International Journal of u- and e- Service, Science and Technology
Vol.8, No. 6 (2015), pp.105-116
http://dx.doi.org/10.14257/ijunesst.2015.8.6.11

On the Application of Information Entropy-based Multi-attribute Decision in UML Class Diagram Metrics

Tong Yi

School of Information Technology,
Jiangxi University of Finance and Economics, Nanchang, China
tongyi@jxufe.edu.cn

Abstract

The research, development and applications of software measurement have been carried out for more than forty years. Many researchers have done much in it, obtained lots of theoretical results, and developed a series of practical applications. At present, with the rapid development of object-oriented technology used in software theories and application, how to measure software in an effectively and scientifically way is becoming a hot and difficult issue. Class diagram as the most important UML model, its reasonable design has a remarkable effect on the ultimate system. This paper applies information entropy-based multi-attribute decision in UML class diagram metrics, providing a method to measure UML class diagram complexity weights. Its attributes out of information entropy and attributes' weigh vectors can measure the complexity of object-oriented software effectively and scientifically. Besides, case analysis can prove this metric's usability. The more precise experiment outcome has proved that the method is connected to human's experience, and also can be applied to improve software quality.

Keywords: *information entropy, software measurement, class diagram, unified modeling language, Multi-attribute Decision-making*

1. Introduction

Since R. J. Rubey and R. D. Hartwick first proposed the concept of software measurement in 1968[1], the research, development and applications of it have been carried out for more than forty years. Many researchers have done much in it [2-7], obtained lots of theoretical results, and developed a series of practical applications. At present, with the rapid development of object-oriented technology in the field of software theory and applications, how to scientifically and effectively measure the complexity of object-oriented software, which is the key problem to be solved at present. Unified Modeling Language (UML) is a popular and visual standard modeling language, which captures relevant decision and understanding of systems under development and is suitable for different stages of software life cycle. It has got extensive support from the industry field [8]. Because it merged a lot of methods and technologies, it is too redundant and complex to be comprehended [9]. Therefore how to comprehend, analyze, test and maintain systems modeled by UML has become the key to correctly using UML. Class diagram, one of the most important diagrams, shows a set of classes, interfaces, their internal structure and their relationships such as dependence, association, generalization, aggregation, composition, realization, etc. So analysis of the class diagram is the core of this problem, and the complexity of class diagrams directly affects the complexity of the object-oriented software, thus the study of class complexity research work has essentially taken the place of object-oriented software complexity metrics.

During the last several years, many different metrics for class diagrams have been suggested [10-27]. These metrics help software developers to analyze reliability, maintainability and complexity of systems in the early phase of the OO software lifecycle.

ISSN: 2005-4246 IJUNESST

Early metrics on object-oriented software quality set were completely or partially based on source code, which belongs to the MOOSE measurement and MOOD measurement of the maximum. M. Marehesi [10] research work mainly focused on the measurement of use case diagram and class diagram, and proposes a set of indicators to measure the complexity of class diagrams. They are the total number of classes (OA1), the total number of inheritance hierarchies (OA2), the average weighted responsibilities of classes (OA3), standard deviation of the weighted number of responsibilities of classes (OA4), the average number of direct dependencies of classes (OA5), standard deviation of the number of direct dependencies (OA6), and percentage of inherited responsibilities with respect to their total number (OA7). Obviously, OA1 and OA2 are used to evaluate the complexity of class diagram; OA3 and OA4 to the number of class responsibilities; OA5 and OA6 to the number of direct dependencies among classes; and OA7 to inherited responsibilities in class diagrams.

M. Genero also discusses a group of indicators to measure the complexity of class diagrams [11-20]. They are the respective total number of classes (NC), attributes (NA), methods (NM), associations relationships (NAssoc), aggregation relationships (NAgg), dependency relationships (NDep), generalization relationships (NGen), generalization hierarchies (NGenH), and aggregation hierarchies (NAggH); respective percentage of the NAssoc, NAgg, NDep, and NGen with respect to the NC whose results can be abbreviated as NAssocVC, NAggVC, NDepVC, and NgenVC respectively; the respective maximum DIT and Hagg value whose results can be abbreviated as MaxDIT and MaxHagg respectively. The DIT value for a class within a generalization hierarchy is the longest path from the class to the root of the hierarchy while the Hagg value for a class within an aggregation hierarchy is the longest path from the class to the leaves.

