32 (0.35)</td>
</tr>
<tr>
<td>Efficiency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c33 (0.2)</td>
<td>Data commonality</td>
<td>c33 (0.35)</td>
</tr>
</tbody>
</table>

A. Matter-Element Model of Software Quality Metrics

As shown in Table 1, once software quality evaluation index system is established, the matter-element model for quality characteristics (the first-class index) needed by software quality extension evaluation algorithm can be built. In order to discuss conveniently and without loss of generality, maybe the number of quality characteristics can be assumed to be m (such as in Table 1, m=3), which could be denoted by c1, c2, …, cm respectively. According to the requirements of comprehensive evaluation, software quality evaluation grades can be quantitatively divided into n-levels (such as excellent, good, qualified, unqualified) by software evaluation experts or through statistical analysis methods, and the value range of the evaluation index c1, c2, …, cm can also be determined when software quality achieves to different levels. Thereby, the matter-element model of software quality evaluation (also known as "classical domain") can be established as follows:
It shows that \( v_k \) is not in the range of \( \rho \) which \( \rho \leq \) shows that \( v_k \) is within the interval \( V_{0jk} \) indicates the value range (Value interval) of the index \( c_k \) in \( N_p \), \( V_{pk} = \langle a_{pk}, b_{pk} \rangle \) indicates the value range (Value interval) of the index while the level of software quality is \( j \). Here, the range can be open, closed or half-open-closed.

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

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

Where \( R_p \) means the section field of matter-element model of software quality comprehensive evaluation, \( N_p \) refers to the j-class matter-element model of software quality, \( V_{0jk} = \langle a_{0jk}, b_{0jk} \rangle, k=1, 2, \ldots, m \) is the value range of \( c_k \) which is the k-th evaluation index while the level of software quality is \( j \). Here, the range can be open, closed or half-open-closed.

For the software to be evaluated, the actual measurement data or analysis results of all indicators can be expressed by the following matter-element model:

\[
R = \begin{bmatrix}
N & c_1 & V_1 \\
& c_2 & V_2 \\
& \ldots & \ldots \\
& c_m & V_m
\end{bmatrix}, \text{ where } k=1, 2, \ldots, m; \text{ N refers to the quality of software to be evaluated, } v_k \text{ indicates the measurement value of the k-th index of software to be evaluated.}

B. Matter-Element Extension Evaluation Algorithm

After the establishment of the above matter-element model of software quality comprehensive evaluation, it is necessary to assess the quality of software to be evaluated according to 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, the calculation of "correlation" should be selected on the basis of the characteristics of each index. Here, the elementary correlation function of extenics is used. Suppose that:

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

where \( k=1, 2, \ldots, m; j=1, 2, \ldots, n \).

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

where \( k=1, 2, \ldots, m \).

They indicate the "correlation" between point \( v_k \) and interval \( V_{0jk}, V_{pk} \) respectively. For example, \( \rho(v_k, V_{pk}) \geq 0 \) shows that \( v_k \) is not in the range of \( V_{pk} \), while \( \rho(v_k, V_{pk}) \leq 0 \) shows that \( v_k \) is within the range of \( V_{pk} \) and various negative values express the different location of \( v_k \) in the range of \( V_{pk} \). The suppose that:

\[
D(v_k, V_{pk}, V_{0jk}) = \rho(v_k, V_{pk}) - \rho(v_k, V_{0jk})
\]

indicates the location of point \( v_k \) between interval \( V_{0jk} \) and interval \( V_{pk} \).

\[
K_j(v_k) = \frac{\rho(v_k, V_{0jk})}{D(v_k, V_{pk}, V_{0jk})}, \text{ where } k=1, 2, \ldots, m; j=1, 2, \ldots, n.
\]

It means the correlation degree of the k-th index \( c_k \) of matter-element to be evaluated and the j-th level evaluation results, \(-\infty < K_j(v_k) < \infty \).

