Prediction of Reusability of Object Oriented Software Systems using Clustering Approach

Anju Shri, Parvinder S. Sandhu, Vikas Gupta, Sanyam Anand

***Abstract*—**In literature, there are metrics for identifying the quality of reusable components but the framework that makes use of these metrics to precisely predict reusability of software components is still need to be worked out. These reusability metrics if identified in the design phase or even in the coding phase can help us to reduce the rework by improving quality of reuse of the software component and hence improve the productivity due to probabilistic increase in the reuse level. As CK metric suit is most widely used metrics for extraction of structural features of an object oriented (OO) software; So, in this study, tuned CK metric suit i.e. WMC, DIT, NOC, CBO and LCOM, is used to obtain the structural analysis of OO-based software components. An algorithm has been proposed in which the inputs can be given to K-Means Clustering system in form of tuned values of the OO software component and decision tree is formed for the 10-fold cross validation of data to evaluate the in terms of linguistic reusability value of the component. The developed reusability model has produced high precision results as desired.

***Keywords*—** CK-Metric, Desicion Tree, Kmeans, Reusability.

I. Introduction

SOFTWARE reusability is an attribute that refers to the expected reuse potential of a software component. A component can be considered an independent replaceable part of the application that provides a clear distinct function. A component can be a coherent package of software that can be independently developed and delivered as a unit, and that offers interfaces by which it can be connected unchanged with other components to compose a larger system [1]. Software reuse not only improves productivity but also has a positive impact on the quality and maintainability of software products [1].

The software industry is moving toward large-scale reuse, resulting in savings of time and money. To develop a new system from scratch is very costly. This has made custom software development very expensive. It is generally assumed that the reuse of existing software will enhance the reliability of a new software application. This concept is almost universally accepted because of the obvious fact that a product will work properly if it has already worked before.

In this paper, tuned CK metric suit i.e. WMC, DIT, NOC, CBO and LCOM, is used to obtain the structural analysis of OO-based software components. An algorithm has been proposed in which the inputs can be given to Kmeans Clustering system in form of tuned values of the OO software component and decision tree is formed for the 10-fold cross validation of data to evaluate the in terms of reusability.

The paper organized as: the second section describes the problem formulation, the third section explains the steps undertaken to achieve the problem solution. Fourth section of this paper illustrates the results taken and finally the conclusions are written in the last section.

II. Problem Formulation

The aim of Metrics is to predict the quality of the software products. Various attributes, which determine the quality of the software, include maintainability, defect density, fault proneness, normalized rework, understandability, reusability etc. The requirement today is to relate the reusability attributes with the metrics and to find how these metrics collectively determine the reusability of the software component. To achieve both the quality and productivity objectives it is always recommended to go for the software reuse that not only saves the time taken to develop the product from scratch but also delivers the almost error free code, as the code is already tested many times during its earlier reuse.

A great deal of research over the past several years has been devoted to the development of methodologies to create reusable software components and component libraries, where there is an additional cost involved to create a reusable component from scratch. That additional cost could be avoided by identifying and extracting reusable components from the already developed large inventory of existing systems. But the issue of how to identify good reusable components from existing systems has remained relatively unexplored. Our approach, for identification and evaluation of reusable software, is based on software models and metrics. As the exact relationship between the attributes of the reusability is difficult to establish so a Clustering based approach is experimented to discover whether a clustering based model could serve as an economical, automatic tool to generate reusability ranking of a object oriented software.

III. Proposed Methodology

Reusability evaluation System for function Based Software Components can be framed using following steps:

Anju Shri is doing her Masters from Computer Science & Engineering Department, Rayat Institute of Engineering & Information technology, Rail Majra, Punjab, India.

Vikas Gupta is working as Asstt. Prof. at Deptt. of CSE, RIEIT, Rail Majra, Punjab, India.

Sanyam Anand is associated with CEC, Landran, Punjab India

Parvinder S. Sandhu is working as Professor at Computer Science & Engineering Department, Rayat & Bahra Institute of Engineering & Bio-Technology, Sahauran, Distt. Mohali (Punjab)-140104 INDIA

A. Selection & Refinements of Metrics

Selection and refinement of metrics targeting the quality of function based software system and perform parsing of the software system to generate the Meta information related to that Software. The proposed five metrics for Object-Oriented Paradigm are as follows [2] [3] that are tuned as explained in [4] [5] [6] [7]:

A) Weighted Methods per Class (WMC)
B) Depth of Inheritance Tree (DIT)
C) Number of Children (NOC)
D) Coupling Between Object Classes (CBO)
E) Lack of Cohesion in Methods (LCOM)
.

