

# Empirical Analysis of Entropy Distance Metric for UML Class Diagrams

Tong Yi

Fangjun Wu

Department of Computer Science & Engineering, Southeast University, Nanjing 210096, China

Laboratory of Computer Center, Yichun University, Yichun 336000, China

<tongyi@seu.edu.cn>

## Abstract

Many software systems built in recent years have been developed using the UML and, in some cases, they already need adaptive maintenance in order to satisfy market and customer needs. Thus a strong emphasis on analysis metrics for software development is necessary. Analysis metrics play an important role in helping developers understand software and, hence, improve software quality and developer productivity. In this paper, we provide empirical evidence for supporting the role of the structure complexity metrics for UML class diagrams, specifically Zhou's metric. Our results, based on data related with bank information system, indicate that the metric is basically consistent with human beings' intuitions.

**Keywords:** UML class diagram, software measure, structure complexity, entropy distance, correlation analysis

## Introduction

Since software metrics were proposed in 1970s, numerous efforts have been done on software complexity metrics. Therefore a lot of metrics have been proposed for measuring OO software systems [1-4,6,7,13,21], which mainly focus on the later phase of the development of software systems. However there is only little work [8-12,15-19,23,25,26] has been done on analyzing UML class diagrams [20]. This can help software developers analyze reliability, maintainability and complexity of systems in the early phase of the OO software lifecycle. Statistical data also show that it is much less expensive to correct software bugs at the early stage rather than the later stage of the development process when the final system has already been implemented and integrated together.

A literature search for publications describing metrics for UML class diagrams identified the following work in this area. *M. Marchesi* proposes a set of indicators to measure the complexity of class diagrams [18]. They are the total number of classes (*OA1*), the total number of inheritance hierarchies (*OA2*), the average weighted responsibilities of classes (*OA3*), the standard deviation of the weighted number of responsibilities of classes (*OA4*), the average number of direct dependencies of classes (*OA5*), the standard deviation of the number of direct dependencies (*OA6*), and the percentage of inherited responsibilities with respect to their total number (*OA7*). To overcome the limitations of *M. Marchesi's* metric, *M. Genero* also discusses a group of indicators [8-12,19]. 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*); the 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* re-

spectively. For the sake of brevity, related works done by *P. In* [15], *R. Rufai* [23] are cited, but are not explored in detail. These studies believe that attributes and methods in classes have impact on the complexity of a class diagram. However, there is still another metric proposed by *Y. Zhou* [25,26], who uses entropy distance and believes that relationships among classes are the key factors of the structure complexity. Similarly, *D. Kang* also uses entropy distance [16,17]. But in essence, the Zhou's and Kang's metrics are the same. Up to now, no consensus has yet arisen as to which viewpoint is better. One of the disadvantages of the former viewpoint lies in that it is difficult to compare the complexity of different class diagrams. And another lies in that different indicators might report conflicting results. These two cases do not exist in the latter.

Although there exists an extensive body of work about software measures, we are unable to find any published proof that entropy distance is "adequate" as a metrics. Besides the metric proposed by *Y. Zhou* [25,26] and *D. Kang* [16,17] themselves, little or no attention is paid to empirical analysis of them. In order to study the Zhou's metric systematically and deeply, this paper analyzes it from empirical estimation. Our focus is primarily on work done in Ref. [16,17,25,26].

This paper is organized as follows: Section 2 gives a brief overview of the entropy distance based structure complexity metric. Section 3 empirically analyzes the metric on data related with bank information system, and the conclusions are drawn in Section 4.

## Entropy Distance Metric for UML Class Diagrams

In comparison with the previous metrics, the metric proposed in Ref. [16,17,25,26] only uses one indicator, namely entropy distance based structure complexity metric, to evaluate the complexity of class diagrams. The metric considers the number of relationships among classes, the interaction pattern of classes, and the kinds of relationships. It almost involves in all relationships that exist among classes during the analysis phase, such as realization, generalization with abstract or concrete superclass, composition and binding relationships. Moreover, the association relationships are classified into common association, qualified association and association class. The metric first specifies weight complexities for various relationships respectively. Then it gives some rules to transform a class diagram into a weighted class dependence graph. Finally the structure complexity of a class diagram is defined as the entropy distance of the corresponding weighted class dependence graph.

