Volume 1 No. 8, December 2011 ISSN-2223-4985
International Journal of Information and Communication Technology Research

http://www.esjournals.org

Code to Design Migration from Structured to Object Oriented Paradigm

[1]Dineshkumar V, [2]Deepika J
[1]Assistant System Engineer,
Tata Consultancy Services, Chennai
[2]Assistant Professor,
Dept. of Computer Science and Engineering,
United Institute of Technology,
Coimbatore-20

ABSTRACT

In software development, programming is considered as an important step because the design created for the software gets into actual form while coding. Intuitively, in the last two decades most software were developed using structure oriented programming. But this scenario deviates in a manner that most of software are being developed based on object oriented paradigm due to certain good features like graphical user interface design, reusability, maintainability, portability, flexibility, scalability etc. Hence this change in technology from structured programming to object oriented programming requires re-engineering and reverse engineering techniques. In this proposed system automated conversion of structured oriented code to object oriented design is attempted using clustering algorithm based on reverse engineering concept. In this approach, initially the relationship between variables and functions of the structured program are identified using Jaccard distance measure that leads to grouping or clustering. Thereby, each cluster is considered to be a class. The individual classes are further processed to obtain object oriented design.

Keywords: *Reverse Engineering, Entity set, UML Class Diagram, Clustering.*

1. INTRODUCTION

Almost all software was developed in console environment in earlier period. In contrast, during recent years all the software is being developed using graphical user interface. As it is being so, companies need to improve their console applications to GUI applications which becomes the need of the hour. Companies willing to extend their projects by including new features are proved to be an easier task in object oriented paradigm than in structure oriented paradigm. Reusability is a major factor that is preferred by software organizations [4]. They always like to use the existing system than building software from scratch. But the existing software is mostly in structure oriented programming which as mentioned earlier is difficult to extend than object oriented code. This necessitates conversion of structure oriented code to object oriented for better maintenance, extendibility and reusability. Reverse engineering afford a best solution for code migration. There are three methods for deriving object oriented code from structure oriented code[4]. 1) Developing from scratch (Greenfield engineering) – manual analysis on the functionalities of existing software and development of a new software from scratch. 2) Re-engineering - a method which derives object oriented code from structure oriented code. 3) Reverse engineering – a method which derives object oriented design from structure oriented code which can be used for implementation purposes.

The problem of developing software from scratch is its high cost. Re engineering needs user interaction to produce accurate results. Reverse engineering is an optimal method for converting structure oriented code as it reduces the need to check code errors, reduces the complexity of adding new features and reduces the cost of conversion. Thus a method is proposed for conversion of structure oriented to object oriented code by means of reverse engineering based on clustering and dependency between program elements.

The better object oriented design representation is UML. Since its introduction, the Unified Modeling Language has attracted many organizations and practitioners. UML is now the de facto modeling language for software development. Several features account for its popularity: it's a standardized notation, rich in expressivity.[13]

Section 2 tells about the related works, discusses about reverse engineering, dependency measure and clustering in reverse engineering. The proposed approach is discussed in Section 3. Results and discussions is given in Section 4 and the future work is discussed in Section 5.

2. RELATED WORK

The term reverse engineering has it's origin in the hardware analysis. Since 1895 the term reverse engineering is on the field of engineering. The reverse engineering helps several software engineering area, in computer hardware re design, electronic chips manufacture, software engineering, quality measurement, etc[1][2].

In software engineering, the main aim of the reverse engineering is to making the code more

understandable for maintenance and reuse of code. [3,17]. In our proposed system we mainly focus on getting design from code, that is Design recovery. There are many tools and approaches are available for reverse engineering. There are two most general approaches used for reverse engineering, first defines a set of diagram that capture a required knowledge by means of knowledge editor and the second one is to automate the design generation from code.[5].

The automatic migration between code to design was introduced by hennery M.Seed in his paper he converts a COBOL program into a object oriented design document. He follows a three step approach, [18]

1. Finding objects- objects are identified on the basics of seven related groups like user interface object, File object, and Record object.
2. Extracting the operations – extract the operations upon the objects.
3. Connecting the object – make the relationship between objects.