B. Calculate Metric Values

Calculate the metric values of the sampled software components.

C. Perform Clustering

The Clustering is an approach that uses software measurement data for analyzing software quality. In this step, K-Means clustering algorithm is used for partitioning the data into different level of reusability value based on the structural metric values as K-means is the well known approach that classify data into different K groups where K is a positive integer, based on the attributes or some features. Grouping of data is done on the basis of minimizing sum of squares of distances between data and their cluster centroid.

D. Building Decision Tree

Decision tree is generated for extended dataset. As Decision trees are powerful classification methods which often can also easily be understood. The actual type of the tree is determined by the criterion entropy gain. The 10 fold cross validation performance of the proposed approach is evaluated on the basis of the criteria discussed in the next step.

E. Comparison Criteria

The comparisons are made on the basis of the least value of *Accuracy*, *Precision*, *Recall*, and *RMSE* values.

In case of the two-cluster based problem, the confusion matrix has four categories: True positives (TP) are modules correctly classified as faulty modules. False positives (FP) refer to fault-free modules incorrectly labeled as faulty modules. True negatives (TN) correspond to fault-free modules correctly classified as such. Finally, false negatives (FN) refer to faulty modules incorrectly classified as fault-free modules as shown in Table 1.

TABLE 1 CONFUSION MATRIX OF PREDICTION OUTCOMES

	Real Data Value	
Predicted Value	***Fault***	***No fault***
Fault	**TP**	**FP**
No Fault	**FN**	**TN**

With help of the confusion matrix values the precision and recall values are calculated described below:

- *Precision*

The *Precision* is the proportion of the examples which truly have class *x* among all those which were classified as class *x*. The technique having maximum value of probability of detection and lower value of probability of false alarms is chosen as the best fault prediction technique.

Precision for a class is the number of true positives (i.e. the number of items correctly labeled as belonging to the positive class) divided by the total number of elements labeled as belonging to the positive class (i.e. the sum of true positives and false positives, which are items incorrectly labeled as belonging to the class) [8]. The equation is:

$$Precision = TP / (TP + FP) \qquad (1)$$

- *Recall*

Recall in this context is defined as the number of true positives divided by the total number of elements that actually belong to the positive class (i.e. the sum of true positives and false negatives, which are items which were not labeled as belonging to the positive class but should have been) [8]. The *recall* can be calculated as follows:

$$Recall = TP / (TP + FN) \qquad (2)$$

- *Accuracy*

The percentage of the predicted values that match with the expected values of the reusability for the given data.

- *Root mean-squared error*

RMSE is frequently used measure of differences between values predicted by a model or estimator and the values actually observed from the thing being modeled or estimated [9]. It is just the square root of the mean square error as shown in equation given below:

$$RMSE = \sqrt{\frac{(a_1 - c_1)^2 + (a_2 - c_2)^2 + \dots + (a_n - c_n)^2}{n}} \qquad (3)$$

The mean-squared error is one of the most commonly used measures of success for numeric prediction. This value is computed by taking the average of the squared differences between each computed value and its corresponding correct value. The root mean-squared error is simply the square root of the mean-squared-error. The root mean-squared error gives the error value the same dimensionality as the actual and predicted values.

The best system is that having the high *Accuracy,* High *Precision,* High *Recall* and low *RMSE* value.

F. Conclusions Drawn

The conclusions are made on the basis of the comparison made in the previous section.

IV. RESULTS & DISCUSSION

The implementation of algorithm is done in Open Source Tool known as Rapidminer 4.6. The Snapshot of the operator tree or the steps of algorithm followed is shown in figure 1.

