A simplified measurement scheme for software quality

Che-Wei Chang *

Department of Information Management*

Cheng-Ru Wu †

Department of Finance*

Hung-Lung Lin ‡

Graduate Institute of Business Management*

* Yuanpei University
306 Yuanpei St., Hsin Chu 300
Taiwan
R.O.C.

Abstract

An approach of software quality evaluation model and its computing algorithm is proposed in this paper. The software support can reduce barriers of application of MCDM procedures. In this study, we propose the software quality measurement. The relevant criteria of this model are derived from the international norm ISO 9126-1. AHP is a measurement theory that prioritizes the hierarchy and consistency of judgemental data provided by a group of decision makers. AHP incorporates the evaluations of all decision makers into a final decision, without having to elicit their utility functions on subjective and objective criteria, by pair-wise comparisons of the alternatives. The Analytic Hierarchy Process (AHP)-based decision-making method can provide decision makers or purchasers a valuable reference for evaluating software quality. Importantly, the proposed model can assist the user and developer to assess the software quality, making it highly applicable for academia and commercial purposes.

Keywords and phrases : Software quality characteristics, software quality model, evaluating the software quality, analytic process, MADM.

* E-mail: chewei@mail.yust.edu.tw
† E-mail: alexru004@ms41.hinet.net
‡ E-mail: hsa8936.hsa8936@msa.hinet.net

Journal of Information & Optimization Sciences
Vol. 27 (2006), No. 3, pp. 723–732
© Taru Publications 0252-2667/06 $2.00 + 0.25
1. **Introduction**

Among the challenges encountered in software development include devising quality software, accurately controlling overhead costs, complying with a progress schedule, maintaining the software system, coping with unstable software systems and satisfying consumer demand with respect to software quality. Above challenges may incur a crisis in software development if not effectively addressed. Problems within the software sector can be summed up as follows:

(a) Inability to accurately forecast or control software costs.
(b) Substandard quality, low reliability and ambiguous requests on how to enhance software quality.
(c) Unnecessary risks while offering and maintaining quality assurance.
(d) High personnel turnover rate, leading to human error in software development.
(e) To resolve the above problems, multi-attribute characteristics or factors of software quality must be considered. Effective management strategies in a technology setting are thus essential to resolving crises in software development. The technological aspect refers to the software technology adopted and design expertise of a software developer.

The AHP-based decision-making method to construct an evaluation method can provide decision makers, user and buyer with a valuable reference for either evaluating the software quality to identify the most appropriate quality in develops the software system. Importantly, the proposed model can assist the medical sector to assess the organizational performance of hospitals, making it highly applicable for academia and commercial purposes.

2. **Software quality**

Software quality generally refers to the ability of a particular software to comply with consumer demand with respect to function and characteristics (degree of conforming to requirements). Schulmeer and McManus (1996) defined software quality as all functions and characteristics of a software product that satisfy consumer demand. The ISO 9000 standard of
2001 defines software quality as the totality of features and characteristics of a software product that can satisfy stated or implied needs. Rogers (2001) defined software quality as conforming with explicitly stated functional and performance requirements, explicitly documented development standards and implicit characteristics that are expected of all professional developed software. Ward and Venkataraman (1999) suggested that software quality includes the following:

(a) User-based: evaluated by users, software quality refers to the degree of user expectation;

(b) Product deliver-based: evaluated by the product itself, software quality refers to the degree of system effectiveness and program maintainability;

(c) Manufacturing-based: software quality is controlled by the development process, emphasizing quality control and management; and

(d) Organizational control-based: software quality refers to the project costs, production time and resource control and risk management.

Deutsch and Willis (1998) categorized software quality as software procedure quality and software product quality, as illustrated in Figure 1. As a characteristic of software product development process, software procedure quality consists of the following software engineering-related elements: technology, tools, people, organization and equipment. As a characteristic of software product release, software product quality includes document clarity and integrity, design traceability, program reliability and test integrity.