For the purposes of discussion and self-containment, the definitions of class diagram *D*, weighted class dependence graph (WCDG) and its corresponding matrix *W* are displayed as Definitions 1, 2, and 3, respectively.

**Definition 1** [25]. Let *D* be a UML class diagram and  $T(D) = \{t_i \mid 1 \leq i \leq 10\}$ , then  $V(D) = \{c \mid c \text{ is a class or template in } D\}$ ,  $R(D) = \{(n_1, n_2,$

$t, d) | n_1, n_2 \in V(D) \wedge t \in T - \{t_4\} \wedge d \in \{0, 1, 2\} \wedge$  there is a relationship  $t$  between  $n_1$  and  $n_2$ , and the direction of  $t$  is  $d\} \cup \{(n_1, n_2, t_4, n_3) | n_1, n_2, n_3 \in V(D) \wedge$  there is a relationship  $t_4$  between  $n_1$  and  $n_2$ , and  $n_3$  is an association class $\}$ , where  $d$  could be 0, 1 or 2. If  $d$  is 0, then the direction is bi-directional. If  $d$  is 1, then the direction is from  $n_1$  to  $n_2$ . If  $d$  is 2, then the direction is from  $n_2$  to  $n_1$ .

**Definition 2**[25]. Let  $D$  be a UML class diagram, then the corresponding weighted class dependence graph (WCDG) of  $D$  is a two-tuples  $(V(D), E(D))$ , where,  $V(D)$  is the same as that in Definition 1;  $E(D) = \{ \langle n_1, n_2, h \rangle \mid (n_1, n_2, t, d) \in R(D) \wedge t \neq t_4 \wedge h = \sum_{\substack{t_i \in T(D) \wedge ((n_1, n_2, t_i, 0) \in R(D) \\ \vee (n_1, n_2, t_i, 1) \in R(D) \vee (n_2, n_1, t_i, 2) \in R(D))}} h(t_i) \} \cup \{ \langle n_1, n_3, h_4 \rangle, \langle n_3, n_1, h_4 \rangle, \langle n_2, n_3, h_4 \rangle, \langle n_3, n_2, h_4 \rangle \mid (n_1, n_2, t_4, n_3) \in R(D) \}$ .

Definition 2 states that the weight of an edge in a WCDG represents the degree of dependence among the corresponding classes or templates in a class diagram. Because there might exist more than one relationship between two classes, the weight of the corresponding edge in a WCDG should equal to the sum of the weights of all these relations that have the same direction.

**Definition 3**[25]. Let  $D$  and  $WCDG(D)$  be a UML class diagram and its corresponding weighted class dependence graph respectively, then a weighted matrix  $W$  represents the weights of the edges in  $WCDG(D)$ . For any two nodes  $n_1$  and  $n_2$  in  $V(D)$ , it is easy to know that the weight between  $n_1$  and  $n_2$  is

$$W(n_1, n_2) = \begin{cases} h & \text{if } \langle n_1, n_2, h \rangle \in E(D) \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

For any two different class diagrams, the corresponding weighted class dependence graphs might not be the same because the relationships among class nodes (or template nodes) are different. Therefore, the nodes will have different number of incoming or outgoing edges. That is to say, the occurrence of incoming or outgoing edges of nodes is random, which can be described by two discrete random variables  $X$  and  $Y$ .

**Definition 4**[25]. Let  $D$  and  $WCDG(D)$  be a UML class diagram and its corresponding weighted class dependence graph respectively,  $X$  and  $Y$  be two random variables that describe the occurrence probabilities of outgoing edges and incoming edges that every node in  $V(D)$  has, then the structure complexity of class diagram  $D$  is:

$$Complexity(D) = \begin{cases} H(X, Y) - I(X; Y) & \text{if } R(D) \neq \Phi \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Where  $H(X, Y)$  is joint entropy of  $X$  and  $Y$ , which can be calculated by formula (7);  $I(X; Y)$  is mutual information between  $X$  and  $Y$ , which can be calculated by formulas (8) and (9).

