An Investigation of Design Level Class Cohesion Metrics

Kuljit Kaur and Hardeep Singh
Department of Computer Science and Engineering, Guru Nanak Dev University, India

Abstract: *Design level class cohesion metrics are based on the assumption that if all the methods of a class have access to similar parameter types, then they all process closely related information. A class with a large number of parameter types common in its methods is more cohesive than a class with less number of parameter types common in its methods. In this paper, we review the design level class cohesion metrics with a special focus on metrics which use similarity of parameter types of methods of a class as the basis of its cohesiveness. Basically three metrics fall in this category: Cohesion Among Methods of a Class (CAMC), Normalized Hamming Distance (NHD), and Scaled NHD (SNHD). Keeping in mind the anomalies in the definitions of the existing metrics, a variant of the existing metrics is introduced. It is named NHD Modified (NHDM). A major point of difference is that the NHD metric counts a disagreement only if class methods taken as pairs disagree on a parameter type that one method uses but the other method, in the pair, does not use. It ignores the case when both methods of a pair do not use a parameter type. NHD indirectly counts it as an agreement, but NHDM considers such a case as a disagreement. An automated metric collection tool is used to collect the metrics data from an open source Java based software program containing 884 classes. Metrics data is then subjected to statistical analysis. The NHDM metric shows the maximum amount of variation in data values in comparison to other metrics in the group. NHDM is strongly correlated with CAMC. Unlike the previous studies, no significant correlation is found in CAMC and NHD.*

Keywords: *Design metrics, class cohesion metrics, product quality, cohesion among methods of a class, normalized hamming distance, scaled NHD, NHD modified.*

Received September 10, 2009; accepted March 9, 2010

1. Introduction

In object oriented paradigm, cohesion of a class refers to the degree to which members of the class are interrelated. Metrics have been defined to measure cohesiveness of a class both at design and source code levels. Chidamber and Kemerer [10] defined the first metric to measure cohesiveness of a class. Since then, several cohesion metrics have been proposed (discussed in the next section). Empirical studies report that class cohesion metrics are useful to assess the software design quality [2, 7, 24], to predict fault proneness of classes [18, 25, 28], and to identify reusable components [17, 21].

Design level cohesion metrics are based on the assumption that the types of the method parameters match the types of the attributes accessed by the method. It is further assumed that the set of attribute types accessed by a method is the intersection of this method's parameter types and the set of parameter types of all the methods in the class [3, 12].

In this paper, we review the design level class cohesion metrics with a special focus on metrics which use similarity of parameter types of methods of a class as the basis of its cohesiveness. These metrics are: Cohesion Among Methods of a Class (CAMC), Normalized Hamming Distance (NHD), and Scaled NHD (SNHD) [12]. Keeping in mind the anomalies in the definitions of the existing metrics, a variant of the existing metrics is introduced. NHDM metric is based on the definition of the NHD metric. NHD metric counts a disagreement only if method pairs disagree on a parameter type that one method uses but the other method does not use. It ignores the case when both methods of a pair do not use a parameter type. NHD indirectly counts it as an agreement, but NHDM considers such a case as a disagreement. Metric data is collected from an open source Java program taken from the largest repository of open source software available at www.sourceforge.com.

The paper is organized as follows. Section 2 reviews the related work. Section 3 explains the existing design level class cohesion metrics and introduces a modified version of the existing metrics. Section 4 describes the procedure followed to collect data. Section 5 presents the statistical analysis of the data collected from an open source project, and section 6 concludes the paper.

2. Related Work

Several researchers have proposed metrics for measuring cohesion of a class. Broadly, these proposals fall in two categories - metrics which can be computed at design level (high level) and metrics

which can be computed one step later i.e. at source code level (low level).

A number of class cohesion metrics are defined in the low level metrics category [1, 4, 5, 6, 8, 9, 10, 11, 13, 16, 19, 20, 22, 23, 26, 27]. However, a few proposals for design level class cohesion metrics are there [3, 12, 14, 15]. Design level class cohesion metrics are based on the assumption that if all the methods of a class have access to similar parameter types then they all process closely related information. A class with a large number of parameter types common in its methods is more cohesive than a class with less number of methods sharing common parameter types [3, 12].