![Figure 1](image-url)

Concept of software quality (Deutsch and Willis (1998))
The ISO 9126-1 standard was developed in 2001 to identify not only the major quality attributes for computer software, but also six major quality attributes as illustrated in Figure 2. This proposal method adopts the ISO 9126-1 model to evaluate the software quality. The evaluation criteria and sub-criteria used to evaluate the software quality are defined as follows:

(1) **Functionality** $(C_1)$: The degree to which the software satisfies stated requirements, including the four sub-criterions of suitability, accuracy, interoperability and security.

   (i) **Suitability**: ability of software to provide an appropriate set of functions for specified tasks and user objectives.
   (ii) **Accuracy**: ability of software to provide correct or agreed results or effects.
   (iii) **Interoperability**: ability of software to interact with one or more specified systems.
   (iv) **Security**: ability of software to prevent prohibited access and withstand deliberate attacks intended to gain unauthorized access to confidential information, or to make unauthorized access.

(2) **Reliability** $(C_2)$: How long the software is available for use; it includes the three sub-criterions of maturity, fault tolerance and recoverability.

   (i) **Maturity**: ability of software to avert failure owing to errors in the software.
   (ii) **Fault tolerance**: ability of software to maintain a specified level of performance in case of software errors or infringement of its specified interface.
   (iii) **Recoverability**: ability of software to re-establish its level of performance and recover the data directly affected if a failure occurs.

(3) **Usability** $(C_3)$: The degree to which software is easily implemented; it includes the four sub-criterions of understandability, learn-ability, operability and attractiveness.

   (i) **Understandability**: ability of software to enable users to understand the appropriateness of a software and its use for particular tasks and conditions of use.
   (ii) **Learning ability**: ability of software to enable users to learn its application.
(iii) **Operability**: ability of software to enable users to operate and control it.

(iv) **Attractiveness**: ability of software to gain user acceptance.

(4) **Efficiency** ($C_4$): How the software optimally uses system resources; it includes the two sub-criterions of time behavior and resource behavior.

(i) **Time behavior**: ability of software to provide appropriate responses, processing times and throughput rates when performing its function under stated conditions.

(ii) **Resource behavior**: ability of software to use appropriate resources in time when the software implements its function under stated conditions.

(5) **Maintainability** ($C_5$): The ease which repairs can be made to the software; it includes the four sub-criterions of analyzability, changeability, stability and testability.

(i) **Analyzeability**: ability of software to be diagnosed for deficiencies or causes of failures in the software, or for the parts to be modified to be identified.

(ii) **Changeability**: ability of software to enable a specified modified to be implemented.

(iii) **Stability**: ability of software to minimize unexpected effects from software modifications.

(iv) **Testability**: ability of software to validate modified software.

(6) **Portability** ($C_6$): How software can be easily transposed from one environment to another; it includes the four sub-criterions of adaptability, installability, co-existence and replaceability.

(i) **Adaptability**: ability of software to be modified for specified environments without applying actions or means other than those provided for the software considered.

(ii) **Install-ability**: ability of software to be installed in a specified environment.

(iii) **Co-existence**: ability of software to co-exist with other independent software in a common environment sharing common resources.

(iv) **Replace-ability**: ability of software to replace other specified software in the environment of that software.
Figure 2
ISO 9126-1 Quality Model (ISO, 2001)
3. AHP methodology

As a decision method that decomposes a complex multi-criteria decision problem into a hierarchy (Saaty (1980)), AHP is also a measurement theory that prioritizes the hierarchy and consistency of judgemental data provided by a group of decision makers. AHP incorporates the evaluations of all decision makers into a final decision, without having to elicit their utility functions on subjective and objective criteria, by pairwise comparisons of the alternatives (Saaty (1990)). Ngai (2003) applied AHP to select web sites. Kima, Yang, Yeo and Kim (2005) adopted AHP to develop a housing performance evaluation model for multi-family residential buildings in Korea. AHP has thus been successfully applied to a diverse array of problems, with the calculation procedure as follows:

Establishment of pair-wise comparison matrix $A$. Let $C_1, C_2, \ldots, C_n$ denote the set of elements, while $a_{ij}$ represents a quantified judgement on a pair of elements $C_i, C_j$. The relative importance of two elements is rated using a scale with the values 1, 3, 5, 7 and 9, where 1 refers to “equally important”, 3 denotes “slightly more important”, 5 equals “strongly more important”, 7 represents “demonstrably more important” and 9 denotes “absolutely more important”. This yields an $n$-by-$n$ matrix $A$ as follows:

$$A = [a_{ij}] = \begin{bmatrix}
C_1 & C_2 & \ldots & C_n \\
1 & a_{12} & \ldots & a_{1n} \\
\frac{1}{a_{12}} & 1 & \ldots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{1}{a_{1n}} & \frac{1}{a_{2n}} & \ldots & 1
\end{bmatrix}. \quad (1)$$

Where $a_{ii} = 1$ and $a_{ij} = \frac{1}{a_{ij}}, i, j = 1, 2, \ldots, n$. In matrix $A$, the problem becomes one of assigning to the $n$ elements $C_1, C_2, \ldots, C_n$ a set of numerical weights $W_1, W_2, \ldots, W_n$ the reflects the recorded judgments. If $A$ is a consistency matrix, the relations between weights $W_i$ and judgments $a_{ij}$ are simply given by $W_j / W_i = a_{ij}$ (for $i, j = 1, 2, \ldots, n$). Saaty (1990) suggested that the largest eigenvalue $\lambda_{\text{max}}$ would be

$$\lambda_{\text{max}} = \sum_{j=1}^{n} a_{ij} \frac{W_j}{W_i}. \quad (2)$$
If \( A \) is a consistency matrix, eigenvector \( X \) can be calculated by
\[
(A - \lambda_{\text{max}} I)X = 0.
\] (3)

Saaty (1990) proposed utilizing consistency index (C.I.) and consistency ratio (C.R.) to verify the consistency of the comparison matrix. C.I. and R.I. are defined as follows:
\[
\text{C.I.} = \frac{\lambda_{\text{max}} - n}{n - 1},
\] (4)
\[
\text{C.R.} = \frac{\text{C.I.}}{\text{R.I.}}.
\] (5)

Where R.I. represents the average consistency index over numerous random entries of same order reciprocal matrices. If C.R \( \leq 0.1 \), the estimate is accepted; otherwise, a new comparison matrix is solicited until C.R \( \leq 0.1 \).

4. Model computation

Step 1. The ultimate goal of evaluating the ideal model can be achieved, followed by six-evaluation criterion, twenty one sub-criteria and finally the alternatives (Figure 2).

Step 2. As shown in Table 1, administer the AHP questionnaire to a sample group, with each respondent making a pair-wise comparison of the decision elements and then assigning them relative scores. Next, the relative scores provided by the decision markers are aggregated via the geometric mean method.

Step 3. Calculate the eigenvalue \( \lambda_{\text{max}} \) and eigenvector \( W \).

Table 1
Aggregate pair-wise comparison matrix for criteria of level 2

<table>
<thead>
<tr>
<th></th>
<th>( C_1 )</th>
<th>( C_2 )</th>
<th>( C_3 )</th>
<th>( C_4 )</th>
<th>( C_5 )</th>
<th>( C_6 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C_1 )</td>
<td>1</td>
<td>( a_{12} )</td>
<td>( a_{13} )</td>
<td>( a_{14} )</td>
<td>( a_{15} )</td>
<td>( a_{16} )</td>
</tr>
<tr>
<td>( C_2 )</td>
<td>( 1/a_{12} )</td>
<td>1</td>
<td>( a_{23} )</td>
<td>( a_{24} )</td>
<td>( a_{25} )</td>
<td>( a_{26} )</td>
</tr>
<tr>
<td>( C_3 )</td>
<td>( 1/a_{13} )</td>
<td>( 1/a_{23} )</td>
<td>1</td>
<td>( a_{34} )</td>
<td>( a_{35} )</td>
<td>( a_{36} )</td>
</tr>
<tr>
<td>( C_4 )</td>
<td>( 1/a_{14} )</td>
<td>( 1/a_{24} )</td>
<td>( 1/a_{34} )</td>
<td>1</td>
<td>( a_{45} )</td>
<td>( a_{46} )</td>
</tr>
<tr>
<td>( C_5 )</td>
<td>( 1/a_{15} )</td>
<td>( 1/a_{25} )</td>
<td>( 1/a_{35} )</td>
<td>( 1/a_{45} )</td>
<td>1</td>
<td>( a_{56} )</td>
</tr>
<tr>
<td>( C_6 )</td>
<td>( 1/a_{16} )</td>
<td>( 1/a_{26} )</td>
<td>( 1/a_{36} )</td>
<td>( 1/a_{46} )</td>
<td>( 1/a_{56} )</td>
<td>1</td>
</tr>
</tbody>
</table>
According to Table 1, the matrix is