There is an entirely different method of re engineering also, Bradley Schmerl, et al [8] produces architecture diagram form a running software system. Terence J. Harmer et al[10] using knowledge based transformation between code to design.

The clustering algorithm is used in software engineering in many places mostly the software re modularization and design recovery. Onaixa Maqbool et al [6] uses hierarchical clustering algorithm for generating architecture diagram from code and explain about the importance of hierarchical clustering algorithms. They introduce some dependency matrices for clustering purpose. Mitchal et al [7] uses clustering algorithm for re modularizes the existing code, here clustering algorithm used for grouping related elements. Ducasse, et al[9] proposes a methodology for finding object oriented class design from object oriented code. Lot of research follows clustering for getting re modularization and design recovery . But no one is to propose any method for converting code to Detailed design, instead all researches getting architecture design from code. [6].

For finding dependents between elements of programs we have lots matrices, which are introduced at deferent times. Jaccard distance metrics, MoJo metric, etc.[15][11] MoJo is used to find the similarities of program elements. Sisimon et al[16] uses jaccard distance method for finding dependencies based on the mapping method. Fokaefs et al[15] find the dependencies using entity set of every elements.

The Clustering algorithm and the dependency between elements work tougher for grouping related elements of a code. Marios Fokaefs et al [14] use a clustering algorithm and dependency metric for re engineering a bad small class. UML Design representation is the best design representation[12].

In the proposed system agglomerative clustering algorithm is used along with jaccard distance measure matrices for grouping the related elements of structured oriented program.

3. PROPOSED METHODOLOGY

Various tasks ranging from software assessment to system re-development employs reverse engineering as a supplementary activity. Code to design is one of the growing and ever green reverse engineering technologies. It is very useful for software maintenance, Re development, identifying the quality of software, Re engineering code [4]. For automating code re engineering there are several techniques available [18]. In our proposed work we introduce a new technique for code to design migration which uses agglomerative clustering algorithm for grouping related elements in the structured oriented program and dependency between elements using Jaccard distance matrices.

3.1 Code to Design without Clustering

An easy way to find the design from structure oriented code is to convert the code into a single object oriented class. The global variables and the functions are the elements of the newly created design. In code to design conversion, the global variables are often considered as ‘attributes’ of new class and ‘functions’ as methods. For better understanding refer Figure2 where the code transforms to design.

Though, this is an easy way to get design, most of the structured programs are big in size with many variables and functions which may lead to bigger class (Bad Smell class). Hence converting them into a single class produces very less cohesion between class elements [14][15]. So the single class conversion is not a novel method.

3.2 Code to Design using Clustering

As mentioned above, single class design conversion is not assessed to be a good idea for reverse engineering. The problem in single class conversion is the generation of ‘low cohesive’ design. A Better way to improve the cohesion is to decompose the single class into more than one class [14]. In our approach, dividing the related global variables and functions into many divisions is performed rather than a single division which eventually increases cohesion. If this division is done randomly, the cohesion between class elements does not increase and also the coupling of the newly generated class will increase. Hence in order to get ‘*low coupling and high cohesive design*’ grouping related global variables and functions are novel thereby making every group as a class.

For such grouping (related attributes and methods) agglomerative clustering algorithm is used. Agglomerative clustering is the best selection of algorithm for clustering the elements [14]. This algorithm works as follows: Firstly, it assigns each element to a single cluster.

Later, after performing every iteration it merges the two closest clusters. Finally, the algorithm terminates when all entities are contained in a single cluster. Eventually, the outcome of the algorithm is the hierarchy of clusters. To determine the actual clusters a threshold value is chosen for finding the minimum distance as a cut-off value. The hierarchy of the clusters is usually represented by a dendrogram. An example of a dendrogram is shown in Figure 1.