To analyze architecture complexity, P. In [21], a researcher, uses metric tree to help a project manager early in the development lifecycle.

Apart from above works, R. Rufai discusses different similarity indicators for assessing the similarity between a pair of UML models based on information gleaned from their class diagrams [22]. The first approach is to use the semantic distance measure of the terms that appear in the model such as class names, attribute names, method names in a class model. For this approach, he devises two metrics, namely shallow semantic similarity metric (SSSM) and deep semantic similarity metric (DSSM). The former concerns the names of the classes in the models to be compared to compute their similarity, while the latter the names of attributes and methods instead. The second approach is based on comparing the signatures of the classes involved (signature-based similarity metric—SBSM). Yet the third approach is to use the relationships among the classes of a class model as the criteria for the comparison of the models to be compared (relationships-based similarity metric—RBSM).

Dr. Z. Chen [23] of Southeast University for the cohesion and coupling of the classes, used the system theory point of view to measure the class diagram.

Dr. Y. Zhou [24] put UML class diagram into weighted class dependence graph, using information entropy method to measure the UML class diagram, and has obtained the very good measurement effect. The metric first defines weights for various relationships respectively. Then it gives some rules to transform a class diagram into a weighted class dependence graph. For any two different class diagrams, because the relationships among class (or template) are different, the corresponding weighted class dependence graphs might not be the same. Therefore, the occurrence of incoming or outgoing edges of nodes in a weighted class dependence graph is random. Finally the structure complexity of a class diagram is defined as the entropy distance of the corresponding weighted class dependence graph. To evaluate the structure complexity of a class diagram, the conditional entropy $H(X|Y)$ and the mutual information $I(X,Y)$ of X and Y must be computed first, where X and Y are two classes.

Dr. T. Yi [25-27] improved Dr. Y. Zhou method. He taked the relationships between classes, attributes and methods within classes together, and put forward a method of UML class diagrams complexity metric based on dependence analysis.

At the same time, after nearly 20 years' development, multi-attribute decision theories and methods has become a hotspot in the field of decision science, system engineering, management operations, and so on. In the management of multiple attribute decision making problems, people tend to be more indicators as evaluation criteria for selection of alternatives. Because the importance of every index is different, the weights are different. Generally, there are two kinds of ways: subjective weighting method and objective weighting method [28-29]. Subjective weighting method is that experts base on subjective preference of each attribute to determine weights. Objective weighting method is using objective information given by attributes to determine weights. In 1948, American mathematician Shannon put forward the information entropy as an important measure to describe things uncertain [30], which has been widely used in Information Science, decision science and related fields in recent years. The concept of information gain based on information entropy is use as the attribute selection's metric [31], which size reflect the importance of each attribute. Putting it as a method of constructing the weights of attributes not only reduced subjectivity, but also can reflect the internal structure of the real information system, which helps people understand things from nature.

In this paper, the research work mentioned above is a part of the existing domestic and international research work, but there is no doubt that the results of the research in the UML class diagram model is too little. One of the important reasons is that UML standard released by the object management group OMG only gives the semantic concept level description on various modeling elements, which leads to weight indicators to measure model often differed from man to man.

This paper based on information entropy-based multi-attribute decision, adopts a new theoretical perspective, providing a metric of UML class diagram complexity weights putting the UML class diagram modeling elements into decision matrix; standardization computing can turn it into column normalized matrix; attribute output by information entropy and the weight vectors of attributes can effectively measure the complexity of object-oriented software measurement. At last, in order to prove the effectiveness and usability, this paper compares its method to the method by Dr. Y. Zhou [24] and the method by Dr. T. Yi [25-27], which has proved that the method is connected to human's experience.

2. A method of measuring UML Class Diagram Complexity Weights

2.1. Measuring Complexity of UML Class Diagrams

In information theory, entropy can be used to express the uncertainty of things. This paper thinks that the theory will help greatly to solve problem that the calculation of the previous modeling elements in UML class diagrams cannot be quantitative, namely the object management group OMG released UML standard only giving the semantic concept level description on various modeling elements. Applying information entropy-based multi-attribute decision to UML class diagram metric, the specific steps are as follows [32].