Let  $X$  and  $Y$  be two random variables that take on two sets of possible values,  $W_x = \{x_i \mid 1 \leq i \leq m\}$  and  $W_y = \{y_j \mid 1 \leq j \leq n\}$  respectively, having probability [5]

$$p(x_i) = \sum_{n_2 \in V(D)} W(x_i, n_2) / \sum_{n_1 \in V(D)} \sum_{n_2 \in V(D)} W(n_1, n_2), \quad (3)$$

$$\text{and } p(y_j) = \sum_{n_1 \in V(D)} W(n_1, y_j) / \sum_{n_1 \in V(D)} \sum_{n_2 \in V(D)} W(n_1, n_2). \quad (4)$$

According to the information theory, if the joint probability of  $x_i$  and  $y_j$  is [5]

$$p(x_i, y_j) = W(x_i, y_j) / \sum_{n_1 \in V(D)} \sum_{n_2 \in V(D)} W(n_1, n_2), \quad (5)$$

$$\text{and } p(x_i \mid y_j) = p(x_i, y_j) / p(y_j), \quad (6)$$

then the joint entropy  $H(X, Y)$  of  $X$  and  $Y$  is [5]

$$H(X, Y) = - \sum_{x_i \in W_x \wedge y_j \in W_y} p(x_i, y_j) \log p(x_i, y_j) \quad (7)$$

The conditional entropy of  $X$  with a given  $Y$  is [5]

$$H(X \mid Y) = - \sum_{x_i \in W_x \wedge y_j \in W_y} p(x_i, y_j) \log p(x_i \mid y_j), \quad (8)$$

And the mutual information between  $X$  and  $Y$  is [5]

$$I(X; Y) = H(X) - H(X \mid Y). \quad (9)$$

## Empirical Validation

With the development of software metrics, numerous efforts have been done on their empirical validations [1-4, 6-13, 21, 24]. *A. Binkley* plays an experiment to examine the correlation relationship between the maintenance measure and the inheritance-based metrics *NCIM*, *NOC*, and *NOD* [2]. Then he concludes that: there is no statistically significant correlation between the maintenance measure and the inheritance-based metrics *NCIM*, *NOC*, and *NOD*. However, there is a significant correlation between the maintenance measure and the *CHNL* metric. This implies that inheritance does not play a significant role in predicting fault-prone classes. *M. Genero* also performs some experiments and concludes that: there is a high correlation between most of the UML class diagram structural complexity metrics and maintenance time. The experiments highlight the most relevant metrics associated with each maintainability sub-characteristic, which are for understandability *NC*, *NA*, *NM*, *NAGg*, *NGen*, for analysability *NC*, *NA*, *NM*, *NAGg*, and for modifiability *NC*, *NA* and *NM*. *NDep* is the only one that has a lesser correlation. Table 1 summaries some of the typical experiments related with maintenance or its sub-characteristics.

Studies	Dependent variables	Independent variables	Technique	Object system
BS [2]	Maintenance effort	Number of Clients, Fan-in, Simple Class Coupling, RFC, Fan-out, WMC, CHNL, NCIM, NOD, NOC, CDM	Spearman rank correlation	A patient collaborative care system
WD[24]	Maintenance time	DIT	Standard significance testing, Wilcoxon rank sum test	Six systems developed in C++
HC[13]	Modifiability, understandability	DIT	Chi-square analysis	Two systems, each with two versions
BB[4]	Understanding, modification	Coad and Yourdon quality design principles (Coupling, Cohesion, Clarity of design, Generalization-Specialization depth): DIT, NOC, and CBO	2 x 2 factorial Kruskal-Wallis test	Replication package
FN [6]	Adaptive maintenance effort	NCL, NRC, TNM, TLOC, MCC, MNA, MNM	multilinear regression analysis	Music object-oriented distributed system coded in C++
PD [21]	The completeness and correctness of the modifications	The number of levels in the inheritance structure	ANOVA, parametric t-test	Four models of a hotel administration, written in MERODE
GJP[9]	Understandability, analyzability, modifiability	NC, NA, NM, NAssoc, NAgg, NAggH, NDep, NGen, NGenH, MaxHAgg, MaxDIT	Fuzzy classification, regression trees, Spearman rank correlation, Kolmogorov-Smirnov test, Principal Component Analysis	Twenty-eight UML class diagrams related to Bank Information Systems, nine different UML classes of diagrams
BV[1]	Maintenance time	Interaction Level (IL), Interface Size (IS), and Operation Argument Complexity (OAC)	ANOVA, correlation, single and multiple regression analysis	Quadrilateral, tractor-trailer