The metric, named CAMC captures the information about parameter types of methods of a class [3]. A class is cohesive if all methods of the class use the same set of parameter types. Methods which use same type of parameter types are assumed to process related kind of information. CAMC metric values lie in the range [0, 1]. Counsell *et al.* point out some anomalies in definition of the metric, and propose a new metric named NHD [12]. A variant of the NHD metric called SNHD is introduced in the same paper. It addresses shortcomings of both CAMC and NHD, as claimed by the authors [12]. This research finds anomalies in the definitions of NHD and SNHD as well, and proposes a modified version of the NHD metric - NHD modified (NHDM). The NHDM metric gives statistically significant results.

Dallal [14] proposes another metric in this category. Similarity based Class Cohesion (SCC) metric is based on the assumption that that the set of attribute types accessed by a method is the intersection of the set of this method's parameter types and the set of its class attribute types. This metric is not analyzed in this paper as the automated tool developed for this research does not support collection of this metric.

3. Design Metrics

This section describes the class cohesion metrics computable with the information available at design level. At the design level, information regarding name of the class, its attributes/ variable, and signatures of its methods is available. Method signature includes name of the method and its parameter list which describes names of the parameters and their types. A class does not have a detailed or algorithmic description of its methods available at this level.

3.1. CAMC

The CAMC metric measures the extent of intersection of individual method parameter type lists with the parameter type list of all methods in the class [3]. This metric computes the relatedness among methods of a class based upon the parameter list of the methods. It is assumed that methods of a class, having access to similar parameter types, process closely related information.

The CAMC metric uses a parameter-occurrence matrix (PO matrix) that has a row for each method and a column for each data type that appears at least once as the type of a parameter in at least one method in the class. The value in row *i and* column *j* in the matrix is 1 when the *i*th method has a parameter of the *j*th data type and is 0 otherwise. In the original version of the metric, the PO matrix has one column of all 1s. This column corresponds to the type of the class itself which is by default one of the parameters of every method, the 'self' parameter. This means that one of the columns is filled entirely with 1s. In this discussion, the original version of the metric is referred to as $CAMC_s$ (Cohesion among methods of a class with 'self' parameter) and metric definition without the 'self' parameter is named as CAMC [12].

The CAMC metric is defined as the ratio of the total number of 1s in the PO matrix to the total size of the matrix:

$$CAMC(C) = \frac{\sigma}{kl} \; where \; \sigma = \sum_{1}^{k} \sum_{1}^{l} PO[i][j] \qquad (1)$$

Anomalies in the definition:

1. CAMC gives false positives - the metric takes non-zero value for a class with no parameter sharing in its methods.
2. CAMC can not differentiate between two classes having same number of 1s, but with different patterns of 1s in their PO matrices.
3. Smaller classes take high values for the cohesion metric than the larger classes with same properties.

3.2. NHD

Counsell *et. al.* [12] suggested an alternative of CAMC. It is based on the definition of hamming distance. NHD measures agreement between rows in the PO matrix. NHD metric for a class with k methods and l unique parameter types (union of parameter types received by its methods) is defined as:

$$NHD = \frac{2}{lk(k-1)} \sum_{1}^{k-1} \sum_{j+1}^{k} a(i,j) \qquad (2)$$

Where *a(i,j)* is value of the cell at $(i,j)^{th}$ location in the PO matrix. Another easy way to compute NHD is to first find the sum of disagreements between methods for all the parameter types and then subtract it from 1:

$$NHD = 1 - \frac{2}{lk(k-1)} \sum_{1}^{l} cj(k - cj) \qquad (3)$$

Where c_j is the number of 1s in the j^{th} column of the PO matrix. Similarly NHD_s can be defined for a PO matrix with one column of all 1s. Anomalies in the definition:

1. NHD metric also gives false positives. The metric removes the first anomaly of the CAMC for a class with k=l=2. The metrics fails to give correct answer for higher values of k and l.
2. NHD does not give different answers for classes with different properties - metric fails to distinguish a class with no parameter sharing in its methods from a class with substantial amount of parameter sharing in its methods.
3. Class size influences metric value. As size of the class increases, value of the NHD metric also increases (even if the PO matrix gets sparser).

