

A Fuzzy Model for Evaluating Cultivation Quality of Talents of Software Engineering at the Campus Universities

Yongzhong Lu¹, Danping Yan², Bo Liu¹

¹School of Software Engineering, Huazhong University of Science and Technology, Wuhan 430074, P. R. China; ²School of Public Administration, Huazhong University of Science and Technology, Wuhan 430074, P. R. China.
Email: hotmailuser@163.com

Received January 6th, 2009; revised February 9th, 2009; accepted March 2nd, 2009.

ABSTRACT

In order to measure the quality of talent cultivation at the school of software engineering, a quality evaluation model based on fuzzy theory is put forward. In the model, a three-layer architecture, which is composed of overall goal layer, second goal layer, and attribute layer, is set up. It places emphasis on the demand of talents with practicability and engineering in the field of software engineering. Then a case is used in the model to illustrate its effectiveness. The experimental results show that the model can comparatively better evaluate the quality of talent cultivation, reach the expected objective, and fulfill the practical demand. According to the model, a quality evaluation software system is developed while a rainfall lifecycle development model and Microsoft Visual C++ Development Studio are utilized.

Keywords: software engineering, cultivation quality evaluation, fuzzy computing model

1. Introduction

In order to fulfill the urgent social demands of software talents with high quality, practical experiences and comprehensive engineering skills in China, we have carried out a series of reform and innovation pertaining to the teaching contents and approaches, courses system, and management institution and operational mechanism. Up to now, we have come to deeply recognize that training talents of software engineering is similarly deemed to a item of talent production project. In the course of the teaching reform and innovation, it is significantly vital to lay emphasis on its training quality and effect which are the progress signpost in the forthcoming days. Generally speaking, it is rather difficult to measure the quality and effect of bringing up software talents quantitatively because they are closely related to numerous determinants [1,2]. Therefore, an accurate quality evaluation model about the training project of software talents at the universities is still not set up. Based on the social demands for software talents in China, we first put forward a qualitative model of quality evaluation of talent cultivation at the universities, and then exploit a fuzzy approach to give the quantitative computational results. Subsequently a case is used to testify its effectiveness. At last, a quality evaluation software system is developed while a rainfall lifecycle development model and Microsoft Visual C++ Development Studio are utilized.

2. A Fuzzy Quality Evaluation Model for Software Talents

We have referred to the generic ability evaluation standard

of engineering graduates in UK, the USA and other European developed countries [3,4,5,6,7,8]. In addition, we have combined it with present practical situation at the campus schools and amended it properly. As a result, a quality evaluation model of training software talents is presented in Figure 1. In the model there are three layers: the top one is called overall objective layer and expressed by matrix A, the middle layer is called second objective layer and expressed by matrix B, and the lowest layer is called third attribute layer and expressed by matrix C, but it does not mean this layer is no importance. The corresponding statements are shown in Table 1.

3. A Fuzzy Evaluation Approach

It's quite difficult to get the exact values of the attributes in the model above. The fuzzy evaluation approach adapts to solve the problem well. Therefore it is used here to work out the solution to the problem. Its process is described as follows.

1) Establish the evaluation expert group

Different types of software experts are adopted to probe into the quality of training the software talents. They are usually composed of several experts such as field experts, senior managers, and users, and so forth. After the selection of evaluation expert group, a comment set is required to be determined. Supposing that the hierarchical rank of software products is classified into five levels which correspond to a comment set $V = (v_1, v_2, v_3, v_4, v_5)$ (“excellent”, “good”, “medium”, “passed”, “bad”).

