IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 4, No. 2, June 2015, pp. 53~61
ISSN: 2252-8938

Design an Algorithm for Software Development in Cbse Environment using Feed Forward Neural Network

Amit Verma, Er. Pardeep kaur
Department Computer Science Engineering, Chandigarh University

Article Info

Article history:

Received Feb 5, 2016
Revised April 8, 2016
Accepted May 9, 2016

Keyword:

Back propagation algorithm
Keyword based retrieval
Component based development
clustering

ABSTRACT

A In software development organizations, Component based Software engineering (CBSE) is emerging paradigm for software development and gained wide acceptance as it often results in increase quality of software product within development time and budget. In component reusability, main challenges are the right component identification from large repositories at right time. The major objective of this work is to provide efficient algorithm for storage and effective retrieval of components using neural network and parameters based on user choice through clustering. This research paper aims to propose an algorithm that provides error free and automatic process (for retrieval of the components) while reuse of the component. In this algorithm, keywords (or components) are extracted from software document, after by applying k mean clustering algorithm. Then weights assigned to those keywords based on their frequency and after assigning weights, ANN predicts whether correct weight is assigned to keywords (or components) or not, otherwise it back propagates in to initial step (re-assign the weights). In last, store those all keywords into Repositories for effective retrieval. Proposed algorithm is very effective in the error correction and detection with user base choice while choice of component for reusability for efficient retrieval is there. To check the results of our algorithm based on factors like accuracy, precision and recall compare with existing technique i.e. integrated classification scheme for retrieval of components based on keyword search and results are so encouraging.

Corresponding Author:

Amit Verma,
Department Computer Science Engineering,
Chandigarh University.

1. INTRODUCTION

In 21th century current trend of automotive technologies emerging at very high pace. During the past forty years, various approaches for software development come into existence. From the last few years, the basic approach or method used by developer is to separate the software into phases and work according to those phases [1], so that they can concentrate only one phase at a time. The first software development approach after software crisis comes into existence in 1970s. On the basis of different needs of customers and organization's targets, developers refer different approaches for development. In software development, mostly challenges faced by project management and project lead like deadlines of project, extra resources needed, over budget etc. These all issues sometimes occur from project complexity include size, less knowledge etc. To overcome these issues, component based development come into existence in 1990's [8]. Through this method, organizations can develop their software by selecting appropriate Commercial-Off-the-shelf (COTS) components and assemble them. These COTS components develop by different developers by using different platforms and languages. Component can be some code, utility functions or programming unit and component can be product specific, domain specific and domain independent. The aim of CBSE is to achieve multiple quality objectives such as reusability, interoperability, implementation transparency. In

whole process, main emphasis is on reuse a component, clustering [22] and retrieval are two important parts of component based development. Clustering [32] is form of unsupervised learning, is the process of partitioning a set of data into a set of meaningful sub classes called cluster. It is basically organizing data into groups based on their similarity. Clustering is widely used in economic science, Pattern recognition, spatial data analysis, image processing etc. In software reusability, the first and foremost fundamental problem is locating and retrieving right component from large repository [30]. Retrieval of component should be efficient and time consuming. Mainly for reuse a component, developers have to store their relevant component into repository. Various clustering algorithms like K mean [14], K-mode [13], Genetic Algorithm [21] etc. applied on repository to make cluster on based of behavioral, structural or functional attributes. Then next step is to retrieve a component by applying various component retrieval techniques like Keyword based method, Syntax based, Semantic based and Genetic based optimization method [38] etc. In last, they may be apply metrics like cyclometric complexity, Halstead metrics, regularity metrics etc on structural attributes and results feed up into neural network's algorithms [4]. The field of neural networks has a history of some five decades but has found application only in the past fifteen years, and the field is still developing continuously. Neural network is able to use some hidden unknown information in the data. This process of capturing hidden information is called learning or training network. We can train a neural network to perform a particular function by adjusting the values of the connections (weights) between elements. Various algorithms are Back Propagation algorithm [36], self organizing map algorithm and various conjugate gradient algorithms [4] like Fletcher-reeves, Polak Ribiere, Powell Beale and Scaled conjugate gradient algorithm. In back propagation algorithm adjusts the weights in the steepest descent direction (negative of the gradient) and calculates mean square error (MSE) that shows the difference between resulted output and target output. This research paper aims to propose an algorithm that provides error free and automatic process (for retrieval of the components) while reuse of the component. The major objective of this work is to provide efficient algorithm for storage and effective retrieval of components using neural network and parameters based on user choice through clustering.