3.3. SNHD

SNHD is the Scaled NHD metric proposed to interpret values of the NHD metric in a more varied range. Proponents of the NHD metric are of the opinion that NHD metric can take values at two extremes: the minimum or the maximum. But they admit that it is not clear as to which of these extremes represents a cohesive class. However without giving any clear explanation they state that classes at both extremes may be cohesive. They define these extreme values as NHD_{min} and NHD_{max} respectively [12]. SNHD metric value helps to know how close the NHD metric is to the maximum value of the NHD value in comparison to the minimum value. SNHD is defined as follows:

$$SNHD = \begin{cases} 0 & if\ NHD\,min = NHD\,max\ and\ \sigma < kl, \\ 1 & if\ \sigma = kl, \\ 2\left(\dfrac{NHD - NHD\,min}{NHD\,max - NHD\,min}\right) - 1, & otherwise \end{cases} \quad (4)$$

The SNHD metric values lies in the range [-1,1]. SNHD = -1 implies that NHD = NHD_{min}, and SNHD = 1 implies that NHD=NHD_{max}. NHD is closer to its minimum or maximum value depends upon whether SNHD is getting values close to -1 or +1 respectively. A class is considered non-cohesive if SNHD metric for the class is 0. Similarly $SNHD_s$ is defined by considering the 'self' parameter. Anomalies:

1. Difficult to calculate and interpret.
2. False negatives - SNHD metric gives 0 value for a class with good degree of cohesion.

3.4. NHDM

Keeping in view the anomalies of the cohesion metrics discussed above, this research proposes a variation of this metric. This variation of the metric is named as NHDM metric. NHD metric ignored the method pairs with zero values in a column of the PO matrix. It counts only those methods pairs which do not agree, and ignores all other method pairs irrespective of whether they agree on a 0 or a 1. NHDM counts the method pairs which agree on a 0, as a disagreement. NHDM for a class with k methods and l unique parameter types, of its methods, is defined as:

$$NHDM = 1 - \frac{2}{lk(k-1)} \sum_{1}^{l} (cj(k - cj) + \frac{1}{2} zj(zj - 1)) \quad (5)$$

Where c_j is the number of ones and z_j is the number of zeroes in the jth column of the PO matrix for the class. Similarly $NHDM_s$ is defined by including the 'self' parameter in the PO matrix. This metric removes the anomalies present in the definition of CAMC, NHD, and SNHD metrics. NHDM gives correct results. It gives different results for classes with different properties. NHDM metric values are independent of the class size.

4. Data Collection

An open source program in JAVA programming language is downloaded from www.sourceforge.com. A brief description of the procedure followed to collect metrics data from this program is as follows:

1. An XML script is written to generate a list of class names, and parameter lists of individual operations for every class.
2. A parameter occurrence matrix is defined as shown in [12]. It is a mxn matrix, where m represents the number of operations in the class and n represents the number of unique parameters passed to different operations of the class. Columns of the matrix correspond to unique parameters of different operations of the class. Rows correspond to different operations of the class. Parameter occurrence matrix has a 1 in a cell, if the operation corresponding to the row takes the parameters represented by the corresponding column, otherwise it is 0.
3. Calculating Metrics: To calculate CAMC, number of 1s is counted in the parameter occurrence matrix. This number is then divided by the size of the parameter occurrence matrix. To calculate NHD, pairs of operations (which disagree on a parameter type) are considered. For a column of the PO matrix calculate count* (op-count), where count is number of 1s in the column and op is the number of operations in the column. This gives the number of disagreements in pairs of operations in a column. Add the number of such pairs for all the columns. Multiply this number with 2, and call it numer (short form for numerator). Calculate the maximum possible number of pairs of operations of the class. Call this number denom (short for denominator), and calculate it as cplsize x op x (op-1), where op is the number of operations of a class. Divide numer by denom and subtract the number thus obtained from 1. It gives value for the NHD metric. To calculate *SNHD*, NHD is compared with its maximum or minimum value. If *NHD* is closer to its

maximum (minimum) value, *SNHD* takes values closer to 1 (-1). However, if N*HD*'s minimum and maximum values are the same then *SNHD* takes value 0, and it indicates a less cohesive class. To calculate N*HDM*, pairs of operations which disagree on a parameter type are considered. In case of N*HD*, two operations (cells in a column) are said to disagree if they both take different values i.e. 0 and 1. But if they take same values i.e. both are 0, then it is not considered as a disagreement. In this study N*HD* is modified and the modified version is called N*HDM*. N*HDM* considers that a pair of operations does not agree if both the operations (cells in a column) take value 0. For a column of the PO matrix calculate count* (op-count), where count is number of 1s in the column and op is the number of operations in the column. Add to this the number of pairs of operations which disagree on 0 values. This gives the number of disagreements in pairs of operations in a column. Other steps are same with *NHD*. To calculate variations of these metrics like CAMCs, NHDs, SNHDs, and NHDMs, a new column of all 1s is added to the parameter occurrence matrix. Metric calculation after that is on the similar lines as discussed in step number 3.

