A Model for Measuring Software Understandability

Jin-Cherng Lin, Kuo-Chiang Wu
Dept. of Computer Science & Engineering, Tatung University, Taipei, Taiwan
jclin@ttu.edu.tw, d9206006@ms2.ttu.edu.tw

Abstract

When programmers try to reuse a software system developed by other programmers, the difficulty of understanding the system limits reuses [1]. It is not easy to measure software understandability because understanding is an internal process of humans. This paper proposes "integral of understandability" as a model for measuring software understandability which can extract first the best value of software understandability from factors with higher weight. In other words, we gave an integrated aspect of measuring software understandability through the literature on the subject.

Keywords: *Software Understandability*

1. Introduction

When programmers try to reuse a software system developed by other programmers, the difficulty of understanding the system limits reuses [1]. It is not easy to measure software understandability because understanding is an internal process of humans. In order to measure understandability, we need to observe an external process which externalizes understanding, and therefore Boehm defined software understandability as a feature of software quality which means ease of understanding software systems [2].

In Figure 1, the programmers of the original system were in the same organization, they may be transferred, or may retire or change jobs. In software developing, it is necessary for enhancing functions, correcting faults, or adapting them to new circumstances. If the programmers of the original system were absent, the other programmers reusing it need to understand it. If it is difficult to understand, changes to it may cause serious faults and a chain reaction of changes. Such changes may cost more time than remaking the software system [3].

Figure 1 shows communications among humans, software, and hardware in software evolution. Programmer 1 writes version 1 of a software system. Programmer 2 evolves the software system from version 1 into version 2. Software can be considered as the media of communications from the programmers to the computer. Programmer 2 reads version 1 in order to write version 2. Therefore, software can be considered as the media of communications from programmer 1 to programmer 2. Understanding a software system can be considered to be "reading necessary information via the software system to evolve it". Programmer 2 may feel that he/she understood version 1 when he/she finished reading it. However, programmer 2 might misunderstand version 1 and thus introduce faults into version 2. Therefore, we usually consider that programmer 2 understands version 1 when he/she could correctly write version 2. In order to measure understandability, we need to observe an external process (like writing the version 2) which externalize understanding [3].

This paper proposes "integral of understandability" as a model for software understandability. Integral of understandability is to extract the highest factor from the understandability, and then add the secondary factor of understandability gradually basing on weight of highest factor of understandability, how much understandability may be promoted after adding, it can be calculated after weighing fuzzy weight λ, finally, when the

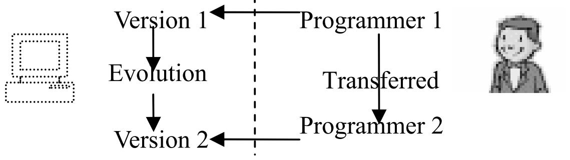


Figure 1 Programmer 2 might misunderstand version 1 and thus introduce faults into version 2.

accumulated importance or weight compromises with the understandability, it will be the result of synthesis metrics that software takes all interacted factors into consideration. Such "result of synthesis metrics" is the understandability of whole software. It is worth noticing that the model for integral of understandability can get rid of repeated and secondary information so as to leave the best result of synthesis metrics.

2. Factors Affecting Software Understandability

We reviewed factors suggested by literatures to affect understandability of software. Furthermore, we defined the membership function in fuzzy mathematics for different factors.

2.1 Understandability of Documentation

The literature [4] provides the following opinions: Software is maintained through the integrated use of source code and documents. Source code readability and quality of documentation should be taken into account while measuring the software maintainability. Comments Ratio (CR) is used to judge the Readability of Source Code (RSC). Quality of Documentation (QOD) is judged using Fog index.