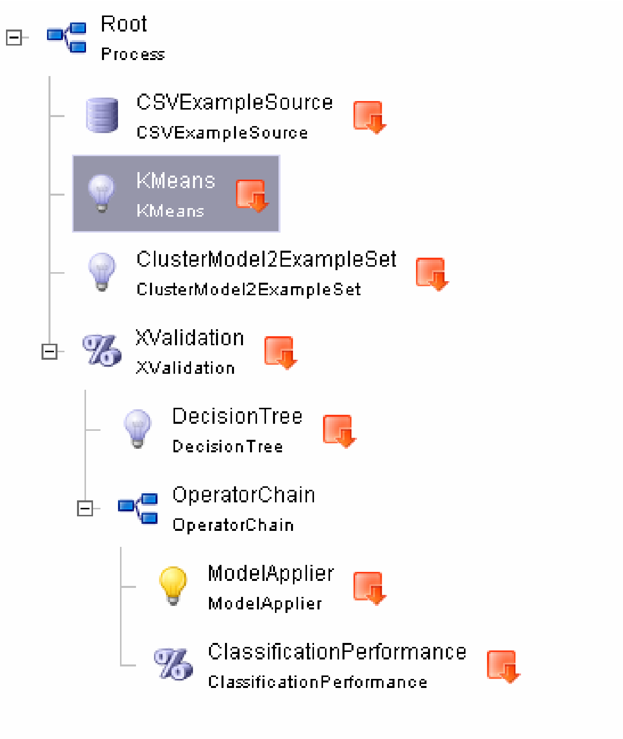
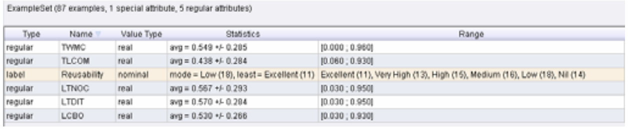


Fig. 1. Snapshot of the Operator Tree Used

First, the dataset file is loaded in the Rapidminer environment. The metadata view of the input dataset is shown in figure 2. In the metadata view type, Name, Value Type, statistics and range of the dataset is calculated. As seen from the figure only reusability value of nominal and all other values are of real type.



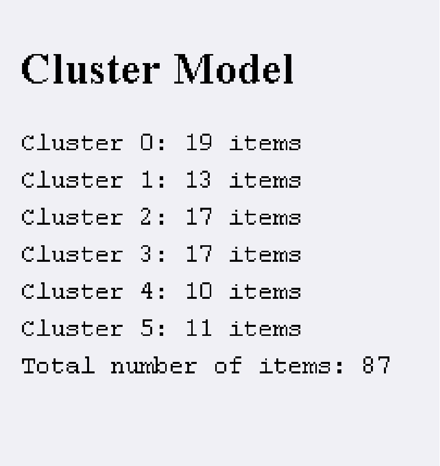
ExampleSet (87 examples, 1 special attribute, 5 regular attributes)

Type	Name	Value Type	Statistics	Range
regular	TWMC	real	avg = 0.549 +/- 0.285	[0.000 ; 0.960]
regular	TLCOM	real	avg = 0.438 +/- 0.284	[0.060 ; 0.930]
label	Reusability	nominal	mode = Low (18), least = Excellent (11)	Excellent (11), Very High (13), High (15), Medium (16), Low (18), Nil (14)
regular	LTNOC	real	avg = 0.567 +/- 0.293	[0.030 ; 0.950]
regular	LTDIT	real	avg = 0.570 +/- 0.284	[0.030 ; 0.950]
regular	LCBO	real	avg = 0.530 +/- 0.266	[0.030 ; 0.930]

Fig. 2. Meta Data View of the Input Dataset

Thereafter, Kmeans clustering algorithm is applied on the dataset. In the Kmeans clustering algorithm the value of K is set to 6 means the the total number of clusters that Kmeans Clustering algorithm is going to generate will be 6. The amximum optimization steps are set to 100 and local random seed value of 1992 is used.

The Text View of the Clusters Assignments is shown in figure 3. The figure 3 shows that there are 19 examples assigned to Cluster 0, 13 examples assigned to Cluster 1, 17 examples assigned to Cluster 2, 17 examples assigned to Cluster 3, 10 examples assigned to Cluster 4 and 11 examples assigned to Cluster 5.

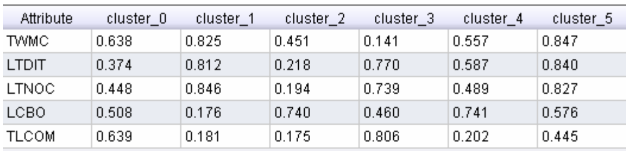


Cluster Model

```
Cluster 0: 19 items
Cluster 1: 13 items
Cluster 2: 17 items
Cluster 3: 17 items
Cluster 4: 10 items
Cluster 5: 11 items
Total number of items: 87
```

Fig. 3.. Text View of the Clusters Assignments

The tabular and graphical plot view of the centriods created is shown figure 4 and figure 5 respectively. The snapshot of the extended reusability dataset is illustrated in figure 6.