1) Analysing the UML class diagrams, and constructing decision matrix A.

$$A=(a_{ij})_{n\times m} \tag{1}$$

2) Considering the attribute type belongs to benefit attributes, use formula (2) to normalize it, which can get matrix $R=(r_{ij})_{n\times m}$.

$$r_{ij} = \frac{a_{ij}}{\max_i(a_{ij})} \quad , i\in N, j\in I_1 \tag{2}$$

3) From formula (3), after calculate matrix $R=(r_{ij})_{n\times m}$, get Column normalized matrix $\dot{R}=(\dot{r}_{ij})_{n\times m}$.

$$\dot{r}_{ij}=\frac{r_{ij}}{\sum_{i=1}^{n} r_{ij}}, \ i\in N, j\in M \tag{3}$$

4) From formula (4), get the information entropy of attribute u_j .

$$E_j=-\frac{1}{\ln n}\sum_{i=1}^{n}\dot{r}_{ij}\ln\dot{r}_{ij}, j\in M \tag{4}$$

When $\dot{r}_{ij}=0$, regulate $\dot{r}_{ij}\ln\dot{r}_{ij}=0$.

5) From formula (5), get the attribute weights vector $\omega=(\omega_1, \omega_2,\ldots, \omega_n)$.

$$\omega_j=\frac{1-E_j}{\sum_{k=1}^{m}(1-E_k)} \tag{5}$$

6) From formula (6), get each class diagram's Comprehensive attribute value $z_i(\omega)$.

$$z_i(\omega)=\sum_{j=1}^{m} r_{ij}\omega_j, \ i\in N \tag{6}$$

2.2. Case Study

To validate the metric proposed in this paper, we will carry out an experiment to estimate the metric in the following. On the condition of getting permission from M. Genero, we also use the same former twenty-seven UML class diagrams related to bank information systems as material [11-20]. M. Genero, et al. carry out some comprehensive controlled experiments [11-20].

How to compute the complexity of UML class diagrams?

1) For the multi-attribute decision problem, construct decision matrix A, as shown in Table 1. For better representation, dependency is represented by u_1, normal association is represented by u_2, aggregation is represented by u_3, generalization is represented by u_4, class method is represented by u_5, class attribute is represented by u_6, the number of classes is represented by u_7.

Table 1. Decision Matrix A

class diagram	u_1	u_2	u_3	u_4	u_5	u_6	u_7
1	0	1	0	0	8	4	2
2	0	1	1	0	12	6	3
3	0	1	2	0	15	9	4
4	0	3	0	0	12	7	3
5	0	1	3	0	21	14	5
6	0	2	0	0	12	6	3
7	1	3	0	0	13	8	4
8	0	2	2	2	14	10	6
9	1	1	0	0	12	9	3
10	0	2	3	2	22	14	7
11	0	2	3	4	30	18	9
12	0	3	3	2	39	19	7
13	1	3	2	2	35	22	8

14	0	0	0	4	30	11	5
15	0	0	0	10	30	12	8
16	0	0	0	18	38	17	11
17	2	11	6	10	76	42	20
18	1	11	6	16	88	41	23
19	1	7	6	20	94	45	21
20	3	13	7	24	98	56	33
21	0	1	5	2	47	28	9
22	0	3	5	20	65	31	18
23	0	11	6	21	79	44	26
24	0	1	5	19	69	32	17
25	2	10	7	11	75	51	23
26	4	14	4	16	84	42	22
27	0	5	9	7	77	34	14

2) Use formula (2) to normalize A, get the following matrix R, as shown in Table 2.

Table 2. Matrix R

class diagram	u_1	u_2	u_3	u_4	u_5	u_6	u_7
1	0	0.071	0	0	0.0816	0.0714	0.0606
2	0	0.071	0.111	0	0.1224	0.1071	0.0909
3	0	0.071	0.222	0	0.1531	0.1607	0.1212
4	0	0.214	0	0	0.1224	0.125	0.0909
5	0	0.071	0.333	0	0.2143	0.25	0.1515
6	0	0.143	0	0	0.1224	0.1071	0.0909
7	0.25	0.214	0	0	0.1327	0.1429	0.1212
8	0	0.143	0.222	0.083	0.1429	0.1786	0.1818
9	0.25	0.071	0	0	0.1224	0.1607	0.0909
10	0	0.143	0.333	0.083	0.2245	0.25	0.2121
11	0	0.143	0.333	0.167	0.3061	0.3214	0.2727
12	0	0.214	0.333	0.083	0.398	0.3393	0.2121
13	0.25	0.214	0.222	0.083	0.3571	0.3929	0.2424
14	0	0	0	0.167	0.3061	0.1964	0.1515
15	0	0	0	0.417	0.3061	0.2143	0.2424
16	0	0	0	0.75	0.3878	0.3036	0.3333
17	0.5	0.786	0.667	0.417	0.7755	0.75	0.6061
18	0.25	0.786	0.667	0.667	0.898	0.7321	0.697
19	0.25	0.5	0.667	0.833	0.9592	0.8036	0.6364