5. Data Analysis

Cohesion metrics discussed above are collected from an open source software system available at www.sourceforge.net. The software is a charting library, and it has evolved in approximately 40 versions over the past 8 years. It is developed using the JAVA platform, and it consists of 884 classes. For automated collection of metrics, a tool CohMetric is developed using the C programming language.

5.1. Descriptive Analysis

Histograms in Figures 1-4 show metrics distributions. Table 1 presents the descriptive statistics. It can be observed that majority of the CAMC metric values lie close to 0. On average a class's cohesion value is 0.21.

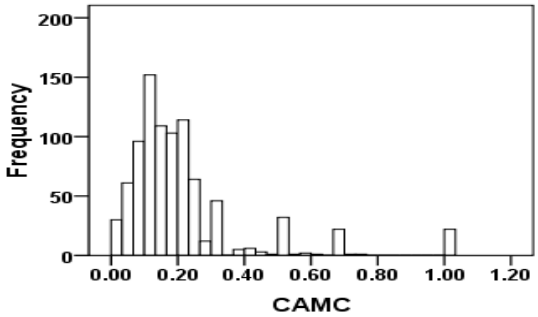

Figure 1. Distribution of CAMC metric.

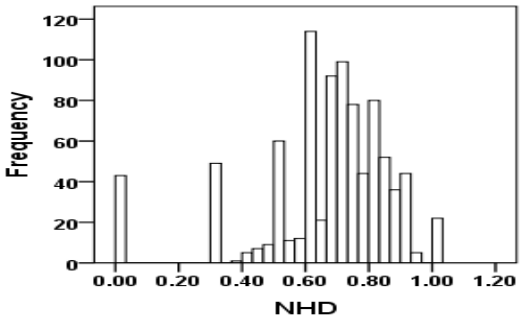

Figure 2. Distribution of NHD metric.

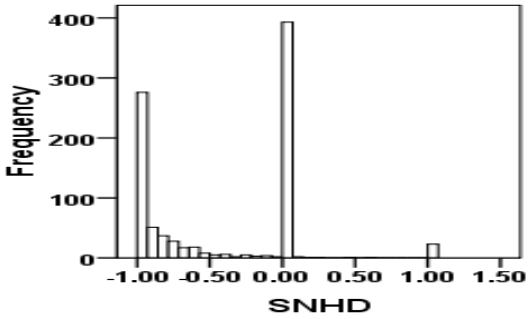

Figure 3. Distribution of SNHD metric.

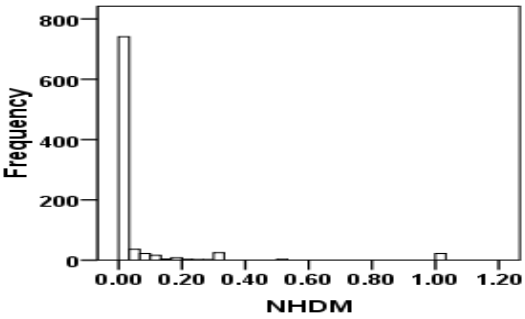

Figure 4. Distribution of NHDM metric.

NHD metric takes values in a higher range. Average NHD metric value is 0.66. NHDM takes very low values. For majority of the classes it is 0. Its average value is just 0.05. As earlier stated, it may be due to the reason that it does not give false positives. SNHD is 0 for maximum of the classes. Its values lie more on the left side of 0 which implies that majority of the classes has NHD more close to NHD_{min} than NHD_{max}. Average SNHD for a class is -0.43 and standard deviation is also very high.

Table 1. Descriptive statistics for cohesion metrics.