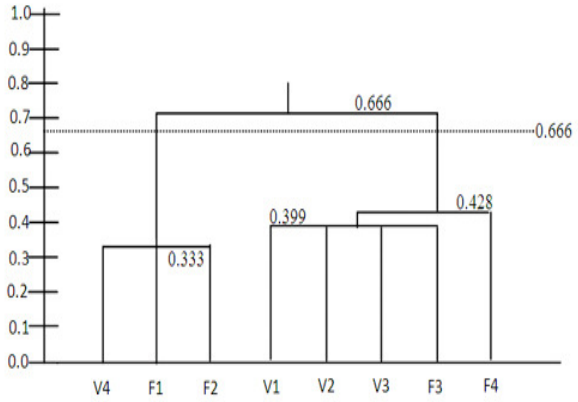

Figure 1: Cluster Dendrogram

For defining the dependencies between the elements of the structure oriented program Jaccard distance metric is used as in equation 1. Where D refers the dependency between two elements, A and B refers the entity set of those elements.

$$D = 1 - \frac{|A \cap B|}{|A \cup B|} \qquad (1)$$

For finding the distance we generate entity set for each elements according to Marios Fokaefs et al[14]. According to their principle, the entity set of a global variable contains the variable itself and the methods which use the global variable. The entity set of methods contains the global variables used by the method, other methods called by the method and the method itself.

Using the entity sets we generate a distance table as shown in Table 1. According to the distance table, the agglomerative clustering algorithm clusters the related global variables and functions into two or more clusters. A dummy name is assigned for each resultant clusters. The individual clusters thus obtained said to possess their global variables and functions which represents class attributes and methods. Each cluster forms a specific class in the UML design. For better understanding consider the following example.

Consider the c program shown in the Figure2. The following steps are used to convert the c program into UML class design.

Step 1: *Finding Global Variables and functions of the structured oriented program*

The main processing elements of a structured programming are the global variables and functions. So the output design will have all of this variables and functions. In our example the variables and functions present in the program are identified.
Global Variables: V1: R ; **V2:**C ; **V3:**T ; **V4**:area
Function: F1: AreaSquare; **F2:** AreaCircul; **F3:** TotalArea; **F4:** Main.

Step 2: *Finding entity set*

For finding the relationship between elements of the structured oriented program the entity set is identified using the above stated methodology. The entity set of the example program elements are as follows
Entity set of **V1**= {V1, F3, F4}
Entity set of **V2** = {V2, F3, F4}
Entity Set of **V3** = {V3, F3,F4}
Entity Set of **V4** = {V4, F1, F2}
Entity Set of **F1** = {F1, V4}
Entity Set of **F2** = {F2, V4}
Entity set of **F3** = {F3, V1,V2,V3,F1,F2}
Entity set of **F4** = {F4, V1, V2, V3, F3}

Step 3: *Finding dependencies between Elements using Jaccard distance metric*

The distance between the global variables and functions of the structured program is found using equation1. The distance measure varies between 0 and 1. If the distance is zero then they are same elements. If the distance is one there is no relationship between the corresponding two elements. The table 1 depicts the distance value among the global variables and functions of the example structured program considered initially.

Table 1 Dependency between structured program elements

	V1	V2	V3	V4	F1	F2	F3	F4
V1	0.0							
V2	0.5	0.0						
V3	0.5	0.5	0.0					
V4	1.0	1.0	1.0	0.0				
F1	1.0	1.0	1.0	0.3	0.0			
F2	1.0	1.0	1.0	0.3	0.6	0.0		
F3	0.7	0.7	0.7	0.7	0.8	0.8	0.0	
F4	0.3	0.3	0.3	1.0	1.0	1.0	0.4	0

Step 4: *Clustering the elements*

Clustering the elements for grouping the related elements and methods is based on the distance table. The agglomerative clustering algorithm is then used to produce clusters containing the related global variables and functions. The cluster results are represented as cluster Dendrogram. Assuming the threshold as 0.666, two

clusters are obtained as follows. [The dendrogram is shown in Figure 1].
Cluster1: V4, F1, F2
Cluster2: V1, V2, V3, F3, F4

Step5: *UML Class diagram Generation*

In this step each cluster is considered as a class. Using the clusters an UML class diagram is drawn as shown in the figure 2.