Attribute	cluster_0	cluster_1	cluster_2	cluster_3	cluster_4	cluster_5
TWMC	0.638	0.825	0.451	0.141	0.557	0.847
LTDIT	0.374	0.812	0.218	0.770	0.587	0.840
LTNOC	0.448	0.846	0.194	0.739	0.489	0.827
LCBO	0.508	0.176	0.740	0.460	0.741	0.576
TLCOM	0.639	0.181	0.175	0.806	0.202	0.445

Fig. 4. Centriod View of the clusters Created

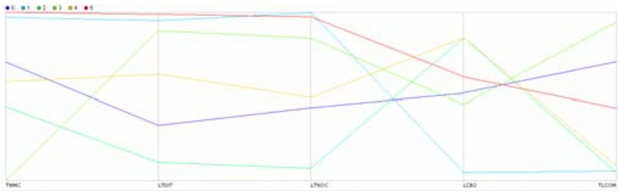
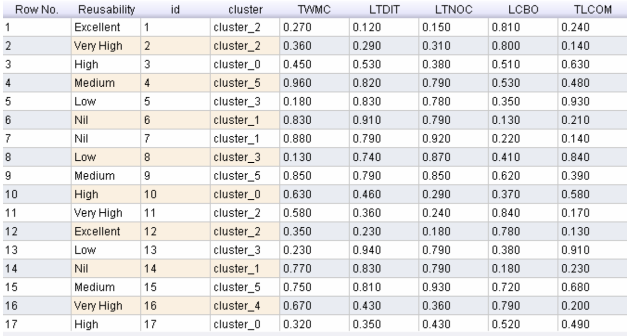


Fig. 5. Centriods' Plot view



Row No.	Reusability	id	cluster	TWMC	LTDIT	LTNOC	LCBO	TLCOM
1	Excellent	1	cluster_2	0.270	0.120	0.150	0.810	0.240
2	Very High	2	cluster_2	0.360	0.290	0.310	0.800	0.140
3	High	3	cluster_0	0.450	0.530	0.380	0.510	0.630
4	Medium	4	cluster_5	0.960	0.820	0.790	0.530	0.480
5	Low	5	cluster_3	0.180	0.830	0.780	0.350	0.930
6	Nil	6	cluster_1	0.830	0.910	0.790	0.130	0.210
7	Nil	7	cluster_1	0.880	0.790	0.920	0.220	0.140
8	Low	8	cluster_3	0.130	0.740	0.870	0.410	0.840
9	Medium	9	cluster_5	0.850	0.790	0.850	0.620	0.390
10	High	10	cluster_0	0.630	0.460	0.290	0.370	0.580
11	Very High	11	cluster_2	0.580	0.360	0.240	0.840	0.170
12	Excellent	12	cluster_2	0.350	0.230	0.180	0.780	0.130
13	Low	13	cluster_3	0.230	0.940	0.790	0.380	0.910
14	Nil	14	cluster_1	0.770	0.830	0.790	0.180	0.230
15	Medium	15	cluster_5	0.750	0.810	0.930	0.720	0.680
16	Very High	16	cluster_4	0.670	0.430	0.360	0.790	0.200
17	High	17	cluster_0	0.320	0.350	0.430	0.520	0.490

Fig. 6. Data View of the extended data

In the next step decision tree with the following parameters is generated:

- Minimum Size of Split : 4
- Minimum Leaf Size : 2
- Minimal Gain : 0.1
- Maximum Depth : 20
- Confidence : 0.25

- Number of Prepruning Alternatives : 3

The generated decision tree is shown in figure 7.

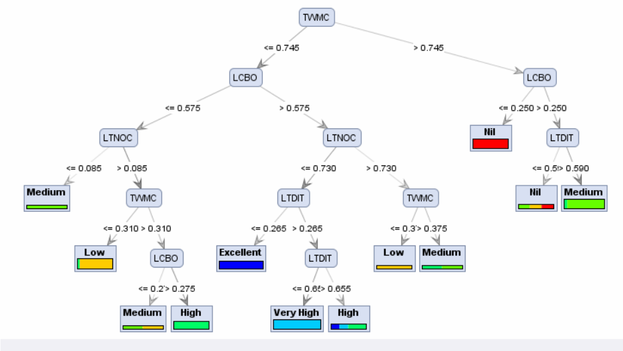
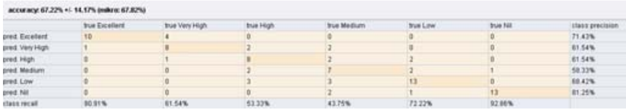