Metric	Average	Std. Dev.	Metric	Average	Std. Dev.
CAMC	0.21	0.18	$CAMC_s$	0.48	0.21
NHD	0.66	0.21	NHD_s	0.81	0.12
SNHD	-0.43	0.51	$SNHD_s$	0.63	0.42
NHDM	0.05	0.16	$NHDM_s$	0.38	0.22

5.2. Metrics Variants

Variants of these cohesion metrics are defined on the basis of the assumption that all the methods of a class by default receive the class type itself (self) as one of the parameter types. $CAMC_s$, NHD_s, $SNHD_s$ and

$NHDM_s$ are defined as variants of CAMC, NHD, SNHD, and NHDM respectively. Cohesion metrics which consider the 'self' parameter are expected to give higher values as the class methods at least agree on one parameter type. It is confirmed by the descriptive analysis of these metrics as shown in Figures 5-8. All the metrics in this category (which consider self parameter type) have higher averages than their counterparts as shown in table 1. It is worth mentioning that $SNHD_s$ takes values in the range from 0 to 1 more frequently, in contrast to SNHD which takes values in the range 0 to -1. It implies that a class whose NHD value is more close to NHD_{max} is more cohesive. Table 1 gives a comparison of average cohesion metrics and their variants with self parameter. The observation is that metric variants, which consider 'self' as one of the parameter types, take values in higher range.

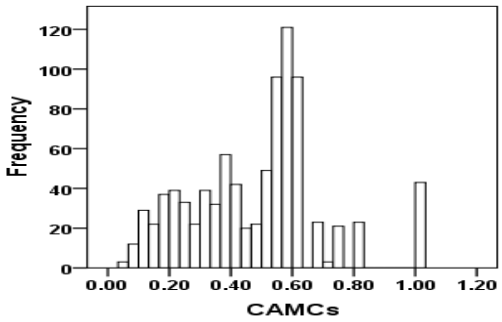

Figure 5. Distribution of $CAMC_s$ metric.

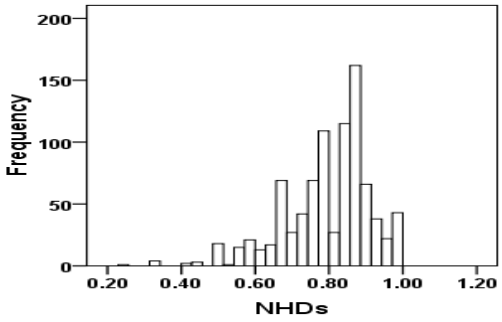

Figure 6. Distribution of NHD_s metric.

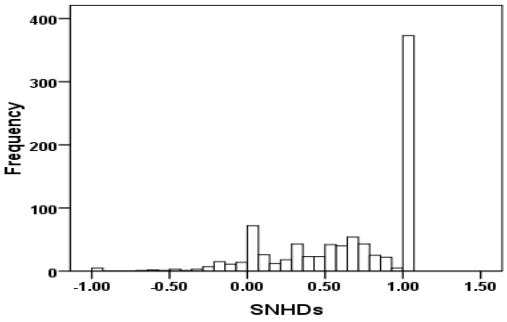

Figure 7. Distribution of $SNHD_s$ metric.

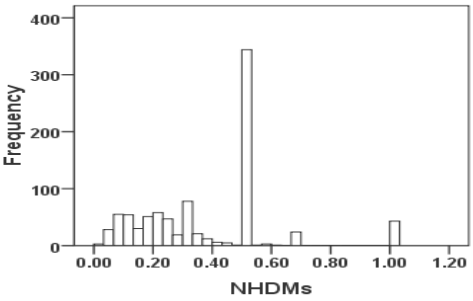

Figure 8. Distribution of $NHDM_s$ metric.

5.3. Size Independence

Figures (9-12) present the relation between cohesion metrics and class size. CAMC takes large values for classes with smaller number of methods. NHD takes large values for classes with larger number of methods. This is in line with the earlier findings about these two metrics [12]. However, if size of the Parameter Occurrence (PO) matrix is taken into consideration then it is found that it does not have significant correlation with any of the metrics (see Table 2). Here *l* represents the number of parameter types, k is the number of methods of the class, and *lk* is the size of the parameter occurrence matrix. This result is unlike the previous studies on these metrics[12, 15]. SNHD is zero for small classes. For large classes, SNHD lies in range [-1,0].

Table 2. Correlation in cohesion metrics and size.

	CAMC	$CAMC_s$	NHD	NHD_s	SNHD	$SNHD_s$	NHDM	$NHDM_s$
l	-.222	-.696	.337	.003	-.356	-.539	-.128	-.645
k	-.307	-.520	.350	.219	-.071	-.179	-.177	-.429
lk	-.158	-.357	.210	.138	.067	-.223	-.075	-.298

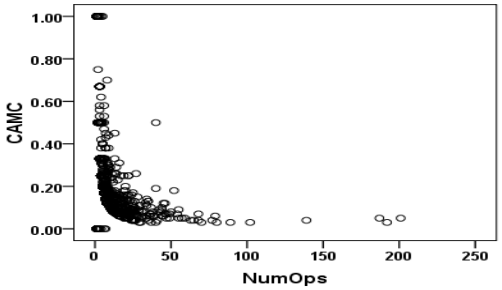

Figure 9. Scatter diagram of CAMC and class size.