Evaluation

The cluster result must be efficient it needs to have the following properties

1. Resulting Cluster atleast want to have two or more elements
2. Every cluster atleast want to have one function.

For evaluating result design we use CK metric for cohesion and coupling as follows[19].

LCOM—Lack of Cohesion on Methods: This metric counts the number of different methods within a class that reference a given instance variable. Complexity and design difficulty increase as LCOM increases.

4. RESULTS AND DISCUSSIONS

The above approach was implemented and the results are analysed by the selected experts. The software

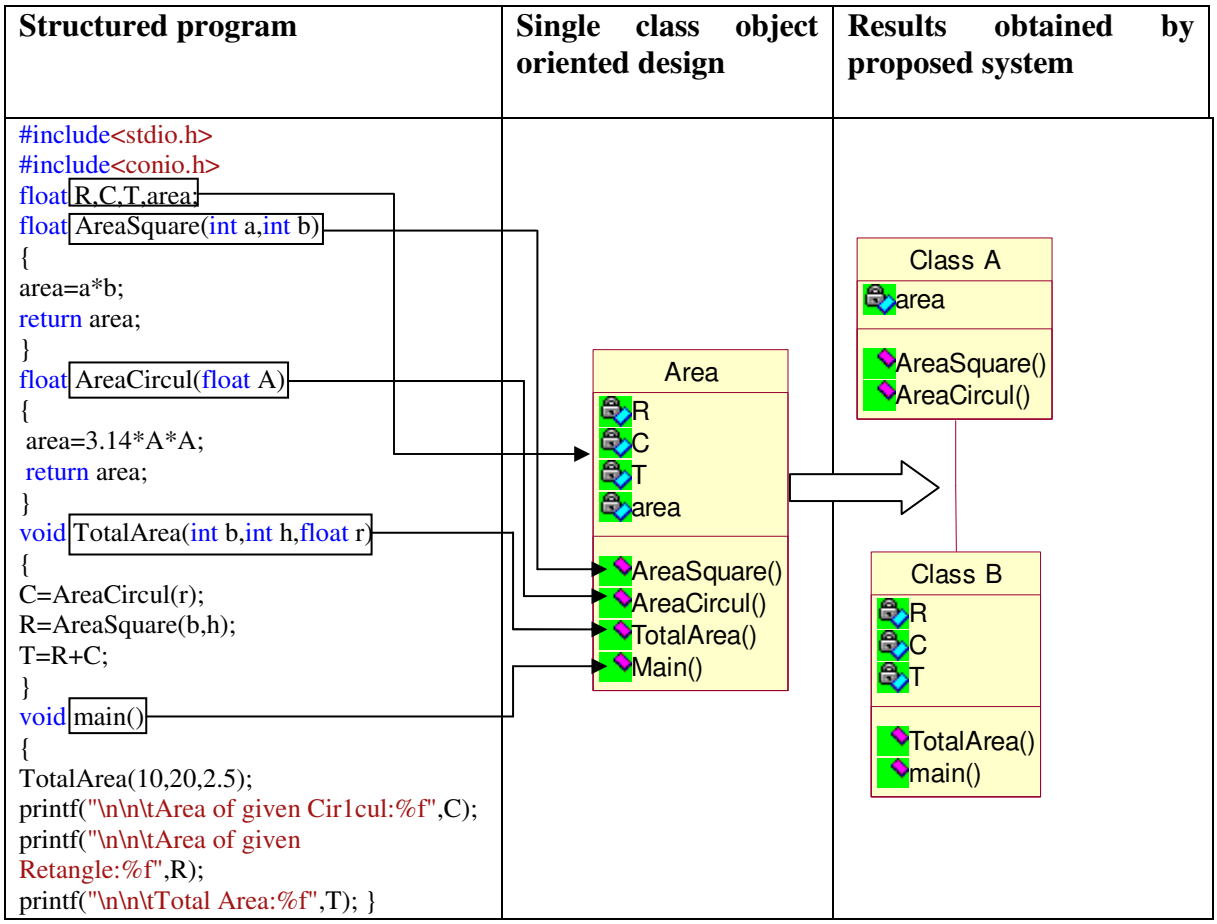

Figure 2: Code to design using clustering

named CUML designer was specially made for C programs. The CUML designer get c program as input and give a design XML as an output. The design XML represents the resultant design textually. This XML file view graphically by XML Viewer. A typical architecture diagram for the proposed methodology is shown in the figure 3.