Fig. 7. Decision Tree Generated

The data is evaluated using 10 fold cross validation and the results are calculated in terms of *Accuracy*, *RMSE* and *classification error*. The confusion matrix along with the *precision* and *recall* values of each class is shown in figure 8.



accuracy: 67.22% +/- 14.17% (mikro: 67.82%)

	true Excellent	true Very High	true High	true Medium	true Low	true Nil	class precision
pred. Excellent	10	4	0	0	0	0	71.43%
pred. Very High	1	8	2	2	0	0	61.54%
pred. High	0	1	8	2	2	0	61.54%
pred. Medium	0	0	2	7	2	1	58.33%
pred. Low	0	0	3	3	13	0	68.42%
pred. Nil	0	0	0	2	1	13	81.25%
class recall	90.91%	61.54%	53.33%	43.75%	72.22%	92.86%	

Fig. 8. Confusion Matrix

During the 10 fold cross validation process *accuracy*, *classification error* and *RMSE* values obtained are 67.22% +/- 14.17%, 32.78% +/- 14.17% and 0.522 +/- 0.134 respectively.

V. Conclusion

In this paper, hybrid K-Means and Decision tree approach is used to predict the reusability value of object oriented software components based on the metric values. Tuned CK metric suit i.e. WMC, DIT, NOC, CBO and LCOM, is used to obtain the structural analysis of OO-based software components.

During the 10 fold cross validation process of the developed model *Accuracy*, *Classification Error* and *RMSE* values obtained are 67.22% +/- 14.17%, 32.78% +/- 14.17% and 0.522 +/- 0.134 respectively. When analyzing the results it becomes clear that among all the classes of the reusability Recall value of the ‘Nil’ class is maximum i.e. 92.86% and the Recall value of ‘Excellent’ class is equally good i.e. 90.91%. Similarly, among all the classes of the reusability Precision value of the ‘Nil’ class is maximum i.e. 81.25% and the Precision value of ‘Excellent’ class is second best i.e. 71.43%. It means that the proposed model is able to differentiate between the components with ‘Nil’ reusability value and ‘Excellent’ reusability value very clearly.

Hence, the proposed reusability model has produced high precision and recall results with satisfactory low error values as desired.

References

[1] 1. Gill, Nasib S., “Importance of Software Component Characterization for Better Software Reusability”, ACM SIGSOFT Software Engineering Notes, vol. 31 No. 1, Jan 2006, pp. 1-3.

[2] 2. Chidamber, S.R. and Kemerer, C.F., “A Metric Suite for Object Oriented Design”, IEEE Trans. Software Eng., vol. 20, 1994, pp. 476-493.

[3] 3. Chidamber, S.R. and Kemerer, C.F., “Towards a Metrics Suite for Object Oriented Design”, Proceedings Conference Object Oriented Programming Systems, Languages, and Applications (OOPSLA’91), vol. 26, no. 11, 1991, pp. 197-211.

[4] 4. Parvinder Singh Sandhu and Hardeep Singh, “Software Reusability Model for Procedure Based Domain-Specific Software Components”, International Journal of Software Engineering & Knowledge Engineering (IJSEKE), Vol. 18, No. 7, 2008, pp. 1–19.

[5] 5. Parvinder S. Sandhu, Parwinder Pal Singh,hardeep Singh, "Reusability Evaluation With Machine Learning Techniques", WSEAS TRANSACTIONS on COMPUTERS, issue 9, Volume 6, September 2007, pp. 1065-1076.

[6] 6. Parvinder Singh Sandhu and Hardeep Singh, "A Reusability Evaluation Model for OO-Based Software Components", International Journal of Computer Science, vol. 1, no. 4, 2006, pp. 259-264.

[7] 7. Parvinder Singh Sandhu and Hardeep Singh, "Automatic Quality Appraisal of Domain-Specific Reusable Software Components", Journal of Electronics & Computer Science, vol. 8, no. 1, June 2006, pp. 1-8.

[8] 8. en.wikipedia.org/wiki/Precision_and_recall

[9] 9. Challagulla, V.U.B. , Bastani, F.B. , I-Ling Yen , Paul,(2005) “Empirical assessment of machine learning based software defect prediction techniques”, 10th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems, WORDS 2005, 2-4 Feb 2005, pp. 263-270.