In literature [4], understandability of software documentation we compiled is composed of source code (RSC) and documents (QOD) with membership function as follows:

(1) Membership functions for documents judged using Fog index:

$$\mu_{poor}\left(x\right)=\begin{cases}1 & x\le 9\\ 0.5x-6 & 10<x\le 12\\ 0 & x>12\end{cases}$$

Note: $\mu_{poor}\left(x\right)$ need being modified in literature [4]

$$\mu_{avg}(x)=\begin{cases}0 & x\le 10 \text{ or } x>18\\ 0.5x-5 & 10<x\le 12\\ 1 & 12<x\le 16\\ 0.5x-9 & 16<x\le 18\end{cases}$$

$$\mu_{good}(x)=\begin{cases}0 & x\le 16\\ 0.5x-8 & 16<x\le 18\\ 1 & x>18\end{cases}$$

(2) Membership functions for RSC judged using CR:

$$\mu_{poor}(x)=\begin{cases}0 & x\le 7\\ x-7 & 7<x\le 8\\ 1 & x>8\end{cases}$$

$$\mu_{avg}(x)=\begin{cases}0 & x<5 \text{ or } 8<x\\ x-5 & 5<x\le 6\\ 1 & 6<x\le 7\\ -x+8 & 7<x\le 8\end{cases}$$

$$\mu_{good}(x)=\begin{cases}1 & x\le 4\\ -0.5x+3 & 4<x\le 6\\ 0 & x>6\end{cases}$$

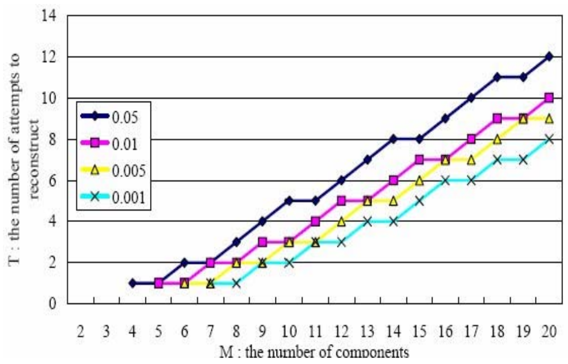


Figure 2 The maximum number of attempts to reconstruct with significance [3]

2.2 Understandability of Structure

When a worker needed to reconstruct one software system many times until he/she correctly reconstructs it, it can be considered the software system is difficult for him/her to understand. Needless to say, understanding depends on not only understandability of the software system, but also comprehension of the worker. If many workers overhauled many software systems, the average number of attempts needed for correct reconstruct can be a metric of understandability or comprehension. The average number of attempts needed for correct reconstruction that one worker reconstructed many software systems means comprehension of the worker. The average number of attempts needed for correct reconstruction that many workers reconstructed one software system means understandability of the software system [3].

The experimental results of Figure 2 [3] show that the number of attempts to reconstruct and the number of components are correlated, so, the number of components can quantify understandability of software structure. Basing on literature [3], we can design membership function of understandability of software structure as follows:

$$\mu_{poor}\left(x\right)=\begin{cases}0 & x\le 75\\ 0.1x-7.5 & 75<x\le 85\\ 1 & x>85\end{cases}$$

$$\mu_{avg}(x)=\begin{cases}0 & x\le 45 \text{ or } x>85\\ 0.1x-4.5 & 45<x\le 55\\ 1 & 55<x\le 75\\ -0.1x+8.5 & 75<x\le 85\end{cases}$$

$$\mu_{good}(x)=\begin{cases}1 & x\le 45\\ -0.1x+5.5 & 45<x\le 55\\ 0 & x>55\end{cases}$$

2.3 Understandability of Components

The literature [5] has developed the cognitive functional size (CFS) on the basis of cognitive weights, which determines software complexity from both architectural and cognitive aspects. The CFS provides a foundation for cross-platform analysis of complexities, sizes, and comprehension effort of software specifications and implementations in the phases of design, implementation, or maintenance in software engineering. The CFS has been shown as a fundamental measure of software artifacts based on the cognitive weights [5]. We adopted CFS to measure understandability of software components. Basing on literature [5], we can design membership function of understandability of software structure as follows:

$$\mu_{poor}(x)=\begin{cases}0 & x\le 200\\ 0.04x-8 & 200<x\le 225\\ 1 & x>225\end{cases}$$

$$\mu_{avg}(x)=\begin{cases}0 & x<100 \text{ or } x>225\\ 0.04x-4 & 100<x\le 125\\ 1 & 125<x\le 200\\ -0.04x+9 & 200<x\le 225\end{cases}$$

$$\mu_{good}(x)=\begin{cases}1 & x\le 100\\ -0.04x+5 & 100<x\le 125\\ 0 & x>125\end{cases}$$

It is noticeable that above described membership function indicates single software component if the complete membership function of software can be calculated with soft computing or fuzzy statistics theory (see section 3).