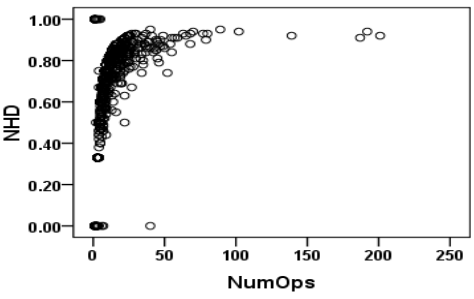

Figure 10. Scatter diagram of NHD and class size.

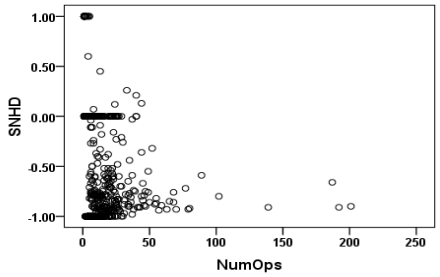

Figure 11. Scatter diagram of SNHD and class size.

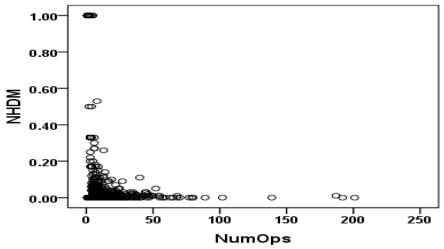

Figure 12. Scatter diagram of NHDM and class size.

NHDM takes values near 0 for most of the classes. However small classes have metric value in the higher range.

5.4. Metrics Inter- Dependencies

The parametric Pearson correlation among the cohesion metrics is given in Table 3. All the correlation figures are significant at the p = 0.01 level. Metric variants, with class itself as its method's parameter types, such as $CAMC_s$, NHD_s, $SNHD_s$, and $NHDM_s$ are moderately correlated with their counterparts. NHD and NHD_s have the highest correlation coefficient in this category. NHDM and CAMC are strongly correlated. Similar is the case for their variants $NHDM_s$ and $CAMC_s$. SNHD is moderately correlated with $NHDM_s$ and $CAMC_s$. Unlike the previous studies, the correlation analysis for this data set does not show any significant correlation in NHD and CAMC [12, 15]. However the scatter plot of values for these two metrics shows a negative trend. CAMC and NHD show a negative relationship in the scatter diagram given in Figure 13. CAMC is very low for the classes for which NHD is very high. On average the NHD metric takes values in higher range. This implies that this metric pair does not have a linear co variation.

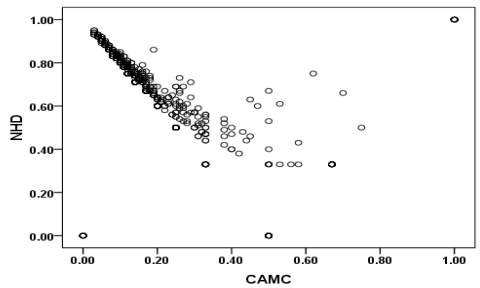

Figure 13. Scatter diagram shows correlation in CAMC and NHD metrics.

Table 3. Correlation values for cohesion metrics.

	CAMC	$CAMC_s$	NHD	NHD_s	SNHD	$SNHD_s$	NHDM
$CAMC_s$	0.575						
NHD	-0.267	-0.542					
NHD_s	-0.372	-0.024	0.623				
SNHD	0.341	0.654	-0.043	0.334			
$SNHD_s$	-0.253	0.347	0.271	0.678	0.478		
NHDM	0.854	0.466	0.122	0.107	0.403	-0.043	
$NHDM_s$	0.456	0.962	-0.356	0.249	0.726	0.520	0.480

Principal Component Analysis (PCA) is used to identify the metrics measuring orthogonal dimensions. Rotated principal components are obtained using the varimax rotation technique. Three principal components are extracted which capture 93.28% of the data set variance as shown in Table 4. Metrics with significant loading coefficients in a particular dimension are highlighted in bold. An analysis of the table shows that $NHDM_s$ and CAMCs and SNHD contribute significantly to the first dimension: PC1. $SNHD_s$ is moderately significant in two dimensions: PC1 and PC2. NHD and NHD_s both load significantly on PC2. NHDM and CAMC both load significantly on PC3.