20	0.75	0.929	0.778	1	1	1	1
21	0	0.071	0.556	0.083	0.4796	0.5	0.2727
22	0	0.214	0.556	0.833	0.6633	0.5536	0.5455
23	0	0.786	0.667	0.875	0.8061	0.7857	0.7879
24	0	0.071	0.556	0.792	0.7041	0.5714	0.5152
25	0.5	0.714	0.778	0.458	0.7653	0.9107	0.697
26	1	1	0.444	0.667	0.8571	0.75	0.6667
27	0	0.357	1	0.292	0.7857	0.6071	0.4242

3) From formula (3), calculate matrix R, can get Column normalized matrix $\dot{R}$, as shown in Table 3.

Table 3. Matrix $\dot{R}$

class diagram	u_1	u_2	u_3	u_4	u_5	u_6	u_7
1	0	0.0089	0	0	0.0067	0.006	0.006
2	0	0.0089	0.01176471	0	0.01	0.009	0.01
3	0	0.0089	0.02352941	0	0.0126	0.014	0.013
4	0	0.0268	0	0	0.01	0.011	0.01
5	0	0.0089	0.03529412	0	0.0176	0.022	0.016
6	0	0.0179	0	0	0.01	0.009	0.01
7	0.063	0.0268	0	0	0.0109	0.013	0.013
8	0	0.0179	0.02352941	0.01	0.0117	0.016	0.019
9	0.063	0.0089	0	0	0.01	0.014	0.01
10	0	0.0179	0.03529412	0.01	0.0184	0.022	0.022
11	0	0.0179	0.03529412	0.019	0.0251	0.028	0.029
12	0	0.0268	0.03529412	0.01	0.0326	0.03	0.022
13	0.063	0.0268	0.02352941	0.01	0.0293	0.035	0.025
14	0	0	0	0.019	0.0251	0.017	0.016
15	0	0	0	0.048	0.0251	0.019	0.025
16	0	0	0	0.086	0.0318	0.027	0.035
17	0.125	0.0982	0.07058824	0.048	0.0636	0.066	0.064
18	0.063	0.0982	0.07058824	0.076	0.0736	0.065	0.073
19	0.063	0.0625	0.07058824	0.095	0.0787	0.071	0.067
20	0.188	0.1161	0.08235294	0.114	0.082	0.089	0.105
21	0	0.0089	0.05882353	0.01	0.0393	0.044	0.029
22	0	0.0268	0.05882353	0.095	0.0544	0.049	0.057
23	0	0.0982	0.07058824	0.1	0.0661	0.07	0.083
24	0	0.0089	0.05882353	0.09	0.0577	0.051	0.054

25	0.125	0.0893	0.08235294	0.052	0.0628	0.081	0.073
26	0.25	0.125	0.04705882	0.076	0.0703	0.066	0.07
27	0	0.0446	0.10588235	0.033	0.0644	0.054	0.045

4) From formula (4), can get the information entropy E_j of attribute u_j, then E_1=0.6210,E_2=0.8461,E_3=0.8588,E_4=0.6813,E_5=0.7721,E_6=0.4177,E_7=0.8035.
5) From formula (5), can get
ω_1=0.1895,ω_2=0.077,ω_3=0.0706,ω_4=0.1594,ω_5=0.114,ω_6=0.2912,ω_7=0.0983.
Then, the attribute weights vector ω is
ω=(0.1895, 0.077, 0.0706, 0.1594, 0.114, 0.2912, 0.0983).
6) Use formula (6) to calculate each class diagram's Comprehensive attribute value $z_i(\omega)$, then the results shown in Table 4.