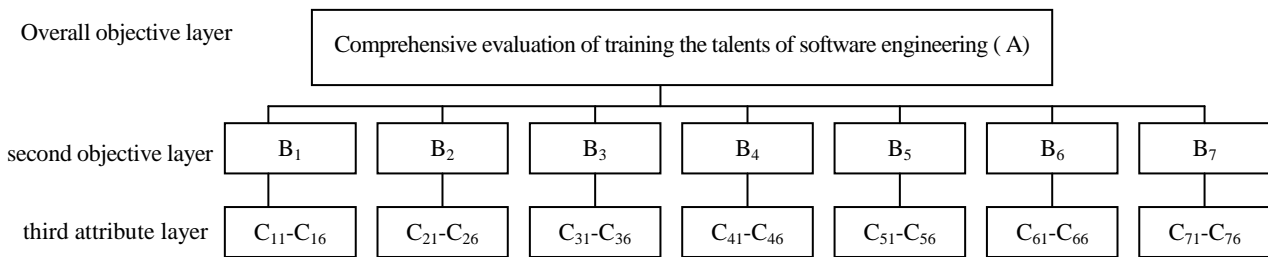


Figure 1. An evaluation model of training the talents of software engineering

Table 1. The generic statements corresponding to the Figure 1

1 Ability to exercise Key Skills in the completion of software engineering-related tasks at a level implied by the benchmarks associated with the following statements (B ₁) a) Communication (C ₁₁) b) Information Technology (C ₁₂) c) Application of Number (C ₁₃) d) Working with Others (C ₁₄) e) Problem Solving (C ₁₅) f) Improving Own Learning and Performance (C ₁₆)
2 Ability to transform existing software systems into conceptual models (B ₂) a) Elicit and clarify client's true needs (C ₂₁) b) Identify, classify and describe software engineering systems (C ₂₂) c) Define real target software systems in terms of objective functions, performance specifications and other constraints (ie, define the problem) (C ₂₃) d) Take account of risk assessment, and social and environmental impacts, in the setting of constraints (including legal, and health and safety issues) (C ₂₄) e) Resolve difficulties created by imperfect and incomplete information (C ₂₅) f) Derive conceptual models of real target software systems, identifying the key parameters (C ₂₆)
3 Ability to transform conceptual models into determinable models (B ₃) a) Construct determinable models over a range of complexity to suit a range of conceptual models (C ₃₁) b) Use mathematics and computing skills to create determinable models by deriving appropriate constitutive equations and specifying appropriate boundary conditions (C ₃₂) c) Use industry standard software tools and platforms to set up determinable models (C ₃₃) d) Recognise the value of Determinable Models of different complexity and the limitations of their application (C ₃₄)
4 Ability to use determinable models to obtain system specifications in terms of parametric values (B ₄) a) Use mathematics and computing skills to manipulate and solve determinable models; and use data sheets in an appropriate way to supplement solutions (C ₄₁) b) Use industry standard software platforms and tools to solve determinable models (C ₄₂) c) Carry out a parametric sensitivity analysis (C ₄₃) d) Critically assess results and, if inadequate or invalid, improve knowledge database by further reference to existing software systems, and/or improve performance of determinable models (C ₄₄)
5 Ability to select optimum specifications and create physical models (B ₅) a) Use objective functions and constraints to identify optimum specifications (C ₅₁) b) Plan physical modelling studies, based on determinable modelling, in order to produce critical information (C ₅₂) c) Test and collate results, feeding these back into determinable models (C ₅₃)
6 Ability to apply the results from physical models to create real target software systems (B ₆) a) Write sufficiently detailed specifications of real target software systems, including risk assessments and impact statements (C ₆₁) b) Select production methods and write method statements (C ₆₂) c) Implement production and deliver products fit for purpose, in a timely and efficient manner (C ₆₃) d) Operate within relevant legislative frameworks (C ₆₄)
7 Ability to critically review real target software systems and personal performance (B ₇) a) Test and evaluate real software systems in service against specification and client needs (C ₇₁) b) Recognise and make critical judgements about related environmental, social, ethical and professional issues (C ₇₂) c) Identify professional, technical and personal development needs and undertake appropriate training and independent re-search(C ₇₃)

2) Determine the single weights of the statements

AHP (Analytical Hierarchy Process) is adopted to figure out the weights of the statements. The detailed steps are followed below.

- According to the model above, a proper questionnaire is well-prepared for the experts. They determine the mutual weights among the statements in three layers. The weight matrix between overall objective layer A and second objective layer B_i is shown in Table 2. The matrix is usually called determinant matrix. We can obtain other determinant matrixes in the same way. Thereafter they fill out the comments about the attribute layer statements as Table 3.