Many techniques and technologies have been proposed and evaluated but so far none of methodology has focused on error correction and detection with user base choice while choice of component for reusability for efficient retrieval, repositories must be well separated (boundaries should be defined through parameter listing) related to the components. This serves as a pivot point for the proposed methodology.

The paper's organization is as follows:

1. Section 2 describes Problem statement
2. Section 3 describes the proposed algorithm for component reusability
3. Section 4 describes implementation work
4. Section 5 describes the Experimental results
5. Section 6 covers the conclusion and Future scope

2. CURRENT AREAS TO BE WORK UPON:

The problem addressed in this paper can be formally stated as: There is set of components SC and components have their own different nature representing FR (Functional Requirements) and NFR (Non Functional Requirements). Each component covers some requirements R. We have to store components into their respective Repositories by passing through Multilayer Feed Forward Neural Network (MLFFNN). There is dire need to manage repository through clustering and making the process automatic. An error correction and detection are the key issues with automation that can overcome using neural network techniques.

Need and Significance: For reusability of components, Component identification or retrieval of exact component is not an easy task in CBSE. Designing a system by reusing existing components leads to faster time to market. However, finding the appropriate component that satisfies the set of requirements is becoming the most difficult and challenging task. There is dire need to manage repositories, So that it could be easy to search component that covers all requirements in a manner that maximizes the quality of product.

3. ALGORITHM

In software development organizations, concept of reuse a component is important to maintain reliability or to increase a quality and a level of productivity. Reuse a part of software is proved to be very beneficial to overcome the challenges occur in software development, is over budget, time complexity, deadlines of complex project etc. It takes large time to implement software from scratch. On that time

component based development must be very helpful. Component is independent part perform complete functionalities. Component Based software engineering is used to develop or assemble software from existing components. For the successes of software project, there is dire need to make the process better and component reusability is optimal solution. For this purpose, Component storage and retrieval are two key issues. In the concept of reusability, efficient component storage and retrieval is challenging task because whole successes of software depend upon these key aspects. Our proposed algorithm must prove to be very beneficial for this purpose that show, how we can effectively manage the repositories through clustering and make the process automatic (for retrieval of components).

Table 1. Pseudo Code of Algorithm

Pseudo code of algorithm	
Initialize T_i	// Test document
For (T_i=1, $T_i < T_n$, i++)	
{	
Perform C_R	// Retrieval of component
}	
$S_R \rightarrow F_R$ and $N_{SR} \rightarrow NF_R$	// F_R- Functional requirement and NF_R- Non functional Requirement
	// C1 and C2 two clusters
$C_1 \rightarrow F_R$	
$C_2 \rightarrow NF_R$	// Word count of Functional Requirement
$F_R \rightarrow W_{C (FR}$	// Word count of non functional Requirement
$NF_R \rightarrow W_{C (NFR}$	
	// Data dictionary
Create DD;	
For each DD_i (i=0,…….,n)	
$DD_0 \rightarrow W_{C(FR)}$	// Store word counts (keywords) into data dictionary
$DD_1 \rightarrow W_{C(NFR)}$	
	// for word count
where $Wc_{(FR)} \neq Wc_{(NFR)}$	
	// Assign weights according to highest value of keyword and so on
$Wi \rightarrow W_{HC(FR}$	
	// weights decrease as word count decrease
W_i-- ;	
Networkweight ← initializeWeights(word, value)	// assign weights to all word count (keywords)
For (i=1 to n)	
If	
{	
$W_i = W_i (W_{HC(FR)})$	// check Neural network Algorithm, Assign first weight or priority to highest word count
{	
Else	
{	
BackPropagateError(Re- initialize weights, word, value)	
}	
}	
}	
Create R	// create Repositories
$R \rightarrow K_{(FR)}$	// Store keywords into repositories
$R \rightarrow K_{(NFR)}$	
$F_K \rightarrow R$	// Fetch keywords from repository to retrieve document