2.4 Understandability of Source Code

Understandability of code can use Halstead complexity metrics to evaluate understandability of code of software [6]. The formula of Halstead complexity is as follows:

Halstead complexity V (Effort of program)

$H = n_1 \times \log_{n_1} + n_2 \times \log_{n_2}$

N= N1 + N2

n= n1 + n2

$V = N \times \log(n_1 + n_2)$

Where:

V = Volume

N : Program length

n : Program vocabulary

n_1 : the number of distinct operators

n_2 : the number of distinct operands

N_1 : the total number of operators

N_2 : the total number of operands

Basing on literature [6][7] and formula, we can design membership function of understandability of software structure as follows:

$$\mu_{poor}(x)=\begin{cases}0 & x\le 0.6\\ 5x-3 & 0.6<x\le 0.8\\ 1 & x>0.8\end{cases}$$

$$\mu_{avg}(x)=\begin{cases}0 & x<0.2 \text{ or } 0.8<x\\ 5x-1 & 0.2<x\le 0.4\\ 1 & 0.4<x\le 0.6\\ -5x+4 & 0.6<x\le 0.8\end{cases}$$

$$\mu_{good}(x)=\begin{cases}1 & x\le 0.2\\ -5x+2 & 0.2<x\le 0.4\\ 0 & x>0.4\end{cases}$$

2.5 Understandability of Data

Literature [8] points out that code and data spatial complexity can measure understandability of software. Data spatial complexity (DMSC) relates to measure of whole program. Fuzzy membership function calculated by the analyzed data according to literature [8] basing on understandability of data of DMSC is as follows:

$$\mu_{poor}(x)=\begin{cases}0 & x\le 0.4\\ 10x-4 & 0.4<x\le 0.5\\ 1 & x>0.5\end{cases}$$

$$\mu_{avg}(x)=\begin{cases}0 & x<0.2 \text{ or } x>0.5\\ 10x-2 & 0.2<x\le 0.3\\ 1 & 0.3<x\le 0.4\\ -10x+5 & 0.4<x\le 0.5\end{cases}$$

$$\mu_{good}(x)=\begin{cases}1 & x\le 0.2\\ -10x+5 & 0.2<x\le 0.3\\ 0 & x>0.3\end{cases}$$

3. Fuzzy Maximum Membership Estimation for Population

To estimate the maximum membership value of population, we suggest the following methods.

Fuzzy maximum membership estimation for population is

$$\widetilde{F}_{\mu_{MM}} = \frac{\frac{S_1}{\sum S_j}}{L_1} + \frac{\frac{S_2}{\sum S_j}}{L_2} + \cdots + \frac{\frac{S_k}{\sum S_j}}{L_k}$$

$$S_j = \frac{1}{n}\sum_{i=1}^{n} m_{ij} \bullet I_{ij},$$

where $I_{ij} = \begin{Bmatrix} 1, if\ m_{ij} = Max(m_{i1,\dots,}m_{ik}) \\ 0, if\ m_{ij} \ne Max(m_{i1,\dots,}m_{ik}) \end{Bmatrix}$ (see Table 1)

Linguist Variables \ Sample	Linguist Variables L_1	Linguist Variables L_2	… …	Linguist Variables k L_k
Sample 1	m_{11}	m_{12}	…	m_{1k}
Sample 2	m_{22}	m_{22}	…	m_{2k}
Sample 3	m_{31}	m_{32}	…	m_{3k}
:			…	
:			…	
Sample x	m_{x1}	m_{x2}	…	m_{xk}
:			…	
Sample n	m_{n1}	m_{n1}	…	m_{nk}
Fuzzy sample mean	$\frac{1}{n}\sum_{i=1}^{N} m_{i1}$	$\frac{1}{n}\sum_{i=1}^{N} m_{i2}$	…	$\frac{1}{n}\sum_{i=1}^{N} m_{ik}$

Table 1 Fuzzy maximum membership estimation

We will explain how to apply the estimation method in the next section.