- Construct the single determinant matrix

The AHP constructs the determinant matrix by terms of relationship among the statement items, and their proportional scales are among 1-9 [9]. Supposing that A represents the object set, U the evaluation item set, u_i ($i=1,2,\dots,n$) the evaluation item, and u_{ij} represents mutual weight between u_i and u_j ($j=1,2,\dots,n$), the determinant matrix is expressed below.

$$U = \begin{matrix} & u_1 & u_2 & \cdots & u_i & \cdots & u_n \\ \begin{matrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{matrix} & \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1j} & \cdots & u_{1n} \\ u_{21} & u_{22} & & u_{2j} & & u_{2n} \\ \cdots & \cdots & & \cdots & & \cdots \\ u_{n1} & u_{n2} & & u_{nj} & & u_{nn} \end{bmatrix} \end{matrix} \quad (1)$$

- Calculate the normalized weights of all evaluation items above

The geometric average method is used to gain the eigenvector corresponding to the most characteristic root λ_{\max} of matrix U above. And it is normalized and shaped into the weights of all evaluation items. The detailed formula is following

$$W_i = \left(\prod_{j=1}^n u_{ij} \right)^{1/n} / \sum_{i=1}^n \left(\prod_{j=1}^n u_{ij} \right)^{1/n} \quad (2)$$

where $i, j = 1, 2, \dots, n$. The result $W = (W_1, W_2, \dots, W_n)^T$ is the above-mentioned eigenvector.

Table 2. Weight matrix of A and B

A	B ₁	B ₂	B ₃	B ₄	B ₅	B ₆	B ₇
B ₁							
B ₂							
B ₃							
B ₄							
B ₅							
B ₆							
B ₇							

Table 3. Subjection degrees about attribute layer statements

Attributer layer statements	Comment set				
	excellent	good	Medium	passed	bad
C ₁₁					
C ₁₂					
C ₇₃					

- Consistency testing

Supposing that U is a matrix with n ranks, u_{ij} ($1 \leq i \leq n, 1 \leq j \leq n$) is an element in U , if all elements of U have a property of transitivity, that is to say $u_{ij} \times u_{jk} = u_{ik}$, the matrix U is called a consistency matrix. A consistency matrix can be verified by the formula (3)

$$CR = CI / RI \quad (3)$$

where CR is called the random consistency ratio of the determinant matrix, RI is called the average random consistency ratio of the determinant matrix, and CI is called the general consistency item which can be expressed by the formula (4)

$$CI = (\lambda_{\max} - n) / (n - 1) \quad (4)$$

where n is the rank of the determinant matrix. λ_{\max} is decided by the following formulae (5) and (6)

$$\lambda_{\max} = \frac{1}{n} \sum_{i=1}^n \frac{(PW)_i}{W_i} \quad (5)$$

$$PW = \begin{bmatrix} (PW)_1 \\ (PW)_2 \\ \dots \\ (PW)_n \end{bmatrix} = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1j} & \cdots & u_{1n} \\ u_{21} & u_{22} & \cdots & u_{2j} & \cdots & u_{2n} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ u_{n1} & u_{n2} & \cdots & u_{nj} & \cdots & u_{nn} \end{bmatrix} \begin{bmatrix} W_1 \\ W_2 \\ \dots \\ W_n \end{bmatrix} \quad (6)$$

when $CR < 0.10$, it can be concluded that the determinant matrix has a satisfactory property of consistency, that is to say that the distributed weights are proper, vice versa.

- Calculate the comprehensive weights

The distributed weights of the second objective layer to the third attribute layer are obtained by the formula (3). The distributed weights of the overall objective layer to the second objective layer is calculated by the formula (7)

$$W = \sum_{j=1}^n WB_j WC_{ij} \quad (7)$$

where WB_j is the important weight of B_j ($1 < j < 7$) corresponding to A , and WC_{ij} is the important weight of C_{ij} corresponding to B_j . When B_j has no bearing with C_{ij} , $WC_{ij} = 0$.

3) Determine the subjection degrees of the quality evaluation

When carrying out the evaluation of talent cultivation of software engineering, field experts, together with senior manager (policy-makers) and customers, give the decisive subjection degree according to the defined comment set above. It can explicitly be expressed by the subjection degree matrix R below

$$R = (r_{ij})_{m \times k} \quad (8)$$

where r_{ij} is the percentage of regarding the i -th evaluation statement as the j -th comment class. And it is also

expressed by $r_{ij} = d_{ij}/d$ where d_{ij} is the number of the members of drawing the conclusion that the i -th evaluation statement belongs to the j -th comment class, d is the total of the members, m is the number of the statements, and k is the evaluation rank.