\( K_j(v_k) \geq 0 \) means that \( v_k \) belongs to \( V_{0jk} \) and the larger \( K_j(v_k) \) is, the more properties of \( V_{0jk} \) \( v_k \) has; \( K_j(v_k) \leq 0 \) means that \( v_k \) does not belong to \( V_{0jk} \) and the smaller \( K_j(v_k) \) is, the farther \( v_k \) is from the interval \( V_{0jk} \).

Thereby, the correlation matrix of the 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 \( K = [K_{j}(v_k)_{m \times n}] \), it can be calculated as follows:

\[
K'(v_k) = K_i(v_k) = \max_{1 \leq j \leq n} K_j(v_k), \text{ where } k=1, 2, \ldots, m.
\]

Then \( K_i(v_k) \) means the k-th evaluation index of software to be evaluated is in the i0-th level assessment, then the comprehensive evaluation of software quality can be decided by \( K_i(v_k) \). And the approach is described as follows:

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

\[
K_j(R) = \sum_{i=1}^{m} \alpha_i K_j(v_{i,j}). \text{ Where } j=1, 2, \ldots, n, \text{ then calculating } K_{j0}(R) = \max_{1 \leq j \leq n} K_j(R). \text{ Thus, we can obtain that the comprehensive evaluation grade of the quality of the software be assessed is at the level of } j_0 \text{ [9-11].}
C. Fuzzy Synthesis Evaluation Algorithm

In order to facilitate the analysis, single-level and multi-level fuzzy synthesis evaluation algorithms are listed below firstly [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) To Determine the Factor Set of Evaluation Object: \( X=\{x_1, \ x_2, \ \ldots, \ x_n\} \);

b) Given the Evaluation Set (Comments Set): \( Y=\{y_1, \ y_2, \ \ldots, \ y_m\} \);

c) To Build a Fuzzy Mapping From X to Y:
\[
f: X \rightarrow F(Y), \ x_i \rightarrow r_{i1}/y_1 + r_{i2}/y_2 + \ldots + r_{im}/y_m,
\]
where: \( 0 \leq r_{ij} \leq 1, \ i=1, 2, \ldots, n; \ j=1, 2, \ldots, m. \)

d) To Write the Fuzzy Evaluation Matrix:
\[
R=(r_{ij})_{n \times m}.
\]
e) To Determine the Weight Distribution of Each Factor:
\[
A=(a_1, \ a_2, \ \ldots, \ a_n), \ a \geq 0, \ \sum_{i=1}^{n}a_i = 1.
\]
f) To Implement Comprehensive Evaluation with Fuzzy Matrix:
\[
B=A \cdot R=(b_1, \ b_2, \ \ldots, \ b_m), \ b_j = \sum_{i=1}^{n}(a_i \cap r_{ij}), \ j=1, 2, \ldots, m.
\]
And to seek the result of \( \max\{b_1, b_2, \ldots, b_m\} = b_j_0 \), \( j_0 = 1, 2, \ldots, m \).
Evaluation conclusion: The review of the object being evaluated is \( b_{j_0} \).

2) Multi-level Synthesis Evaluation Algorithm

Multi-level synthesis evaluation algorithm is mainly used for the comprehensive evaluation of complex issues (system). The principle is to decompose and refine the evaluation factors from the higher to the lower level on the basis of relevant attributes to form factors’ multi-level affiliation, and then judging from the lowest to the high level using single-level synthesis evaluation algorithm and stopping until the highest level (evaluation objects) achieved the conclusion. The algorithm model is as follows:

a) To Divide Factor Set:

The factor set \( X=\{x_1, \ x_2, \ \ldots, \ x_n\} \) can be divided into s sub-sets according to their element attributes, denoted as: \( X_1, X_2, \ \ldots, X_s \) where \( X_i = \{x_{i1}, \ x_{i2}, \ \ldots\} \), \( X \cap X_j = \phi \ (i \neq j; i, j = 1, 2, \ldots, s) \),
\[
\bigcup_{i=1}^{s}X_i = X,
\]
\[
\sum_{i=1}^{s}i(y_i) = n.
\]
b) To Get the Evaluation Results of Each \( X_i \) According to Signal-Level Synthesis Evaluation Algorithm:

Let evaluation set be \( Y=\{y_1, \ y_2, \ \ldots, y_m\} \), the weight distribution of each factor centralized by \( X_i \) is \( A_i=(a_{i1}, \ a_{i2}, \ \ldots, a_{i(s_i)}) \), where \( \sum_{k=1}^{s_i}a_{ik} = 1, \ (a_{ik} \geq 0) \), and the single-level fuzzy evaluation matrix of \( X_i \) is \( R_i \). Thus, the single-level synthesis assessment result of \( X_i \) can be drawn: \( B_i = A_i \cdot R_i = (b_{i1}, b_{i2}, \ldots, b_{im}) \) \( (i=1, 2, \ldots, s) \).
Where:
\[
b_{ij} = \sum_{j=1}^{m}(a_{ij} \cap r_{ij}), \ (i=1, 2, \ldots, s; j=1, 2, \ldots, m)
\]

(2)
c) Single-Level Synthesis Evaluation:

using each \( X_i \) as the element to form the set \{ \( X_1, X_2, \ldots, X_i \) \} once again and adopting \( B_i \) \( (i=1, 2, \ldots, s) \) to be the row vector of fuzzy evaluation matrix \( R^* \) of the advanced signal-level synthesis evaluation of these factors. That is \( R^*=(B_1, B_2, \ldots, B_s)=(b_{ij})_{s \times m} \) is the fuzzy evaluation matrix of the factor set \{ \( X_1, X_2, \ldots, X_i \) \}. Taking the weight distribution of the factor set \{ \( X_1, X_2, \ldots, X_s \) \} as \( A^*=(a_1, a_2, \ldots, a_n) \), then, the synthesis evaluation results can be obtained: \( B^* = A^* \cdot R^* = (b_{1}, b_{2}, \ldots, b_{m}) \), where:
\[
b_{ij} = \sum_{j=1}^{m}(a_{ij} \cap r_{ij}), \ (j=1, 2, \ldots, m)
\]

(3)
d) To Calculate Final Fuzzy Synthesis Evaluation:

Repeating step b) and c) until obtain the fuzzy synthesis evaluation value of first-level index.

IV. CREATING FUZZY EXTENSION EVALUATION ALGORITHM

The two algorithms described above, the fuzzy and extension evaluation algorithms, separately used to evaluate software quality will produce some adverse effects. In [10, 12], all defects are described. For this reason, we extract the advantages and create a new fuzzy extension synthesis evaluation algorithm, which is now described below.

1) To Determine the Evaluation Index System and grade Set of the Object Evaluated:
To 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.

2) To Determine the Weights of Evaluation Indexes at All Levels:
The weight of each evaluation index is achieved through AHP, Delphi (Delphi Method) and 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) To Establish the First-Class Indexes of Matter-Element Model:
Under the guidance of evaluation experts in the field of software, the classical domain and section field matter-element model of the first-class index (i.e. the highest level index) is established. The first-class matter-element model to be evaluated would be calculated and directly obtained by fuzzy synthesis evaluation method.

4) To Establish the Matter-Element Model of the Third-Class Indexes:
To build the classical domain matter-element model of the third-class indexes (metric element).

5) To Change Fuzzy Matrix Operations to the General Matrix Operations:
In order to effectively implement integrated measurement in the comprehensive evaluation algorithm and prevent the loss of a large amount of valid information, fuzzy matrix synthesis operations defined by the formula $b_j = \sum_{i=1}^{n} (a_{ij} \land r_{ij})$ in the comprehensive evaluation algorithm model can be changed to the ordinary matrix operations defined by the formula $b_j = \sum_{i=1}^{n} (a_{ij} \cdot r_{ij})$ (j=1, 2, ..., m). Essentially, it is to identify the calculation of "selecting large and small value" defined by the formula (1), (2) and (3) introduced in signal-level and multi-level comprehensive evaluation algorithm model by the "matrix multiplication" operation.