Table 4. Comprehensive Attribute Value

class diagram	$z_i(\omega)$	class diagram	$z_i(\omega)$	class diagram	$z_i(\omega)$
1	0.042	10	0.167	19	0.672
2	0.067	11	0.216	20	0.931
3	0.097	12	0.218	21	0.285
4	0.076	13	0.272	22	0.479
5	0.141	14	0.134	23	0.645
6	0.065	15	0.188	24	0.468
7	0.133	16	0.285	25	0.699
8	0.126	17	0.635	26	0.786
9	0.123	18	0.645	27	0.453

And we can get
$z_i(\omega)$=(0.042, 0.067, 0.097, 0.076, 0.141, 0.065, 0.133, 0.126, 0.123, 0.167, 0.216, 0.218, 0.272, 0.134, 0.188, 0.285, 0.635, 0.645, 0.672, 0.931, 0.285, 0.479, 0.645, 0.468, 0.699, 0.786, 0.453).

2.3. Comparative Analysis of The Experimental Results

Use the metric provided by this paper, Dr. Y. Zhou's method [24], Dr. T. Yi's method [25-27] to calculate 27 class diagrams' complexity respectively, as shown in Table 5 and Figure 1. In order to highlight the comparative, the numerical results of this paper expanded ten times in Figure 1. Values of Understandability, analyzability and maintainability in Table 5, came from experiments of M. Genero [11-20].

Table 5. Comparative Analysis

Class diagram	Zhou's metric	Yi's metric	This paper's metric	Understand -ability	Analyz -ability	Maintain -ability
1	0	0.614712	0.042	1	1	1
2	0.673012	0.912519	0.067	2	2	2
3	0.940493	1.057623	0.097	2	2	2

4	1.386294	1.287848	0.076	2	2	2
5	0.989909	1.088709	0.141	2	2	2
6	0.693147	0.939484	0.065	2	2	2
7	1.146880	1.159145	0.133	2	3	3
8	1.206376	1.421116	0.126	3	3	3
9	0.381909	0.771502	0.123	2	2	2
10	1.271002	1.415847	0.167	3	3	3
11	1.165030	1.321637	0.216	3	3	3
12	1.553338	1.356727	0.218	3	3	3
13	1.414547	1.342705	0.272	3	3	3
14	0.693147	0.974357	0.134	2	2	2
15	1.303487	1.376364	0.188	2	3	3
16	0.043080	0.65819	0.285	4	4	4
17	1.787461	1.600302	0.635	6	6	6
18	1.861200	1.61728	0.645	6	6	6
19	1.949444	1.661402	0.672	6	5	6
20	1.883662	1.628511	0.931	6	6	7
21	1.277816	1.23228	0.285	3	3	3
22	1.649751	1.531996	0.479	5	5	5
23	1.794866	1.635884	0.645	6	6	6
24	1.480208	1.398272	0.468	5	5	5
25	1.984666	1.68861	0.699	5	6	5
26	2.020782	1.71472	0.786	6	5	6
27	2.030221	1.696749	0.453	4	5	5

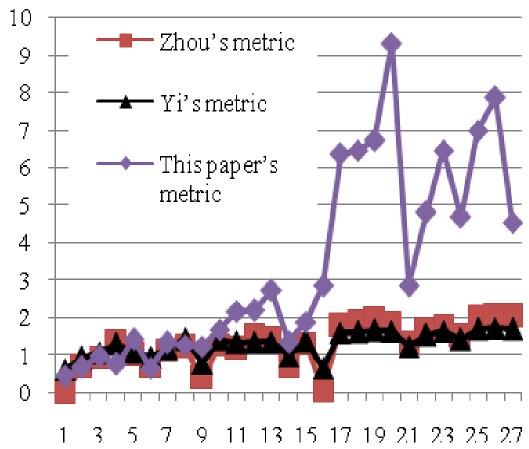

Figure 1. Comparative Analysis

From Figure 1, we can find the results by three metrics are similar, but we can see some interesting conclusions.

1) For class diagram 4, the complexity calculated by the former two metrics is large, but the level given by human experience is low, so the new metric by this paper is similar to human experience.
2) For class diagram 16, the complexity calculated by the former two metrics is low, but the level given by human experience is large, so the new metric by this paper is similar to human experience.
3) For class diagram 26 and 27, the complexity calculated by Dr. Y. Zhou's metric is large, but the level given by human experience is low, so the new metric by this paper is similar to human experience.
4) When the Understandability consistent, but the complexity calculated by the new metric for class diagrams are not equal. Such as, when the value of Understandability is 2, we can get class diagram 2, class diagram 3, class diagram4, class diagram5, class diagram 6, class diagram 7, class diagram9, class diagram 14, and class diagram 15, and the values of those class diagrams by the new metric are not equal, as shown in Figure 2.