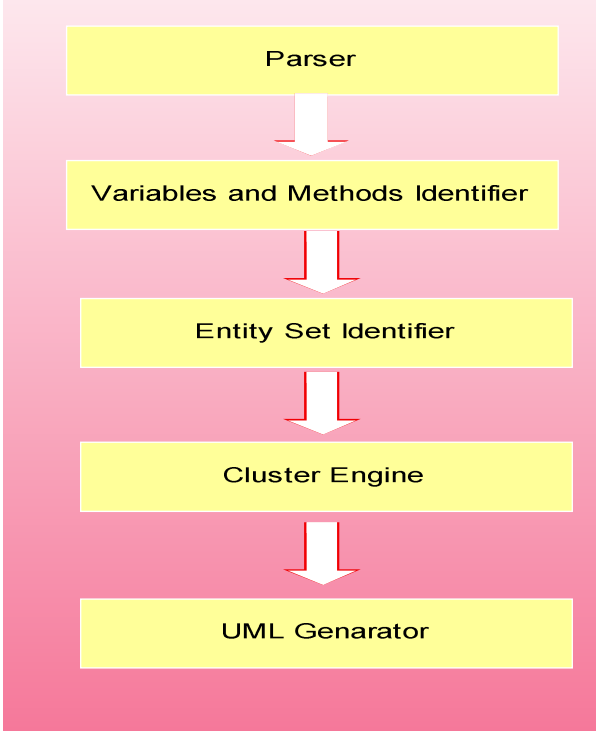

Figure 3: Structured code to UML class design system

The methodology was implemented as five modules. Parser, Element Identifier, Entity set Generator, Distance measurer, Cluster Engine. The 'parser' parses the given c program word by word. The element identifier identifies the variables and functions of the given C program. Entity set generator generates the entity set according to the above approach. 'Distance measurer' measure the dependency between elements of the c program by analyzing the code. The cluster engine then clusters the related elements and methods and then generates an UML representation.

The softwares are given to 50 experts like IT company employees, professors and research students. Finally we give a question sheet that having the following Questions:

1. Is the result UML class diagram useful to understand the code?
2. Do we use the result UML diagram directly as detailed design of the new project?
3. Is the newly generated class having efficient class? (High cohesive an low coupling)
4. Is the newly generated class diagram is useful for Maintainer?

The answers of the following question as shown in table 2.

Table 2: Survey Summary

Question number	Yes	No
1	43	7
2	32	18
3	44	6
4	46	4

From the result we get the result UML diagram as 86% understandable, 64% direct use of design, 88% of high cohesive and low coupling results and it is useful for 92 % of Maintainers. 64% of design direct usage without modifying the result UML class design is very good result. And all other usage like maintainability, class quality, understandability all are above 85% . so it is very good result. For further improvement and for beta testing of this project we posted the code as open source in scourge forge.org as the name CUML designer [21], the screen shot of the CUML as shown figure 4.

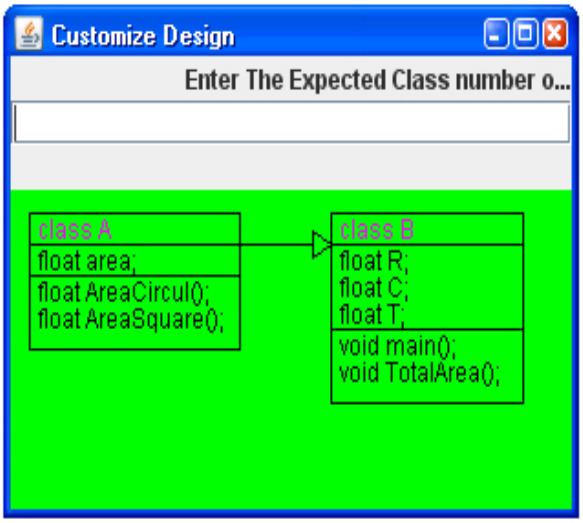

Figure 4: Screen short of CUML

Different author uses different metric for finding dependency between elements of given source and different clustering algorithms for creating architecture from source code. In our proposed system dependency calculated by using entity set and Jaccard distance formula. For clustering the system it uses agglomerative clustering algorithm. All approaches create design from the cluster results. So we compare all cluster results based on the cluster efficiency metric as shown in eq 2.