6) To Determine the Fuzzy Membership Function of the Third-Class Indexes:
After determining the classical domain matter-element model of the third class indexes (metric element), the measuring value of each of third-class indexes can be obtained by metric element measurement tools and formulas provided by RADC (Rome Air Development Center) [15]. These measurement values of metric element are original from the most objective, accurate and basic atomic attribute values of evaluation software, but they can not be used for the fuzzy membership values of metric element to evaluation grades. Therefore, according to the distribution of metric element measurement values of software quality and the evaluation experience, the membership function of metric element for different levels of evaluation can be regarded as normal distribution function:

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

Where: m and c are constants, the value range of variable x is restricted to the classical domain interval of matter-element model of third class indexes, $\mu(x)$ is used to determine the fuzzy membership value of the third-class indexes measurement values at classical domain evaluation grades. The definition of constant m and $c: \mu(x)=1$ and the value of membership degree is maximum, therefore, m could be see as the middle-point value (average value) of the interval of classical domain, that is, $m=(x_l + x_r) / 2$. $x_l$, $x_r$ are the left and the right end of the third-class indexes classical domain interval. Furthermore, deduced by the formula $\mu(x)$, 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. That is, when $e^{-\left(\frac{x_r-x_l}{2c}\right)^2} = 0.5$, we can obtain the value of c in different classical domain range.

So far, the values of constants m and c of fuzzy membership function $\mu(x)$ in different interval of third class indexes classical domains have been determined. With RADC method, the specific measurement value of each third-class indexes can be obtained and then fuzzy membership value of the third-class indexes for different evaluation levels can be get through the formula (4). Thus, fuzzy membership matrix of the third-class indexes is available.

7) To Implement Fuzzy Evaluation Calculation From the Lower to the Higher Level:
After the fuzzy membership matrix of the third-class indexes is determined, on the basis of multi-level fuzzy synthesis evaluation the fuzzy comprehensive assessment could be carry out layer by layer from metric element (the third-class indexes) to quality characteristics (the first-class indexes) until the quality evaluated value of the first-class indexes (quality characteristics) in the software quality evaluation system, thus, the matter-element model to be evaluated of software quality can be available. In other words, the specific measurement values of quality characteristics are calculated level by level from the bottom to the top through fuzzy synthesis evaluation approach.

8) To Implement the Extension Algorithm Evaluation:
When the matter-element model to be evaluated of the first-class indexes is established, one can implement software quality comprehensive evaluation using matter-element model extension assessment method from the first-class indexes of the evaluation system for a start. Up to now, fuzzy extension synthesis evaluation method of software quality has been accomplished.

V. THE APPLICATION OF THE FUZZY EXTENSION SYNTHESIS EVALUATION FOR SOFTWARE QUALITY
The application example of the fuzzy extension synthesis evaluation method for software quality is addressed below.

1) To Determine the Evaluation Index System and Grade Set of Software Quality to Be Evaluated:

The evaluation index system of software quality being evaluated is shown in Table 1. The weights of each index have also been given. Evaluating grade set $Y = \{\text{excellent, good, fair, poor, bad}\}$.