Table 4. Principal components matrix.

	PC1	PC2	PC3
Eigen Value	3.72	2.39	1.35
Percent	46.45	29.93	16.90
Comm. percent	46.45	76.38	93.28
CAMC	0.25	-0.32	0.90
$CAMC_s$	0.91	-0.27	0.30
NHD	-0.39	0.89	0.11
NHD_s	0.27	0.90	-0.13
SNHD	0.84	0.21	0.27
$SNHD_s$	0.66	0.59	-0.31
NHDM	0.24	0.15	0.95
$NHDM_s$	0.95	-0.01	0.25

It is worth mentioning here that NHDM and $NHDM_s$ have the maximum variance among all the metrics in this analysis. So metrics measuring different dimensions are:

PC1: $NHDM_s$, $CAMC_s$
PC2: NHD, NHD_s
PC3: NHDM, CAMC.

6. Conclusions

This paper investigates design level cohesion metrics such as CAMC, NHD, SNHD, and NHDM using empirical data. NHDM is a modified version of the NHD metric. It is defined so as to avoid the anamolies present in the defintions of other metrics. Statistical analysis of the metrics data shows that CAMC and NHD are influenced by the size of the class (measured in terms of number of methods). However, none of the studied metrics correlates with the size of the parameter occurence matrix of the class. Principal

component analysis of the data shows that NHDM and CAMC both give similar results, but NHDM has more variation in its values. Similar, is the case for NHDMs and $CAMC_s$. SNHD or $SNHD_s$ does not contribute significantly to any dimension. NHD and NHD_s are not significantly related to any of the other metrics.

In future, we plan to include SCOM metric in the study. It will be interesting to study these metrics along with the source code metrics to study the behaviour of these metrics.

References

[1] Badri L. and Badri M., "A Proposal of a New Class Cohesion Criterion: An Empirical Study," *Computer Journal of Object Technology*, vol. 3, no. 4, pp. 145-159, 2004.

[2] Bansiya J. and Davis C., "A Hierarchical Model for Object Oriented Quality Assessment," *Computer Journal of IEEE Transactions on Software Engineering*, vol. 28, no.1, pp. 4-17, 2002.

[3] Bansiya J., Etzkorn L., Davis C., and Li W., "A Class Cohesion Metric for Object Oriented Designs," *Computer Journal of Object Oriented Programming*, vol. 11, no. 8, pp 47-52, 1999.

[4] Bieman J. and Kang B., "Cohesion and Reuse in an Object-Oriented System," *in Proceedings of the Symposium on Software Reusability, ACM Press*, USA, pp. 259-262, 1995.

[5] Bonja C. and Kidanmariam E., "Metrics for Class Cohesion and Similarity between Methods," *in Proceedings of the 44th annual Southeast Regional Conference*, New York, pp. 91-95, 2006.

[6] Briand L., Daly J., and Wust J., "A Unified Framework for Cohesion Measurement in Object-Oriented Systems," *Empirical Software. Enggnering*, vol. 3, no. 1, pp. 65-117, 1998.

[7] Briand L., Wust J., Daly J., and Porter D., "Exploring the Relationships Between Design Measures and Software Quality in Object Oriented Systems," *Conference Systems and Software*, vol. 51, no. 3, pp. 245-273, 2000.

[8] Chae H., Kwon Y., and Bae D., "A Cohesion Measure for Object-Oriented Classes," *Computer Journal of Software Practice and Experience*, vol. 30, no. 12, pp. 1405-1431, 2000.

[9] Chen Z., Zhou Y., and Xu B., "A Novel Approach to Measuring Class Cohesion Based on Dependence Analysis," *in Proceedings of the International Conference on Software Maintenance*, USA, pp. 377-384, 2002.

[10] Chidamber P. and Kemerer C., "Towards a Metrics Suite for Object Oriented Design," *in Proceedings of 6th ACM Conference on Object Oriented Programming, Systems, Languages and Applications*, Arizona, pp. 197-211, 1991.

[11] Chidamber S. and Kemerer C., "A Metrics Suite for Object Oriented Design," *Computer Journal of IEEE Transactions on Software Engineering*, vol. 20, no. 6, pp. 476-493, 1994.