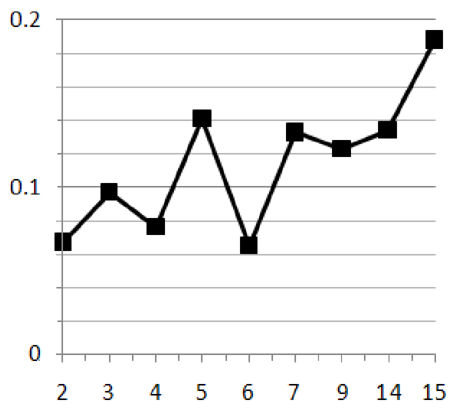

Figure 2. Comparative Understandability 1

At the same time, when the value of Understandability is 3, we can get class diagram 8, class diagram 10, class diagram 11, class diagram 12, class diagram 13, and class diagram 21, and the values of those class diagrams by the new metric are not equal, as shown in Figure 3.

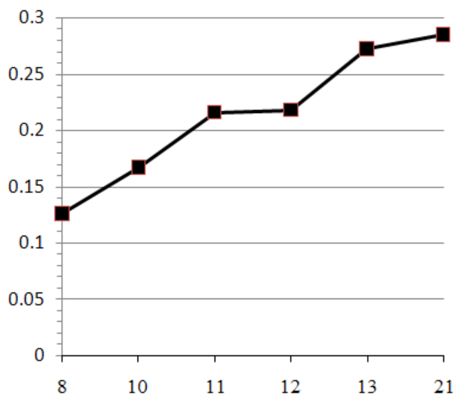

Figure 3. Comparative Understandability 2

Through analysis and comparison, the within class attributes and methods are more important influence on the relationships between classes in Class diagram. Limited to the space, we will discuss which influence the complexity of class diagrams more, the within class attributes and methods or relationships between classes in another paper.

3. Conclusion

This paper, based on the information entropy-based multi-attribute decision theories and methods, presents a new UML class diagram metric. Experimental results show that the metric is highly connected to human's experience, and can effectively measure the complexity of object-oriented software measurement. On one hand, the results of this study can serve the software quality control and evaluation modeling based on software measurement, improve the precision, and can be applied to improve software quality, software maintenance work; on the other hand, it can enrich and improve the software measurement research.

Acknowledgements

In this paper, I would like to show my gratitude to M. Genero at the Department of Computer Science at the University of Castilla-La Mancha, Ciduad Real, Spain, for allowing us to quote the twenty-seven UML class diagrams related to bank information systems and the corresponding metric values. And also thank my students, Longling Huang and Qingtao Yan.

References

[1] R. J. Rubey and R. D. Hartwick, "Quantitative Measurement Program Quality", Proceedings of the 23rd ACM National Computer Conference, **(1968)**, New York.
[2] K. L. Morris, "Metrics for Object-oriented Software Development Environments", PhD thesis, Massachusetts Institute of Technology, Boston, **(1989)**.
[3] M. Rossi, S. Brinkkemper, "Complexity Metrics for Systems Development Methods and Techniques", Information Systems, vol. 21, **(1996)**, pp. 209-227.
[4] T. Mccabe, "A Complexity Measure", IEEE Transactions on Software Engineering, vol. 2, **(1976)**, pp. 308-320.
[5] M. H. Halstead, "Elements of Software Science", Elsevier North-Holland, New York, **(1977)**.
[6] S. Chidamber and C. Kemerer, "A Metrics Suite for Object-oriented Design", IEEE Transactions on Software Engineering, vol. 20, **(1994)**, pp. 476-493.
[7] J. Warmer and A. Kleppe, "The Object Constraint Language: Precise Modeling with UML", Addison Wesley Publishing Company, Boston, **(1999)**.
[8] "OMG, OMG Unified Modeling Language Specification", Version 1.5, Object Management Group, **(2004)**, http://www.omg.org/uml/.
[9] W. Shao, Y. Jiang and Z. Ma, "The Present Problems and Roadmap of UML", Chinese Journal of Computer Research & Development, vol. 40, no. 4, **(2003)**, pp. 509-516.
[10] M. Marehesi, "OOA Metrics for the Unified Modeling Language", Proceedings of the 2rd Euro micro Conference on Software Maintenance and Reengineering, **(1998)** March 8-11, Palazzo degli Affari, Italy.
[11] M. Genero, "Defining and Validating Metrics for Conceptual Models", PhD thesis, University of Castilla-La Mancha, Ciudad Real, **(2002)**.
[12] M. Genero, L. Jiménez and M. Piattini, "A Controlled Experiment for Validating Class Diagram Structural Complexity Metrics", Proceedings of the 8th International Conference on Object-Oriented Information Systems (OOIS2002), **(2002)** September, Montpellier, France.
[13] M. Genero, M. E. Manso, M. Piattini and F. García, "Early Metrics for Object Oriented Information Systems", Proceedings of the 6th International Conference on Object-oriented Information Systems (OOIS2000), **(2000)** December, London, UK.
[14] M. Genero, J. Olivas, M. Piattini and F. Romero, "Using Metrics to Predict OO Information Systems Maintainability", Proceeding of the 13th International Conference on Advanced Information Systems Engineering (CAiSE2001), **(2001)** June, Interlaken, Switzerland.
[15] M. Genero and M. Piattini, "Empirical Validation of Measures for Class Diagram Structural Complexity through Controlled Experiments", Proceedings of the 5th International ECOOP Workshop on Quantitative Approaches in Object-oriented Software Engineering (QAOOSE 2001), **(2001)** June, Budapest, Hungary.
[16] M. Genero, M. Piattini and C. Calero, "Early Measures for UML Class Diagrams", L´Objet. Hermes Science Publications, vol. 6, no. 4, **(2000)**, pp. 489-515.