**Table 1. Summary of empirical studies of object-oriented metrics for maintenances**

Class diagram	Result	Understandability	Analyzability	Modifiability	Class diagram	Result	Understandability	Analyzability	Modifiability
1	0	1	1	1	15	0.566097	2	3	3
2	0.292285	2	2	2	16	0.018709	4	4	4
3	0.4084509	2	2	2	17	0.7762845	6	6	6
4	0.6020599	2	2	2	18	0.808309	6	6	6
5	0.4299120	2	2	2	19	0.846633	6	5	6
6	0.3010299	2	2	2	20	0.818065	6	6	7
7	0.4980837	2	3	3	21	0.554949	3	3	3
8	0.5239225	3	3	3	22	0.716478	5	5	5
9	0.165861	2	2	2	23	0.779500	6	6	6
10	0.551989	3	3	3	24	0.6428462	5	5	5
11	0.505966	3	3	3	25	0.8619295	5	6	5
12	0.674606	3	3	3	26	0.8776146	6	5	6
13	0.614329	3	3	3	27	0.8817138	4	5	5
14	0.301029	2	2	2					

**Table 2. Result values computed by entropy distance metric**

	N	Range	Minimum	Maximum	Mean		Std. deviation	Variance	Skewness		Kurtosis	
					Statistic	Std. error			Statistic	Std. error	Statistic	Std. error
Zhou's metric	27	.8817	.0000	.881714	.55624639	.0487091	.253100206	.064	-.669	.448	-.206	.872
Understandability	27	5.00	1.00	6.00	3.5556	.3217	1.67179	2.795	.402	.448	-1.374	.872
Analyzability	27	5.00	1.00	6.00	3.6296	.3073	1.59683	2.550	.296	.448	-1.338	.872
Modifiability	27	6.00	1.00	7.00	3.7037	.3282	1.70553	2.909	.401	.448	-1.188	.872

**Table 3. Descriptive statistics for the data in Table 2**

For a given UML class diagram  $D$ , we can easily compute its structure complexity according to Zhou's metric. However, whether the metric value is correct or not remains to be investigated.

In Section 2 we introduce the entropy distance metric, which still needs empirical validation though no consensus has yet arisen between theoretical and empirical validations. Generally, a metric may be invalid in practice while be valid in theoretical argument, and vice versa. Whether a metric is valid depends on whether it is consistent with human beings' views of complexity or not.

Practically, people would like to select the practical metrics.

In the following, we will carry out an experiment to estimate the entropy distance metric. 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 [8-12,19]. *M. Genero* et al. carry out some comprehensive controlled experiments [8-12, 19].

- In one study [8-12], experimental subjects are seven professors in the software engineering area with enough experience in the design and development of OO software, and ten students in the final year of computer science in the department of computer science at the university of Castilla-La Mancha in Spain.
- In another study [19], experimental subjects are twenty-four students in the third year of computer science in the department of computer science at the university of Castilla-La Mancha in Spain, and twenty-six students in the fourth year of computer science in the Dipartimento di Informatica, Sistemi e Produzione at the Università degli Studi di Roma Tor Vergata in Italy. Understandability, analyzability and modifiability variables are measured according to subject's rating.

Table 2 shows metric values computed by the metric\*. And Table 3 displays descriptive statistics about the results. It can be seen from Table 2 that: for the No.4 class diagram, the metric value is larger while the value rated by those subjects is lower; for the No.16 class diagram, the thing is contrary. They all have large differences between the metric and rated values. The reason may lie in that the entropy distance metric only considers relationships among classes or it has some limitations, even errors.

To find out the relationships between the metric and human beings' intuition, we adopt correlation analysis techniques [22]. The correlation between two variables reflects the degree to which the variables are related. It ranges from -1 to +1. A correlation of +1 means that there is a perfect positive relationship between variables; -1, a perfect negative relationship; and 0, no relationship. The most common measures of correlation are the Pearson and Spearman correlations. They reflect the degree of linear relationship between two variables. In this paper, we use them to calculate coefficient in SPSS [14], a well-known statistics software tool. Analyzing Table 2, we have the results shown in Tables 4, 5 and 6.

From Tables 4, 5, and 6, we can conclude that

- *Zhou's metric* vs. understandability,
- *Zhou's metric* vs. analyzability,
- *Zhou's metric* vs. modifiability.

are positive correlation. Furthermore, they are perfect positive correlation.

So we come to conclusion that *Zhou's metric* is significant correlated with human beings' intuition in this experiment.