4. IMPLEMENTATION OF ALGORITHM

Input: Software document, (Software Requirement Specification), e.g. Hotel Management System

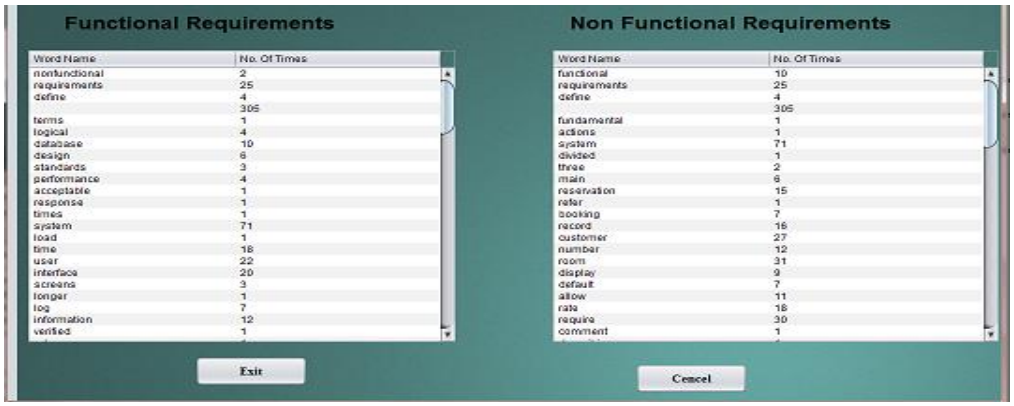

Figure 1. Create two Clusters of Functional and Non Functional Requirements through k mean Algorithm and Show Word Counts of Keywords

Figure 2 Assign Weights to Keywords, According their Word Count Value and Show Working of Back Propagation Algorithm