4) Calculate the final evaluation result

After attaining the subjection degree matrix R , we calculate the comprehensive evaluation vector S of talent cultivation of software engineering. Then we adopt the Weighted Average Model of comprehensive evaluation— $M(*,+)$ in order to consider all relevant factors appropriately and remain their information. The comprehensive evaluation vector S and the comprehensive evaluation result P are displayed in (9) and (10) respectively

$$S = W_c^a \times R \tag{9}$$

$$P = V \times S^T \tag{10}$$

In the formula (9), W_c^a is the comprehensive weights of third attribute layer C corresponding to overall objective layer A . As a result, the quality level of talent cultivation of software engineering at campus universities can easily be performed by the formula (10) and the task of quality evaluation of talent cultivation of software engineering is successfully completed.

4. Illustration

In order to testify the effectiveness of the presented model above, we take a practical case for example. Based on the model, we perform the demonstration in accordance with the following steps.

1) Calculate the single weights of the statements

The AHP is exploited to construct the single determinant matrix as Table 2 and normalized by Formula (2). Then the consistency testing is done by Formula (3). If the CR is less than 0.1, the comprehensive determinant matrix is obtained by Formula (7). The two results are shown in Tables 4 and 5.

2) Calculate the subjection degree matrix

After the mutual weights of three layers are decided,

15 relevant members give their evaluation opinions to the quality of talent cultivation of software engineering with the aid of the comment set above. The subjection degree matrix $R_{30 \times 5}$ is gotten by Formula (8) and normalized into Formula (11).

$$R = \begin{bmatrix} 0.195 & 0.636 & 0.564 & 0.081 & 0.455 & 0.091 & 0.182 & \dots \\ 0.455 & 0.564 & 0.345 & 0.273 & 0.273 & 0.345 & 0.455 & \dots \\ 0.564 & 0 & 0.091 & 0.345 & 0.273 & 0.273 & 0.564 & \dots \\ 0 & 0 & 0 & 0.091 & 0 & 0.091 & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots \end{bmatrix} \tag{11}$$

3) Calculate the comprehensive evaluation value

$$S = W_c^a \times R = (0.2175, 0.4635, 0.2123, 0.0415, 0.067) \tag{12}$$

$$P = V \times S^T = (5, 4, 3, 2, 1) \times (0.2175, 0.4635, 0.2123, 0.0415, 0.067)^T = 3.728 \tag{13}$$

From Formulae (12) and (13), we find that if the subjection degree is 0.2175, the quality is excellent; if the

Table 4. The single weights of the second objective layer B corresponding to the third attribute layer C

		B ₁ (0.476)				B ₂ (0.266) ...	
C ₁₁	C ₁₂	C ₁₃	C ₁₄	C ₁₅	C ₁₆
0.299	0.141	0.105	0.168	0.127	0.160

Table 5. The comprehensive weights of the second objective layer B and the third attribute layer C corresponding to the overall objective layer A