			<i>Zhou's metric</i>	Understandability
<i>Zhou's metric</i>	Pearson	Pearson correlation	1	.741*
		Sig. (2-tailed)	.	.000
	Spearman	Correlation coefficient	1.000	.802**
		Sig. (2-tailed)	.	.000
Understandability	Pearson	Pearson correlation	.741*	1
		Sig. (2-tailed)	.000	.
	Spearman	Correlation coefficient	.802**	1.000
		Sig. (2-tailed)	.000	.

\* and \*\* Correlation are significant at the 0.01 level (2-tailed).

**Table 4 Correlation relationships between *Zhou's metric* and understandability**

\* Notes: In Table 2, metric values in column understandability, analyzability, and modifiability are quoted from the experiments performed by *M. Genero*, and the enrolled subjects rate them according to experiences.  
<http://alarcos.inf-cr.uclm.es/english/asp/labpackage.asp>

			Zhou's metric	Analyzability
Zhou's metric	Pearson	Pearson correlation	1	.773*
		Sig. (2-tailed)	.	.000
	Spearman	Correlation coefficient	1.000	.82**
		Sig. (2-tailed)	.	.000
Analyzability	Pearson	Pearson correlation	.773*	1
		Sig. (2-tailed)	.000	.
	Spearman	Correlation coefficient	.82**	1.000
		Sig. (2-tailed)	.000	.

**Table 5. Correlation relationships between Zhou's metric and analyzability**

			Zhou's metric	Modifiability
Zhou's metric	Pearson	Pearson correlation	1	.775*
		Sig. (2-tailed)	.	.000
	Spearman	Correlation coefficient	1.000	.827**
		Sig. (2-tailed)	.	.000
Modifiability	Pearson	Pearson correlation	.775*	1
		Sig. (2-tailed)	.000	.
	Spearman	Correlation coefficient	.827**	1.000
		Sig. (2-tailed)	.000	.

\* and \*\* Correlation are significant at the 0.01 level (2-tailed).

**Table 6. Correlation relationships between Zhou's metric and modifiability**

## Conclusions and Future Directions

In this paper, we try to analyze the entropy distance metric for UML class diagrams experimentally. The analysis shows that it has some errors as well as shortcomings. It should be marked here that there is a reasonable chance that useful class diagram maintainability models could be built at the initial phase of the OO software lifecycle, thus allowing OO software designers to make better decisions early in the OO software development lifecycle.

More and more attentions are paid to the complexity of UML class diagram, and more and more research results are got. For example, D. Kang improves the entropy distance metric to some degree, which can distinguish complexities among the same kinds of relationships [16,17]. We believe that the work in Ref. [25,26], as an initial step, will encourage a sensible look at entropy-based complexity metrics for UML class diagrams and ultimately lead to the definition of good meaningful and useful metrics.

## Acknowledgements

We are very grateful to M. Genero at the Department of Computer Science at the University of Castilla-La Mancha, Ciudad Real, Spain for allowing us to quote the twenty-seven UML class diagrams related to bank information systems and the corresponding metric values.

## References

- [1] Bandi, R., Vaishnavi, V., Turk, D. (2003): Predicting maintenance performance using object-oriented design complexity metrics. *IEEE Transactions on Software Engineering*, 29(1), 2003, pp.77 -87.
- [2] Binkley, A., Schach S (1997): Inheritance-based metrics for predicting maintenance effort: an empirical study. Technical Report TR97-05, *Computer Science Department, Vanderbilt University*, 1997.
- [3] Binkley, A., Schach S (1998): Validation of the coupling dependency metrics as a predictor of run-time failures and maintenance measures. In *ICSE'98-Proceedings of the 1998 International Conference on Software Engineering*, April 19-25, 1998, Kyoto, Japan, pp. 452-455.
- [4] Briand, L., Bunse, C., Daly, J. (2001): A controlled experiment for evaluating quality guidelines on the maintainability of object-oriented designs. *IEEE Transactions on Software Engineering*, 27(6), 2001, pp.513-530.
- [5] Cover, T. M., Thomas, J. A. (1991): Elements of information theory. *New York: John Wiley & Sons, Inc.*, 1991.
- [6] Fioravanto, F., Nesi, P. (2001): Estimation and prediction metrics for adaptive maintenance effort of object-oriented systems. *IEEE Transactions on Software Engineering*, 27(12), 2001, pp.1062-1084.
- [7] Geert, P., Guido D. (2001): Evaluating the effect of inheritance on the modifiability of object-oriented business domain models. In *CSMR'2001-Lisbon, Portugal. Pedro Sousa, Jürgen Ebert (Eds): 5<sup>th</sup> European Conference on Software Maintenance and Reengineering*, March 14-16, 2001, Lisbon, Portugal, pp.20-29.
- [8] Genero, M. (2002): Defining and validating metrics for conceptual models [Ph.D. thesis]. *University of Castilla-La Mancha*, 2002.
- [9] Genero, M., Jiménez, L., Piattini, M. (2002): A controlled experiment for validating class diagram structural complexity metrics. In *Proceedings of the 8<sup>th</sup> International Conference on object-oriented Information Systems (OOIS 2002)*, Montpellier, France, September, 2002, pp.372-383.
- [10] Genero, M., Piattini, M., Calero, C. (2000): Early measures for UML class diagrams. *L'Objet*. Hermes Science Publications, 6(4), 2000, pp.489-515.
- [11] Genero, M., Piattini, M., Calero, C. (2002): Empirical validation of class diagram metrics. In *Proceedings of 2002 International Symposium on Empirical Software Engineering*, Nara, Japan, October 2002, pp.195-203.