4. A Model for Integral of Understandability

4.1 Calculate Fuzzy Maximum Membership Estimation of Sampling Data

In Table 2, samples x can be fog index, comment ration, the number of components, CFS, Halstead Complexity, or DMSP. Calculate respectively fuzzy maximum membership estimation for population of fog index, comment ration, the number of components, CFS, Halstead Complexity, or DMSP with method in section 3. For example, it is known that Table 2 is the sampling value of Fog index of software (for example, one software has 5 pieces of main components), understandability of documentation in software can be obtained from the sampling value of Fog index calculated with fuzzy maximum membership estimation for population. Meanwhile, understandability of software can be calculated with the sampling values of comment ration, the number of components, CFS, Halstead Complexity, or DMSP (see Table 3).

Linguist Variables \ Components	Linguist Poor(Low)	Linguist Avg(Med)	Linguist Good(High)
Component 1	0	0.2	0.8
Component 2	0.3	0.7	0
Component 3	0	0.3	0.7
Component 4	0	0.3	0.7
Component 5	0	0.4	0.6
S_j	0	0.14	0.56
$S_j/\sum S_j$	**0**	**0.2**	**0.8**

Table 2 Fuzzy Maximum Membership Estimation for Population for Fog Index

4.2 Example of a Model for Integral of Understandability

We will establish a model of integral of understandability to calculated understandability of software. For the convenience to explain, we will draw model of integral of understandability into Table 3, wherein ①②③ represent the setting steps which are described respectively in the following:

① After calculating Table 3, converts it into understandability matrix:

$$\begin{bmatrix} 0 & 0.2 & 0.8 \\ 0 & 1 & 0 \\ 0.5 & 0.5 & 0 \\ 0 & 0.2 & 0.8 \\ 0.4 & 0.6 & 0 \\ 0 & 0.5 & 0.5 \end{bmatrix}$$

② Row weights vector can be obtained through PCA and Factor Analysis (see [12]), we calculated that row weights vector are (0.21,0.16,0.12,0.16,0.15) with PCA and factor analysis.

③ Column weights vector can be obtained through Fuzzy AHP, we calculated that column weights vector are (0.29, 0.46, 0.25) with Fuzzy AHP

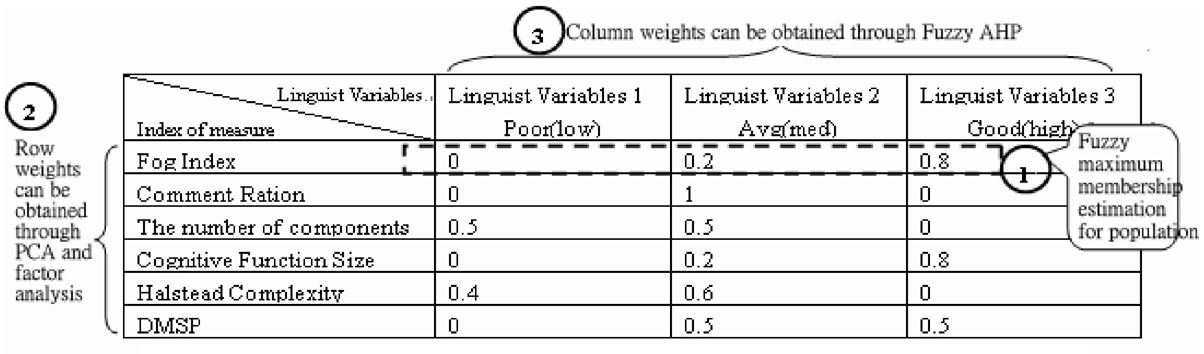
Synthesis vector of understandability can be obtained through Fuzzy Inner Product.