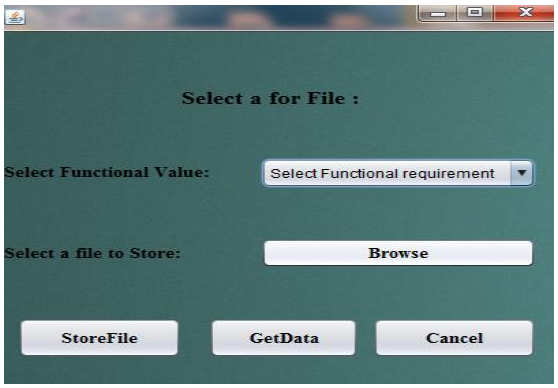

Figure 3. Repository of Functional Requirement

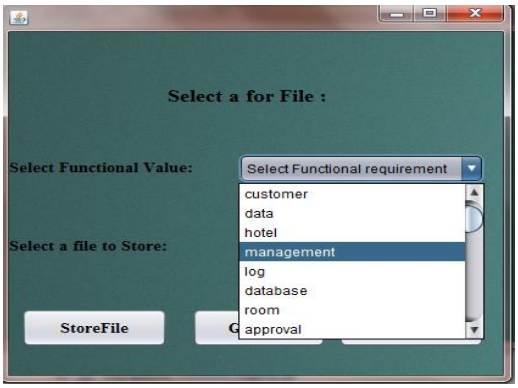

Figure 4. Select Keyword from Drop Down list

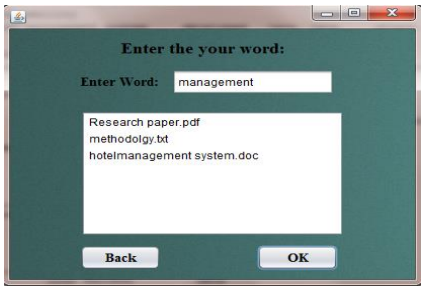

Figure 5. show the Selected Document Based

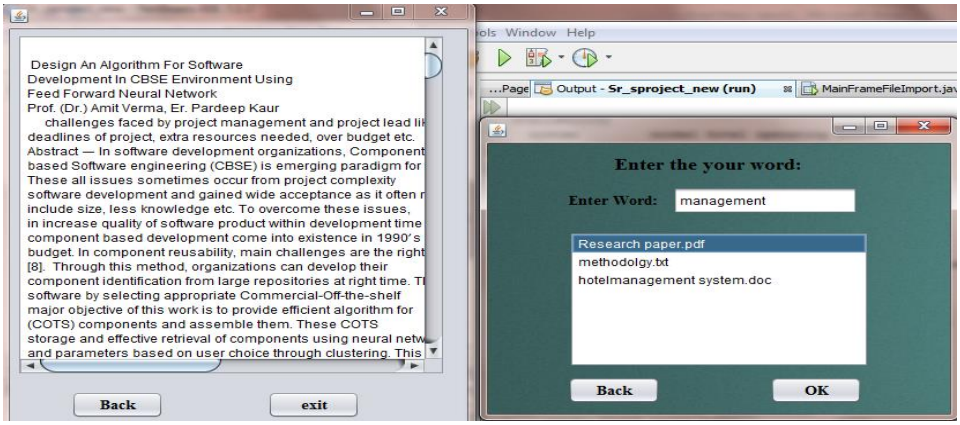

Figure 6. When one clicks on any file to show, it on entering keyword

5. EXPERIMENTAL RESULTS

This section describes the detail of experimental results of our document retrieval process with factors, according to proposed algorithm. It also provides us the comparison of our proposed idea with the existing technique of integrated classification scheme for effective retrieval of components based on keywords.

Table 2. Show Comparisons of Proposed Technique with Existing Retrieval Technique

S. NO.	Techniques	Name of component or keyword	Relevant Component	Accuracy	Precision values	Recall values
1.	Proposed approach	Reservation	1	0.72	0.03	0.96
		System	2	0.12	0.06	0.93
		Display	3	0.17	0.09	0.91
		Document	4	0.23	0.12	0.88
2.	Integrated classification scheme	Search, C	3	0.35	0.08	0.42
		Sort, C	5	0.7	0.14	0.71
		C, linux	5	0.10	0.22	0.67
		Education ,C	13	0.12	0.37	0.46

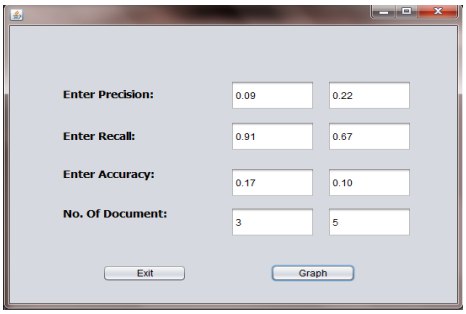

Figure 7. Shows values of all factors and click on graph button to show graphs

In this Figure 7, enter the values of all factors shown above. When someone fetch keyword from repository to search the document, then these factors values based on keyword shown on console and in second column enter the factors values of previous technique with which we compare our results.