[17] M. Genero, M. Piattini and C. Calero, "Empirical Validation of Class Diagram Metrics", Proceedings of the International Symposium on Empirical Software Engineering, **(2002)** October, Nara, Japan.
[18] M. Genero and L. P. Jimenez, "Empirical Validation of Class Diagram Complexity Metrics", Proceedings of the 21th International Conference of the Chilean Computer Science Society (SCCC2001), **(2001)** November, Punta Arenas, Chile.
[19] M. Genero, M. Piattini, M. Manso and G. Cantone, "Building UML Class Diagram Maintainability Prediction Models based on Early Metrics", Proceedings of the 9th International Software Metrics Symposium, **(2003)** September, Sydney, Australia.
[20] M. Manso, M. Genero and M. Piattini, "No-redundant Metrics for UML Class Diagram Structural Complexity", Lecture Notes on Computer Science, vol. 2681, **(2003)**, pp. 127-142.
[21] P. In, S. Kim and M. Barry, "UML-based Object-oriented Metrics for Architecture Complexity Analysis", Department of Computer Science, Texas A&M University, **(2003)**, http://faculty.cs.tamu.edu/hohin.
[22] R. Rufai, "New Structure Similarity Metrics for UML Models, Master thesis, Computer Science", King Fahd University of Petroleum & Minerals, Dhahran, **(2003)**.
[23] Z. Chen, "Research on Technical Analysis based on Program Slicing", PhD thesis, Southeast University, Nanjing, **(2003)**.
[24] Y. Zhou, "Research on Software Measurement", PhD thesis, Southeast University, Nanjing, **(2002)**.
[25] T. Yi and F. Wu, "Empirical Analysis of Entropy Distance Metric for UML Class Diagrams", ACM SIGSOFT Software Engineering Notes, vol. 29, no. 5, **(2004)**, pp. 11-16.
[26] S. Xu, T. Yi and F. Wu, "Evaluating Structure Complexity Metric for UML Class Diagrams", Chinese Journal of Electronics, vol. 15, no. 3, **(2006)**, pp. 389-392.
[27] T. Yi, "Authoritative Complexity Measure for Classes", Journal of Chinese Computer Systems, vol. 30, **(2009)**, pp. 271-274.
[28] C. Yue, "Decision Theory and Method", Science Press, Beijing, **(2001)**.
[29] C. L. Hwang, "Group Decision Making under Multiple Criteria: Methods and Applications", Springer, Berlin, **(1987)**.
[30] Shannon, "A Mathematical Theory of Communication", The Bell System Technical Journal, vol. 27, **(1948)**, pp. 379-423,623-656.
[31] J. Han and M. Kamber, "Data Mining: Concepts and techniques (2nd edition)", China Machine Press, Beijing, **(2006)**.
[32] C. L. Hwang and K. Yoon, "Multiple Attribute Decision Making and Applications", Springer Verlag, New York, **(1981)**.

Author

Tong Yi, he received his Ph.D. degrees from Jiangxi University of Finance and Economics in 2006. His current research interests include UML, metrics, and program analysis.