Synthesis Vector of Understandability =

$$\underset{\text{Row Weights Vector}}{(0.21, 0.16, 0.12, 0.16, 0.15, 0.2)} \circ \overset{\text{Understandability Matrix}}{\begin{bmatrix} 0 & 0.2 & 0.8 \\ 0 & 1 & 0 \\ 0.5 & 0.5 & 0 \\ 0 & 0.2 & 0.8 \\ 0.4 & 0.6 & 0 \\ 0 & 0.5 & 0.5 \end{bmatrix}}$$

$$= \left[0.4, 0.2, 0.21\right] = h(x_1, x_2, x_3)$$

Using definition of Fuzzy Measure in [9][10][11]


3
Column weights can be obtained through Fuzzy AHP
2
Row weights can be obtained through PCA and factor analysis
1
Fuzzy maximum membership estimation for population

Index of measure \ Linguist Variables	Linguist Variables 1 Poor(low)	Linguist Variables 2 Avg(med)	Linguist Variables 3 Good(high)
Fog Index	0	0.2	0.8
Comment Ration	0	1	0
The number of components	0.5	0.5	0
Cognitive Function Size	0	0.2	0.8
Halstead Complexity	0.4	0.6	0
DMSP	0	0.5	0.5

Table 3 Example of Different Fuzzy Maximum Membership Estimation for Population

obtained the value of g. It can be calculated: $g(\{x_1\})=0.29$, $g(\{x_2\})=0.46$, $g(\{x_3\})=0.25$, $g(\{x_1,x_2\})=0.71$, $g(\{x_1,x_3\})=0.57$, $g(\{x_2,x_3\})=0.78$, $g(\{x_1,x_2,x_3\})=1$

At last, calculate understandability of complete software with Fuzzy integral [9][10][11], the calculating process is as follows:

$$\int h\, dg = [h(\{x_1\}) \wedge g(\{x_1\})] \vee [h(\{x_3\}) \wedge g(\{x_1, x_3\})]$$

$$\vee [h(\{x_2\}) \wedge g(\{x_1, x_2, x_3\})] \text{ (refer to[9][10][11])}$$

$$= [0.4 \wedge 0.29] \vee [0.21 \wedge 0.57] \vee [0.2 \wedge 1] = 0.29$$

Meanwhile, understandability of software in different versions can also be calculated (see Table 4). In Table 4, version 4 in integral of understandability is the maximum indicating understandability of version 4 is the best. If take version as time axis, the tendency chart of understandability of software (see Figure 3) that shows changes of software along with time can be drawn, it can track and compare difference of different versions of software understandability.

	Synthesis Vector of Understandability			Integral of Understandability
Version 1	0.4	0.2	0.21	0.29
Version 2	0.33	0.23	0.17	0.29
Version 3	0.42	0.18	0.23	0.29
Version 4	0.35	0.37	0.17	0.37

Table 4 Software Understandability of Different Versions

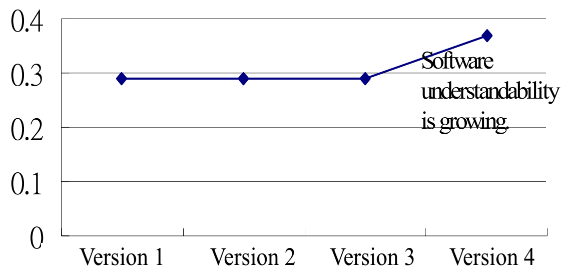


Figure 3 Difference of Understandability of Software of Different Versions

4.3 A Model for integral of understandability

Basing on above examples, we are trying to establish a model for integral of understandability, its steps are as follows:

(1) To collect the original data of each unit according to fog index, comment ration, the number of components, CFS, Halstead Complexity, or DMSP (see section 2). (Note: one unit can be one component or user can decide its scope)

(2) Take the data collected in step 1 as sample, and then calculate fuzzy maximum membership estimation (see Table 3) with fuzzy maximum membership estimation for population.

(3) Set up understandability matrix.

(4) Calculate Row Weights Vector with PCA and factor analysis in [12].

(5) Calculate Column Weights Vector with Fuzzy AHP in [13][14][15][16]

(6) Synthesis vector of understandability can be obtained through Fuzzy Inner Product.