5.1. Comparison of Results based on Accuracy

In this Figure 8, it shows the accuracy of our retrieval process of component (or keyword) from repository**.** We compare our algorithm's efficiency with the existing technique e.g. Integrated classification scheme**.** In this scheme, two keywords were integrated in particular iteration for searching components from repository. The accuracy of this component retrieval method is not better than our proposed method for effective retrieval of components.

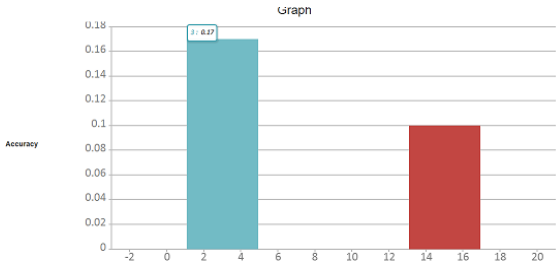

Figure 8. The Accuracy of our Retrieval Process of Component (or keyword) from Fepository

5.2. Comparison of Results based on Recall

In this Figure 9, it shows the precision factor of our retrieval process of component (or keyword) from repository**.** We compare our algorithm's efficiency with the existing technique e.g. Integrated classification scheme**.** In precision factor, it tells us the ratio of retrieval of relevant components from irrelevant components and total no. of components. The accurate value of precision is between 0 and 1. The precision of our retrieval process of components comes very accurate between o and 1 and less than previous results.

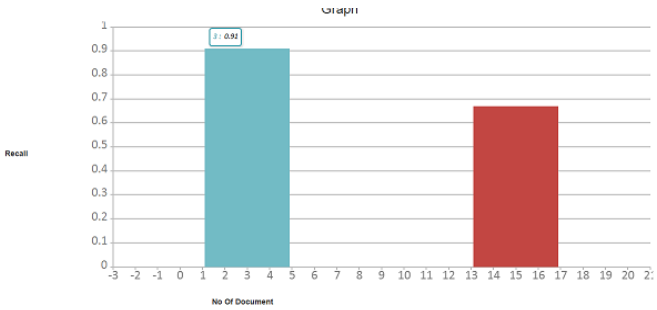

Figure 9. The Precision Factor of our Retrieval Process of Component (or keyword) from Repository

5.3. Comparison of Results based on Precision

In this Figure 10, it shows the recall factor of our retrieval process of component (or keyword) from repository**.** We compare our algorithm's efficiency with the existing technique e.g. Integrated classification scheme**.** In recall factor, it tells us the ratio of retrieval of relevant components from total no. of components. The accurate value of recall is between 0 and 1. The recall of our retrieval process of components comes very accurate between o and 1 and greater than previous results.

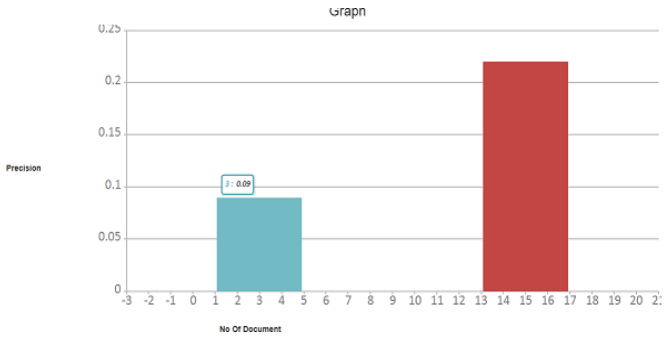

Figure 10. Recall Factor of Our Retrieval Process of Component (or keyword) from Repository

6. CONCLUSION AND FUTURE SCOPE

This study empirically validates the concept of reuse in software development. Reuse a part or component of software may overcome the various challenges exist in development of software. In this research paper, propose an algorithm that provides error free and automatic process (for retrieval of the components) while reuse of the component. It also provides the efficient storage method of components by passing through neural network and parameters based on user choice through clustering. In last, retrieval must be done by using keyword based retrieval method. Proposed algorithm is very effective in the error correction and detection with user base choice while choice of component for reusability for efficient retrieval is there. This algorithm also serves better decision making approach and best optimal component will be selected.