2) To Establish Matter-Element Models of the First-Class and the Third-Class Indexes:

Due to space limitations and understanding easily, the authors demonstrate the calculation process of fuzzy extension algorithm only by establishing matter-element models depending on parts of indexes. Suppose that the classical domain and section field of matter-element model of the first-class indexes are as follows (Note: in order to meet the practices of matter-element model and fuzzy evaluation, we take $c$ as the name of index in matter-element model and take $x$ as the name of index in the fuzzy evaluation. In fact, they are the same. As shown in Table 1, it can be considered that $c_1 = x_1$, $c_2 = x_2$, ..., $c_{333} = x_{333}$):

$$R_{01} = \begin{bmatrix} N_{01} & c_1 \prec 0.821 > \\ c_2 \prec 0.831 > \\ c_3 \prec 0.821 > \end{bmatrix},$$

$$R_{02} = \begin{bmatrix} N_{02} & c_1 \prec 0.62, 0.82 > \\ c_2 \prec 0.70, 0.83 > \\ c_3 \prec 0.60, 0.82 > \end{bmatrix},$$

$$R_{03} = \begin{bmatrix} N_{03} & c_1 \prec 0.45, 0.62 > \\ c_2 \prec 0.50, 0.70 > \\ c_3 \prec 0.46, 0.60 > \end{bmatrix},$$

$$R_{04} = \begin{bmatrix} N_{04} & c_1 \prec 0.35, 0.45 > \\ c_2 \prec 0.36, 0.50 > \\ c_3 \prec 0.35, 0.46 > \end{bmatrix},$$

$$R_{05} = \begin{bmatrix} N_{05} & c_1 \prec 0.35 > \\ c_2 \prec 0.36 > \\ c_3 \prec 0.35 > \end{bmatrix},$$

$$R_p = \begin{bmatrix} N_p & c_1 \prec 0.1 > \\ c_2 \prec 0.1 > \\ c_3 \prec 0.1 > \end{bmatrix}.$$

The classical domain value interval of matter-element model of the third-class indexes $c_{111}$, $c_{112}$ and $c_{113}$ are shown in Table 2.

<table>
<thead>
<tr>
<th>third class indexes</th>
<th>excellent</th>
<th>good</th>
<th>fair</th>
<th>poor</th>
<th>very poor</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_{111}$</td>
<td>(0.85, 1)</td>
<td>(0.75, 0.8)</td>
<td>(0.55, 0.7)</td>
<td>(0.40, 0.5)</td>
<td>(0.0, 0)</td>
</tr>
<tr>
<td>$c_{112}$</td>
<td>(0.86, 1)</td>
<td>(0.74, 0.8)</td>
<td>(0.57, 0.7)</td>
<td>(0.35, 0.5)</td>
<td>(0.0, 0.3)</td>
</tr>
<tr>
<td>$c_{113}$</td>
<td>(0.84, 1)</td>
<td>(0.75, 0.8)</td>
<td>(0.60, 0.7)</td>
<td>(0.40, 0.6)</td>
<td>(0.0, 0.4)</td>
</tr>
</tbody>
</table>

According to the calculation method of constants $m$ and $c$ in the formula (4), the constant values of every classical domain interval corresponded to $m$ and $c$ can be obtained and are shown in Table 3.

3) To Determine the Fuzzy Membership Matrix of the Third-Class Indexes:

According to multi-level fuzzy comprehensive evaluation process, the fuzzy evaluation matrix corresponding to the third-class indexes (metric element) in software quality evaluation index system (Table 1) must be determined firstly. The elements of fuzzy matrix are made of each membership degree of elements in conclusion set $Y$. Here, evaluation factors measurement algorithm studied by U.S. Air Force RADC (Rome Air Development Center) is adopted. Then using software measurement tools to measure actually, actual measured values of the third-class indexes can be available. Due to space limitation, here only " $c_{111}$ " = the traceable demand number / the total demand number, $c_{112}$ = the sum of ratings meeting the terms of completeness / the total number of terms, $c_{113}$ = 1 - the module number violating rules / the module number" are given as the calculation example. The measurement of the remaining third-class indexes will not be repeated here, please refer to the definition of RADC data.

Through the metric elements measuring formula of RADC, we can obtain that $c_{111}$ = 0.73, $c_{112}$ = 0.89, $c_{113}$ = 0.65. Then, according to Table 2, Table 3 and formula (4), membership values for each evaluation grade can be obtained, which are shown in Table 4.