$$Cluster\ Efficiency = \frac{||EC||}{||C||} - - - -2$$

$$EC = \left\{ x \middle| \begin{array}{c} x \in C \\ \wedge\ ||x|| \geq 2 \\ \wedge\ atleast\ one\ \boldsymbol{a} \in F\ such\ that\ a \in x \end{array} \right\}$$

Where EC- Set of Efficient Clusters

C- Set of Resultant Clusters

$$if\ c1 \in C\ then\ c1 = \{x | x \in F \cap V\}$$

Where F- set of Function from the source

V-set of Global Variables from the source

O. Maqbool et al[20] uses three types of metrics for calculating dependency between elements.

i) *Association coefficients*
Jaccard coefficient J = $a/(a+b+c)$ (3)

ii) *Distance measures*
$D = b + c$ (4)

iii) *Correlation Coefficients*

$$\rho = \frac{ad-bc}{\sqrt{(a+b)(c+d)(a+c)(b+d)}} \quad (5)$$

Where a, b, c and d values find based on O. Maqbool et al[20]. The figure 5 shows the cluster efficiency graph for proposed methods cluster efficiency verses Maqubool's approaches cluster efficiency.

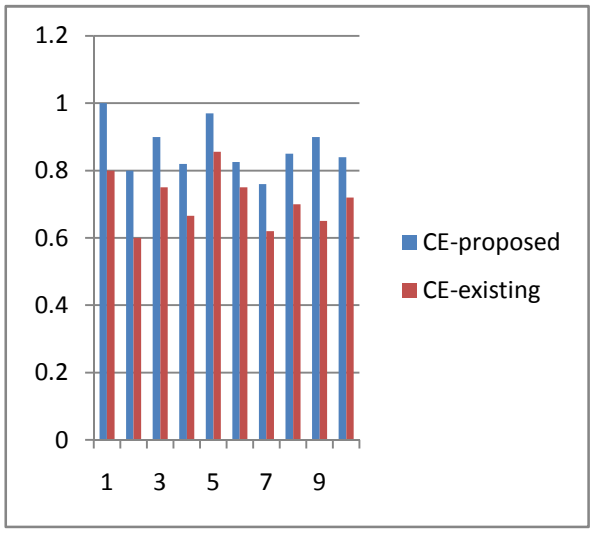

Figure 5: Cluster efficiency graph

5. CONCLUSION AND FUTURE WORK

The proposed system is the one used for the first time in code to design conversion using the clustering algorithm. The existing automated methods are difficult due to high user interaction. The proposed system does not involve any user interaction. The system is fully automated to draw an UML class diagram. Usage of Cluster algorithm with dependency measure makes grouping of related global variables and functions easier. The resultant UML class diagram makes better understandability. The system possess scalability of the UML class design that if any changes are made to the input structured code. Hence, the proposed methodology is said be agreeable compared with other existing systems.

For making the relationship each clustered class little amount of user interaction needed. As spicily c programs having pointers, struct data structure, union data structure, goto statements. Such special characters want to be tackled. Naming the newly generated class is the big issues, the proposed approach using dummy names.

REFERENCES

[1] Lee, J.J. and Strader, N.R. 1987. CMOS ROM arrays programmable by laser beam scanning. IEEE Journal of Solid-State Circuits , vol.22, no.4, pp. 622- 624, Aug 1987.

[2] Stern, R.H. 1989. MicroLaw-protecting hardware against competition by copyrighting it as a compilation of data. Micro, IEEE Journal , vol.9, no.1, pp.2-5, Feb 1989

[3] Biggerstaff, T.J.1989. Design recovery for maintenance and reuse. Computer, IEEE Journal , vol.22, no.7, pp.36-49, Jul 1989.