6.1. Future scope

In the Future Scope, text document can be used for the effective component (or keywords) storage and retrieval method for reusability in development of software. Other can take cost and time attributes into consideration for optimal component selection and for effective retrieval process of components along with the test or text documents. In future, the approach of effective storage and retrieval of components can be embedded with other allied technologies such as Matlab, PHP and Python. Implementation can be carried out with the help of various simulators such as NS2, NS3 and with the help of microprocessors for interrupt generation in case wrong keyword selected or path of document file will not be correct.

REFERENCES

[1] X Winston W Royce. Managing the development of large software systems. *IEEE*, 1970: 1-9.
[2] William HE Day, Herbert Edelsbrunner. Efficient Algorithms for Agglomerative Hierarchical Clustering Method. *Journal of Classification*. 1984: 7-24.
[3] James C Bezdek, Robert Ehrlich, William Full. FCM: The Fuzzy C-Means Clustering Algorithm. *Computers & Geosciences*. 1984; 10(2-3): 191-203.
[4] Martin Fodslette Mollar. A scaled conjugate gradient algorithm for fast supervised learning. University of Aarhus, 1993; 6: 525-533.
[5] Daniel Svozil, Vladimir Kvasnieka, Jie Pospichal. Introduction to multi-layer feed-forward neural networks. *ELSEVIER*, 1997: 43-62.
[6] Johannes Sametinger. Software Engineering with Reusable Components. *Springer-Verlag*, 1997: 1-272.
[7] Vu Tran' Dar-Biau Liu. Component-based Systems Development: Challenges and Lessons Learned. *IEEE*, 1997: 452-462.
[8] Brown, AW, Wallnau, KC. The current state of CBSE. *IEEE software*. 1998; 15(5): 37-46.
[9] Marie Chavent. A monothetic clustering method. Pattern Recognition Letters 19, 1998: 989-996.
[10] A.k Jain, MN Murthy, PJ Flynn. Data Clustering: A Review. ACM Computing Surveys, 1999; 31(3).