<table>
<thead>
<tr>
<th>third class indexes</th>
<th>excellent</th>
<th>good</th>
<th>fair</th>
<th>poor</th>
<th>very poor</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_{111}$</td>
<td>0.73</td>
<td>0.01</td>
<td>0.26</td>
<td>0.64</td>
<td>0.00</td>
</tr>
<tr>
<td>$c_{112}$</td>
<td>0.89</td>
<td>0.21</td>
<td>0.01</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>$c_{113}$</td>
<td>0.65</td>
<td>0.00</td>
<td>0.93</td>
<td>0.21</td>
<td>0.03</td>
</tr>
</tbody>
</table>
4) To Evaluate the First-Class Indexes From the Lower to the Higher Level:

Fuzzy membership matrix of the third-class indexes are \( c_{11}, c_{12}, \ldots, c_{16} \), which can be directly obtained from Table 4. It is as follows:

\[
R_{11} = \begin{bmatrix}
0.01 & 0.26 & 0.64 & 0.00 & 0.00 \\
0.80 & 0.21 & 0.01 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.93 & 0.21 & 0.03
\end{bmatrix}
\]

From the weights of the index system in Table 1, it can be available that \( A_1 = (0.30, 0.35, 0.35) \).

Fuzzy membership values of the second-class indexes can be obtained through the fuzzy matrix. As regards to \( c_{11} \), the fuzzy evaluation result of \( c_{11} \) can be got through the following formula.

\[
A_1 \cdot R_{11} = (0.30, 0.35, 0.35) \cdot \begin{bmatrix}
0.01 & 0.26 & 0.64 & 0.00 & 0.00 \\
0.80 & 0.21 & 0.01 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.93 & 0.21 & 0.03
\end{bmatrix}
\]

\[
= (0.28, 0.15, 0.52, 0.07, 0.01)
\]

Here, the evaluation results can be directly used as fuzzy membership value of the second-class index \( c_{11} \). Furthermore, fuzzy membership values of all the second-class indexes can be obtained in this way. Merging the membership values of the second-class index related to the first-class indexes into a fuzzy membership matrix and selecting the weights of the second-class indexes from Table 1, the fuzzy evaluation results of the first-class indexes can be achieved according to fuzzy operator. Take the first-class index \( c_1 \) for example, the fuzzy membership matrix resulting in the combination of the second-class index \( c_{11}, c_{12}, c_{13}, c_{14} \) and \( c_{15} \) can be expressed as follows:

\[
R_1 = \begin{bmatrix}
0.28 & 0.15 & 0.52 & 0.07 & 0.01 \\
0.22 & 0.62 & 0.12 & 0.05 & 0.01 \\
0.12 & 0.63 & 0.18 & 0.06 & 0.02 \\
0.14 & 0.16 & 0.62 & 0.05 & 0.03 \\
0.12 & 0.65 & 0.15 & 0.06 & 0.02
\end{bmatrix}
\]

It can be resulted from Table 1 that the weight matrix of \( c_{11}, c_{12}, c_{13}, c_{14}, c_{15} \) and \( c_{16} \) is \( A_1 = (0.25, 0.20, 0.15, 0.20, 0.20) \). Then, the fuzzy calculation results of \( c_1 \) can be obtained by \( A_1 \cdot R_1 = (0.18, 0.42, 0.34, 0.06, 0.02) \). Assumed that the fuzzy results of indexes \( c_2, c_3 \) calculated by the similar method is \((0.15, 0.48, 0.30, 0.05, 0.02)\) and \((0.12, 0.26, 0.44, 0.14, 0.04)\), respectively. Then, the maximum calculation results and its corresponding evaluation level are taken as the fuzzy calculated values of the first-class indexes, which is logged by \( c_{0k} \). 0j stands for the assessment level, \( k \) refers to \( c_k \). Fuzzy calculated value of index \( c_1 \) is \( \max(0.18, 0.42, 0.34, 0.06, 0.02) = 0.42 \).