		B ₁ (0.476)				B ₂ (0.266) ...	
C ₁₁	C ₁₂	C ₁₃	C ₁₄	C ₁₅	C ₁₆
0.138	0.089	0.049	0.078	0.059	0.096

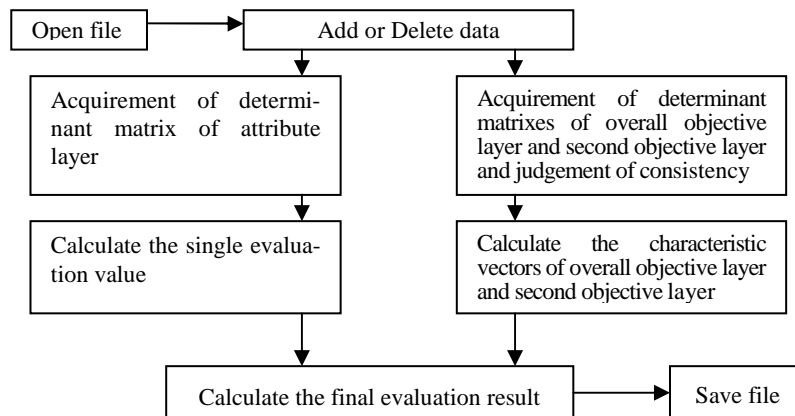


Figure 2. The system workflow

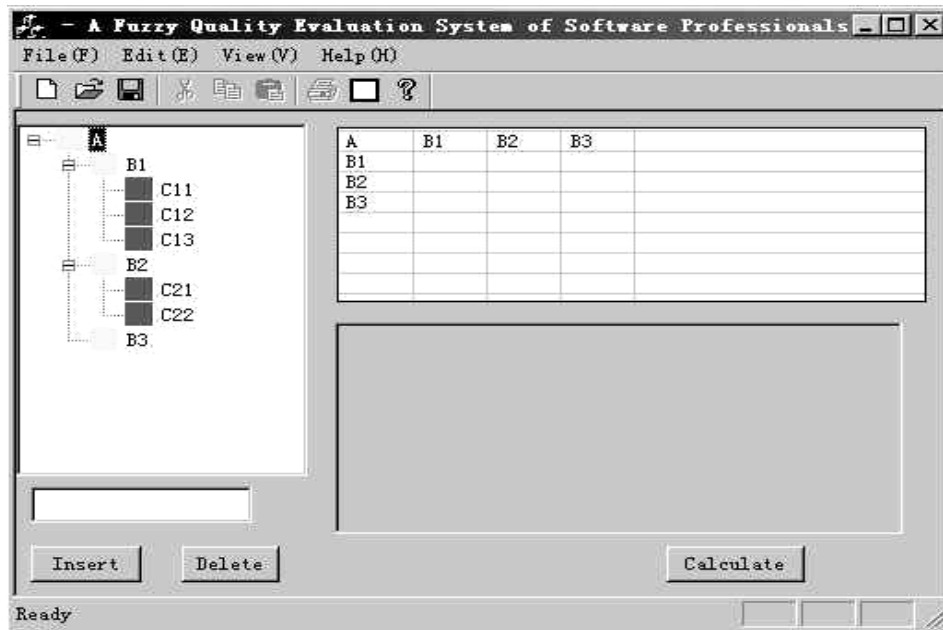


Figure 3. The quality evaluation system

subjection degree is 0.4635, the quality is good; if the subjection degree is 0.2123, the quality is medium; if the subjection degree is 0.0415, the quality is passed; if the subjection degree is 0.067, the quality is bad. If $V = \{5,4,3,2,1\}$ is quantified, the comprehensive evaluation value is 3.728 and its final evaluation quality is “medium”.

5. Developing the Quality Evaluation System

The workflow of the quality evaluation system is described as Figure 2. In the figure, we divide the system into five modules which include Add or Delete module, Calculate the single evaluation value of certain attribute module, Consistency testing module, Calculate the characteristic vector module, and Calculate the final evaluation value module.

The quality evaluation software system is developed as Figure 3 while a rainfall lifecycle development model and Microsoft Visual C++ Development Studio are utilized.

6. Conclusions

Based on the quality evaluation model of talent training of software engineering, a fuzzy quality evaluation system of talent training of software engineering is developed. It can easily measure the quality level of talent cultivation of software engineering and provide a good evaluation platform for software talent cultivation. However, some aspects on the consistency testing and determinant matrix construction will be further addressed in the future.

7. Acknowledgement

The support from the Natural Science Foundation at

Huazhong University of Science and Technology grant number 2007Q006B is gratefully acknowledged for this work by the authors.

REFERENCES

- [1] J. S. J. Lin, “Study of university curriculum development for human resource management and the expectations of business managers,” *International Journal of Business and Systems Research*, 2(4), pp. 418–430, 2008.
- [2] P. Xiao, Y. Cheng, and Y. Yang, “Higher education human resources development criterion inquiry,” *Future and Development*, 29(2), pp. 9–12, 2008.
- [3] N. E. Gibis, “The SEI education program the challenge of teaching future software engineers,” *Communications of ACM*, 32(5), pp. 594–605, 1989.
- [4] O. Hazzan, “The reflective practitioner perspective in software engineering education,” *Journal of Systems and Software*, 63(3), pp. 161–171, 2002.
- [5] A. J. Cowling, “What should graduating software engineers be able to do?” *IEEE Proceedings of the 16th Conference on Software Engineering Education and Training*, pp. 88–98, 2003.
- [6] D. T. Holt and Mitchell Crocker, “Prior negative experiences: their impact on computer training outcomes,” *Computers & Education*, 35(4), pp. 295–308, 2000.
- [7] M. M. Boulet, C. Dupuis, and N. Belkhit, “Selecting continuous training program and activities for computer professionals,” *Computers & Education*, 36(1), pp. 83–94, 2001.
- [8] D. Garlan, “Making formal methods education effective for professional software engineers,” *Information and Software Technology*, 37(5–6), pp. 261–268, 1995.
- [9] Q. Wang, “Practical fuzzy mathematics,” *Science and Technology Documentary Press*, Peking, 1995.

(Edited by Vivian and Ann)