\[
\det(A - \lambda I) = \det \begin{bmatrix}
1 - \hat{g} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\
a_{12} & 1 - \hat{g} & a_{23} & a_{24} & a_{25} & a_{26} \\
\frac{1}{a_{13}} & \frac{1}{a_{23}} & 1 - \hat{g} & a_{34} & a_{35} & a_{36} \\
\frac{1}{a_{14}} & \frac{1}{a_{24}} & \frac{1}{a_{34}} & 1 - \hat{g} & a_{45} & a_{46} \\
\frac{1}{a_{15}} & \frac{1}{a_{25}} & \frac{1}{a_{35}} & \frac{1}{a_{45}} & 1 - \hat{g} & a_{56} \\
\frac{1}{a_{16}} & \frac{1}{a_{26}} & \frac{1}{a_{36}} & \frac{1}{a_{46}} & \frac{1}{a_{56}} & 1 - \hat{g}
\end{bmatrix}.
\]  

\[(6)\]

\[
AW_2, \text{ where } I \text{ is unitary matrix and } \hat{\lambda}_{\text{max}} = \hat{g} \text{ substitution formula (2) and (3)}
\]

\[
AW_2 = \begin{bmatrix}
1 - \hat{g} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\
\frac{1}{a_{12}} & 1 - \hat{g} & a_{23} & a_{24} & a_{25} & a_{26} \\
\frac{1}{a_{13}} & \frac{1}{a_{23}} & 1 - \hat{g} & a_{34} & a_{35} & a_{36} \\
\frac{1}{a_{14}} & \frac{1}{a_{24}} & \frac{1}{a_{34}} & 1 - \hat{g} & a_{45} & a_{46} \\
\frac{1}{a_{15}} & \frac{1}{a_{25}} & \frac{1}{a_{35}} & \frac{1}{a_{45}} & 1 - \hat{g} & a_{56} \\
\frac{1}{a_{16}} & \frac{1}{a_{26}} & \frac{1}{a_{36}} & \frac{1}{a_{46}} & \frac{1}{a_{56}} & 1 - \hat{g}
\end{bmatrix} \begin{bmatrix}
W_{21} \\
W_{22} \\
W_{23} \\
W_{24} \\
W_{25} \\
W_{26}
\end{bmatrix} = \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
0 \\
0
\end{bmatrix}.
\]  

\[(7)\]

And we know

\[
W_{21} + W_{22} + W_{23} + W_{24} + W_{25} + W_{26} = 1.
\]  

\[(8)\]

Calculate the formula (7) and (8). The weight of the level 2 is calculated. The same calculations can obtain the weight of each level.

5. Conclusions

This study presents an optimal operating model and algorithm capable of effectively monitoring software in relation to users and purchasers, thus enabling administrators or decision makers to identify the most appropriate software quality. Applying AHP to measure software
product quality by quantifying its attributes can reduce the ambiguity and uncertainty of software quality attributes. Based on measurement results in this study, software users and developers can not only more thoroughly understand the merits and limitations of software products, but also ultimately enhance its overall quality. We recommend that administrators or decision makers adopt the measurement results of this study to evaluate software quality.

References


Received November, 2005