[4] Chikofsky, E.J. and Cross, J.H. 1990. Reverse engineering and design recovery: a taxonomy. Software, IEEE Journal , vol.7, no.1, pp.13-17, Jan 1990.

[5] Torchiano, M., Ricca, F. and Tonella, P. 2010. Empirical comparison of graphical and annotation-based re-documentation approaches. IET Software/IEE Proceedings, vol.4, no.1, pp.15-31, Feb. 2010.

[6] Maqbool, O. and Babri, H.A. 2007. Hierarchical Clustering for Software Architecture Recovery. IEEE Transactions on Software Engineering, vol.33, no.11, pp.759-780, Nov. 2007.

[7] Mitchell, B.S. and Mancoridis, S. 2006. On the automatic modularization of software systems using the Bunch tool. IEEE Transactions on Software Engineering, vol.32, no.3, pp. 193- 208, March 2006.

[8] Schmerl, B., Aldrich J., Garlan, D., Kazman, R. and Yan, H. 2006. Discovering Architectures from Running Systems. IEEE Transactions on Software Engineering, vol.32, no.7, pp.454-466, July 2006.

[9] Ducasse, S. and Lanza, M. 2005. The class blueprint: visually supporting the understanding of glasses. IEEE Transactions on Software

Engineering, vol.31, no.1, pp. 75- 90, Jan. 2005.pp. 150- 165, Feb. 2005.

[10] T.J. Harmer, P.J. McParland. and J.M. Boyle. 1996. Using Knowledge-Based Transformations to Reverse-Engineer COBOL Programs. Proceeding Conference on Knowledge Based Software Engineering, pp. 114-123, 1996.

[11] Tzerpos, V. and Holt, R.C. 1999. MoJo: a distance metric for software clusterings. Proceedings Sixth Working Conference on Reverse Engineering, vol., no., pp.187-193, 6-8 Oct 1999.

[12] Briand, L.C., Labiche, Y. and Leduc, J.2006. Toward the Reverse Engineering of UML Sequence Diagrams for Distributed Java Software. IEEE Transactions on Software Engineering, vol.32, no.9, pp.642-663, Sept. 2006.

[13] Lange, C.F.J., Chaudron, M.R.V. and Muskens, J. 2006.In practice: UML software architecture and design description. *Software, IEEE* Journal, vol.23, no.2, pp. 40- 46, March-April 2006

[14] Fokaefs, M. , Tsantalis, N. , Chatzigeorgiou, A. and Sander, J. 2009.Decomposing object-oriented class modules using an agglomerative clustering technique. IEEE International Conference on Software Maintenance .ICSM 2009, vol., no., pp.93-101, 20-26 Sept. 2009.

[15] N. Tsantalis and A. Chatzigeorgiou. Identification of Move Method Refactoring Opportunities. IEEE Transactions on Software Engineering, vol. 35, no. 3, pp. 347-367, 2009.

[16] Simon, F.,Steinbruckner, F. and Lewerentz, C. 2001. Metrics based refactoring. Fifth European Conference on Software Maintenance and Reengineering 2001, vol., no., pp.30-38, 2001.

[17] Eleni Stroulia and Tarja systa. 2002. Dynamic Analysis For Reverse Engineering and Program understanding. ACM Sigapp Applied Computing Review , vol. 10, no. 1, pp. 8-17, 2002

[18] Harry M. Sneed and Erika Nydry. 1995. Extracting object-Oriented Specification from Procedurally Qriented Programs. Proceedings of the Second Working Conference on Reverse Engineering WCRE 95.

[19] Linda M. Laird and M. Carol Brennan. 2006. Software Measurement and Estimation A Practical Approach. Published by John Wiley & Sons, Inc., Hoboken, New Jersey, 2006.

[20] O. Maqbool and H.A. Babri,2004. TheWeighted Combined Algorithm: A Linkage Algorithm for Software Clustering" Proc. Eighth European Conf. Software Maintenance and Reeng., pp. 15-24, 2004.

[21] http://cumldesigner.sourceforge.net