Synthesis Vector of understandability = Row Weights Vector • Understandability Matrix (Fuzzy Inner Product)

(7) Use definition of Fuzzy Measure in [9][10][11] to obtain the value of g.

(8) Calculate complete understandability of software with Fuzzy integral (refer to [9][10][11]).

(9) Calculate understandability of software (see Table 4) of different versions.

(10) If take version as time axis, the tendency chart of understandability of software that shows changes of software along with time can be drawn (see Figure 3).

5. Conclusions

Many researches (see [3][4][5][7][8]) have focused on the issue of understandability of software and of maintenance difficulty. This paper presents a unified model of the synthesis metrics for the understandability of software that has integrated the metrics of understandability of software concluded by past researchers into a unified model of the synthesis metrics for understandability of software, which can completely estimate understandability of software but not the specific factor.

In future the researchers can still apply the model this paper prepared when putting forward new measurement of software understandability. Because the integral of understandability can extract first the best value of understandability of software from factors with higher weight. In addition, this paper has put forward Fuzzy maximum membership estimation for population to calculate Fuzzy maximum membership estimation of understandability in a component.

References

[1] Caldiera, G. and Basili,V R , The qualification of reusable software components, in Software Reusability (Schafer, W., Prieto-D'iaz, R. and Matsumoto, M., eds.), Ellis Horwood. 1994, pp. 117-119.
[2] Boehm, B.W., et al, , Characteristics of Software Quality, North-Holland, 1978.
[3] K. Shima,Y. Takemura,K. Matsumoto, An Approach to Experimental Evaluation of Software Understandability, Proceedings of the 2002 International Symposium on Empirical Software Engineering (ISESE'02)
[4] Aggarwal KK, Singh Y., and Chhabra J K , An Integrated Measure of Software Maintainability, Proceedings of Reliability and Maintainability, 2002: 235-241
[5] Shao, J. and Y. Wang, A New Measure of Software Complexity Based on Cognitive Weights , Canadian Conference on Electrical and Computer Engineering, 2003: 1333-1338
[6] M. H. Halstead, Elements of Software Science, Amsterdam: Elsevier North Holland, 1977.
[7] Jin, He, Qin, Su, Jie, Gao, Structure of Software Understandability and Its Measurement by Using Fuzzy Logic, Computer Engineering, November 2005,Vol.31 No.21
[8] Chhabra , J.K., Aggarwal, K.K., and Singh, Y, Code and Data Spatial Complexity: Two Important Software Understandability Measures, Information and Software Technology, 2003, 45(8), pp. 539-546
[9] Ralescu, D. A., and Adams, G., Fuzzy integral Journal of Mathematical Analysis and Applications, 1980, Vol.75, No.2, pp.562-570.
[10] Chen, Ting-Yu, Wang, Jih-Chang, and Tzeng, Gwo-Hshiung, Identification of General Fuzzy Measures by Genetic Algorithms Based on Partial Information, IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics, Vol. 30B, No. 4, pp. 517-528 2000.
[11] Lee, K. M., and LeeKwang, H., Identification of λ-fuzzy measure by genetic algorithm, Fuzzy Sets and Systems, Vol. 75, No. 3, pp.301-309, 1995.
[12] Johnson, R.A. and Wichern, D.W., Applied multivariate statistical analysis 5th edition, 2002, p426-475
[13] Buckly, J.J., Fuzzy hierarchy analysis, Fuzzy Sets and Systems, Vol.17, No.3, 1985, pp.233-247.
[14] Saaty, TL, Analytic hierarchy process, Prefare, Wiley, N.Y., 1980, pp.18-21, pp.50-57, pp61, pp.223-225
[15] Chang , D.Y. , Application of Extent Analysis Method on Fuzzy, European Journal Operational Research, The Netherlands, 1996, 95(3):pp.649-655.
[16] Chang , D.Y., Extent Analysis and Synthetic Decision, Optimization Techniques and Application, ICOTA'92, World Scientific, Singapore, 1992,1(5) p.352.