[12] Counsell S., Swift S., and Crampton J., "The Interpretation and Utility of Three Cohesion Metrics for Object-Oriented Design," *Computer Journal of ACM Transactions on Software Engineering and Methodology*, vol. 15, no. 2, pp. 123-149, 2006.

[13] Cox G., Etzkorn L., and Hughes W., "Cohesion Metric for Object-Oriented Systems Based on Semantic Closeness from Disambiguity," *Computer Journal of Applied Artificial Intelligence*, vol. 20, no. 5, pp. 419-436, 2006.

[14] Dallal J., "A Design-Based Cohesion Metric for Object-Oriented Classes," *Computer Journal of World Academy of Science, Engineering and Technology*, vol. 34, pp. 301-306, 2007.

[15] Dallal J. and Briand L., "An Object-Oriented High-Level Design-Based Class Cohesion Metric," *Technical Report*, Simula Research Laboratory, 2009.

[16] Fernández L. and Peña R., "A Sensitive Metric of Class Cohesion," *International Journal of Information Theories and Applications*, vol. 13, no. 1, pp. 82-91, 2006.

[17] Gui G. and Scott D., "Measuring Software Component Reusability by Coupling and Cohesion Metrics," *Journal of Computers*, vol. 4, no. 9, pp 797-805, 2009.

[18] Gyimothy T., Ferenc R., and Siket I., "Empirical Validation of Object-Oriented Metrics on Open Source Software for Fault Prediction," *Journal of Computers IEEE Transactions on Software Enineering*, vol. 31, no. 10, pp. 897-910, 2005.

[19] Henderson-Sellers B., Constantine L., and Graham I., "Coupling and Cohesion (Towards a Valid Metrics Suite for Object-Oriented Analysis and Design)," *Object Oriented Systems*, vol. 3, no. 3, pp. 143-158, 1996.

[20] Hitz M. and Montazeri B., "Measuring Coupling and Cohesion in Object-Oriented Systems," *in Proceedings of International Symosium on Applied Corporate Computing*, pp. 1-10, 1995.

[21] Lee J., Jung S., Kim S., Jang W., and Ham D., "Component Identification Method with Coupling and Cohesion," *in Proceedings of Eighth Asia-Pacific Software Engineering Conference*, USA, pp. 79-86, 2001.

[22] Li W. and Henry S., "Object-Oriented Metrics that Predict Maintainability," *Conference System Software*, vol. 23, no. 2, pp. 111-122, 1993.

[23] Makela S. and Leppanen V., "Client Based Object Oriented Cohesion Metrics," *in Proceedings of 31st Annual International*

Computer Software and Applications Conference, Beijing, pp. 743-748, 2007.

[24] Marchetto A. and Trentini A., "A Framework to Build Quality Models for Web Applications," *Computer Journal of The International Arab Journal of Information Technology*, vol. 4, no. 2, pp. 168-176, 2007.

[25] Marcus M. and Poshyvanyk D., "Using the Conceptual Cohesion of Classes for Fault Prediction in Object-Oriented System," *Computer Journal of IEEE Transactions on Software Engineering*, vol. 34, no. 2, pp. 287-300, 2008.

[26] Marcus A. and Poshyvanyk D., "The Conceptual Cohesion of Classes," *in Proceedings of 21st IEEE International Conference on Software Maintenance*, USA, pp. 133-142, 2005.

[27] Wang J., Zhou Y., Wen L. Chen Y., Lu H., and Xu B., "DMC: A More Precise Cohesion Measure for Classes," *Computer Journal of Information and Software Technology*, vol. 47, no. 3, pp. 167-180, 2005.

[28] Zhou Z. and Leung H., "Empirical Analysis of Object-Oriented Design Metrics for Predicting High and Low Severity Faults," *Computer Journal of IEEE Transactions on Software Engineering*, vol. 32, no. 10, pp 771-789, 2006.

Kuljit Kaur is a lecturer at the Department of Computer Science and Engineering, Guru Nanak Dev University, India. Her areas of interest are software engineering and object oriented analysis and design. Currently, she is working on software component evaluation metrics. She has 14 years of teaching experience. She received an MCA from Guru Nanak Dev University, Amritsar in 1996.

Hardeep Singh is a professor at the Department of Computer Science and Engineering, Guru Nanak Dev University, India. His research interests are information system software metrics. He has 22 years of teaching and research experience. He received his PhD in computer science and engineering from Guru Nanak Dev University, Amritsar in 2003.