In order to maintain the consistency among fuzzy calculated value, matter-element model and size of extension data an avoid the data distortion in the transmission and calculation process of fuzzy extension algorithm, the formula, \( v_k = \frac{a_{0jk} \cdot c_{0jk}}{b_{0jk} - a_{0jk}} \), can be used to obtain measurement values of matter-element model to be evaluated of the first-class indexes. Where, \( v_k \) means the measurement value of \( c_k \), \( (b_{0jk} - a_{0jk}) \) is the range of the classical domain of matter-element model that \( c_k \) corresponded at the grade of the 0j. Then, the measurement value of \( c_1 \) is \( 0.62 + (0.82 - 0.62) \cdot 0.42 = 0.704 \). Similarly, the value of \( c_2 \) and \( c_3 \) could be calculated. The matter-element model to be evaluated may be formed as follows:

\[
R = \begin{bmatrix}
N & c_1 & 0.704 & c_2 & 0.762 & c_3 & 0.522
\end{bmatrix}
\]

5) To Calculate Final Extension Evaluation Results

After determining the classical domain, section field and matter-element model to be evaluated of the first-class indexes by the above method, the matter-element model extension algorithm can be applied to evaluate software quality. The correlation matrix obtained by using correlation function is:

\[
K = [K_j(v_k)]_{3 \times 5}
\]

\[
= \begin{bmatrix}
-0.28 & 0.40 & -0.22 & -0.46 & -0.55 \\
-0.22 & 0.35 & -0.21 & -0.52 & -0.63 \\
-0.38 & -0.14 & 0.15 & -0.12 & -0.27
\end{bmatrix}
\]

Thus available: \( K^*(v_1) = K_2(v_1) = 0.40 \), \( K^*(v_2) = K_3(v_2) = 0.35 \), \( K^*(v_3) = K_4(v_3) = 0.15 \). Thus, the evaluation conclusions of each quality characteristics can be acquired, including per-formability, modifiability which is the second-level (i.e. good) and transferability which is the third-level (i.e. fair or middle).

From Table 1, the weights of the first-class indicators are, respectively, \( A = (0.40, 0.30, 0.30) \). Then, using \( A \cdot K = (-0.29, 0.22, -0.11, -0.38, -0.49) \), obviously, we can know the correlation degree of evaluation software are \( K_1(R) = -0.29 \), \( K_2(R) = 0.22 \), \( K_3(R) = -0.11 \), \( K_4(R) = -0.38 \), \( K_5(R) = -0.49 \). May be we can select \( K_5(R) = \max_{j=1,2,3,4,5} K_j(R) = 0.22 \). Then, the comprehensive evaluation grade of the software could be measured out quantitatively, which is at level 2, that is, the total rated as good.

6) The Indication of Software Quality Evaluation Results.

The representation and analysis of the final comprehensive evaluation conclusion of the evaluated software quality is an important work in the later period of software quality evaluation. Generally speaking, software quality evaluation results are very important for
the demand side and development side because the correct and effective denotation and comprehensive analysis of the evaluation conclusion can help to make a clear understanding of the quality in various aspects, which is essential to strengthen software management, improve the ability of software development, enhance the quality of software development and competitiveness in the market. Currently, software quality quantitative measurement evaluation results has many kinds of expression forms, such as digital representation of percentile, text description ranked in excellent, good, qualified and unqualified. In this paper, the radar figure (Kiviat) which is shown in Fig.1 is used to express it graphically (the solid dots represent the evaluation grades of quality characteristics, the solid five-pointed stars represent comprehensive evaluation grades of software quality), which provide 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 process, 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 better method for software quality comprehensive evaluation [14].

REFERENCE


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 some research projects, and won 6 times scientific and technological progress awards from the province and military.

Mr. DONG current research interests include software engineering, integrated software engineering environment, 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.

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.