- [12] Genero, M., Piattini, M., Manso, M., Cantone, G. (2003): Building UML class diagram maintainability prediction models based on early metrics. In *Proceedings of 9<sup>th</sup> International Software Metrics Symposium*, Sydney, Australia, September 2003, pp.263-275.
- [13] Harrison, R., Counsell, S., Nithi, R. (2000): Experimental assessment of the effect of inheritance on the maintainability of object oriented systems. *Journal of Systems and Software*, 52, 2000, pp.173-179.
- [14] <http://www.spss.com>
- [15] In, P., Kim, S., Barry, M. (2003): UML-based object-oriented metrics for architecture complexity analysis. *Department of computer science, Texas A&M University*, 2003. <http://faculty.cs.tamu.edu/hohin>
- [16] Kang, D., et al. (2004): A structural complexity measure for UML class diagrams. In *International Conference on Computational Science 2004 (ICCS 2004)*, Krakow Poland, June 2004, pp.431-435.
- [17] Kang, D., et al. (2004): A complexity measure for ontology based on UML. In *IEEE 10<sup>th</sup> International Workshop on Future Trends in Distributed Computing Systems (FTDCS 2004)*, Suzhou, China, May 2004, pp.222-228.
- [18] Marchesi, M. (1998): OOA metrics for the unified modeling languages. In *Proceedings of 2<sup>nd</sup> Euromicro Conference on Software Maintenance and Reengineering (CSMR'98)*, Palazzo degli Affari, Italy, March, 1998, pp.67-73.
- [19] Manso, M., Genero, M., Piattini, M. (2003): No-redundant metrics for UML class diagram structural complexity. *Lecture Notes on Computer Science*, 2681, 2003, pp.127-142.
- [20] OMG unified modeling language specification 2.0, *Object Management Group, Inc.*, March 2004. <http://www.omg.org/uml/>
- [21] Prechelt, L., Unger, B., Philippsen, M., Tichy, W. (2003): A controlled experiment on inheritance depth as a cost factor for code maintenance. *Journal of Systems and Software*, 65(2), 2003, pp.115-126.
- [22] Rencher, A. C. (2002): Methods of multivariate analysis (second edition). *New York: John Wiley & Sons, Inc.*, 2002.
- [23] Rufai, R. (2003): New structure similarity metrics for UML models [Master Thesis]. *Computer Science, King Fahd University of Petroleum & Minerals*, 2003.
- [24] Wood, M., Daly, J., Miller, J., Roper, M. (1999): Multi-method research: an empirical investigation of object-oriented technology. *Journal of Systems and Software*, 48(1), 1999, pp.13-26.
- [25] Zhou, Y. (2002): Research on software measurement [Ph.D. Thesis]. *Department of Computer Science and Engineering, Southeast University, Nanjing, P.R. of China*, 2002.
- [26] Zhou, Y., et. al. (2003): Measuring structure complexity of UML class diagrams. *Journal of Electronics (China)*, 20(3), 2003, pp.227-231.

Correspondence to: Tong Yi, Department of Computer Science and Engineering, Southeast University, Nanjing 210096, P. R. China. E-mail: [tongyi@seu.edu.cn](mailto:tongyi@seu.edu.cn)