[11] Takashi Kobayashi and Motoshi Saeki. Software Development Based on Software Pattern Evolution. *IEEE*, 1999: 18-25.
[12] Bogdan Gabrys, Andrzej Bargiela. General Fuzzy Min-Max Neural Network for Clustering and Classification. *IEEE Transactions on Neural Networks.* 2000; 11(3): 769-783.
[13] Anil chutervedi, Kraft foods, Paul E. Green. K modes clustering. *Journal of classification*. 2001: 35-55.
[14] Tapas Kanungo, David M. Mount, Nathan S Netanyahu, Christine D Piatko, Ruth Silverman, Angela Y Wu. An Efficient k-Means Clustering Algorithm: Analysis and Implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 2002; 24(7): 881-892.
[15] Mark Crowne. Why software product start-ups fail and what to do about it. *IEEE*, 2002: 338-343.
[16] Vitharana, P. Risks and challenges of component-based software development. *Communications of the ACM*. 2003; 46(8): 67-72.
[17] Ming-Chuan Hung, Jungpin Wu, Jin-Hua Chang, Don-Lin Yang. An Efficient k-Means Clustering Algorithm Using Simple Partitioning. *Journal of Information Science and Engineering*. 2005; 21: 1157-1177.
[18] Richard W Selby. Enabling Reuse-Based Software Development of Large-Scale Systems. *IEEE*. 2005: 495-510.
[19] Hans-Hermann Bock. Origins and extensions of the k-means algorithm in cluster analysis. *Electronic Journal for History of Probability and Statistics*, 2008; 4(2): 1-18.
[20] Abbas Y Al Bayati, Najmaddin A Sulaiman, Gulnar W. Sadiq. Modified Conjugate Gradient Formula for Back Propagation Neural Network Algorithm. *Journal of Computer Sciences.* 2009; 5(11): 849-856.
[21] Dixit A, Saxena PC. *Software Component Retrieval Using Genetic Algorithms*. International Conference on Computer and Automation Engineering, IEEE. 2009; 151-15.
[22] Gholam Reza Shahmohammadia, Saeed Jalili, Seyed Mohammad Hossein Hasheminejad. Identification of System Software Components Using Clustering Approach. *Journal of object Technology*, 2010; 9(6): 77-98.
[23] S Ajitha, TV Suresh Kumar, D Evangelin Geetha, K Rajnikanth. *Performance Prediction In Early Stages Of Software Systems: Artificial Neural Network Model.* ICCCCT, 2010: 743-747.
[24] Arvinder Kaur, Kulvinder Singh Mann. Component Selection for Component based Software Engineering. *International Journal of Computer Applications*, 2010; 2(1): 109-114.
[25] Sonia Manhas, Parvinder S Sandhu, Vinay Chopra, Nirvair Neeru. Identification of Reusable Software Modules in Function Oriented Software Systems using Neural Network Based Technique. World Academy of Science, Engineering and Technology. 2010; 4: 07-23.
[26] Fuyuan Cao, Jiye Liang, Deyu Li, Liang Bai, Chuangyin Dang. A dissimilarity measure for the k-Modes clustering algorithm. *Elsevier.* 2011: 120-127.
[27] Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. 2011: 1-29.
[28] Harpreet Singh, Vishal Kumar Toora. Neuro Fuzzy Logic Model for Component Based Software Engineering. *An International Journal of Engineering Sciences*. 2011; 1: 303-314.
[29] Zhen-Guo Che, Tzu-An Chiang, Zhen-Hua Che. Feed Forward Neural Networks Training: A comparison between Genetic algorithm and Back propagation learning Algorithm. *International Journal of Innovative Computing, Information and Control ICIC International*. 2011; 7(10): 5839-5850.
[30] Vaneet Kaur, Shivani Goel. Facets of Software Component Repository. *International Journal on Computer Science and Engineering.* 2011; 3(6): 2473-2476.
[31] Rachna Ratra, Navneet Singh Randhawa, Parneet Kaur, Gurdev Singh. Early Prediction of Fault Prone Modules using Clustering Based vs. Neural Network Approach in Software Systems. *International Journal of Electronics & Communication Technology,* 2011; 2(4): 47-50.
[32] N Md Jubair Basha, Chandra Mohan. A strategy to identify components using clustering approach for component reusability. *Computer Science & Information Technology*. 2012; 397–406.
[33] Anupama Kaur. Impact of Training Function Based Neural Network on Reusable Software Modules. *International Journal of Computer Science and Information Technologies,* 2012; 3: 4024 – 4027.
[34] N Koteswara Rao, G Sridhar Reddy. Discovery of Preliminary Centroids Using Improved K- Means Clustering Algorithm. *International Journal of Computer Science and Information Technologies.* 2012; 3: 4558-1561.
[35] Suresh Chand Gupta, Ashok Kumar. A Neural Network based Method to Optimize the Software Component Searching Results in K-Model. *International Journal of Computer Applications.* 2013; 72(7): 20-27.
[36] Kuldip Vora, Shruti Yagnik. A Survey on Back propagation Algorithms for Feed forward Neural Networks. *International Journal of Engineering Development and Research*. 2013: 193-197.
[37] Sandeep Kumar Jain, Manu Pratap Singh. Estimation for Faults Prediction from Component Based Software Design using Feed Forward Neural Networks. *International Journal of Advanced Research in Computer and Communication Engineering*, 2013; 2(7): 2631-2660.
[38] Kammna Mahajan, Mandeep kaur. Component Retrieval Using Genetic Algorithm Based Optimization Technique. *International journal of computer science and Technology*. 2013; 4(2) 585-587.
[39] Mahsa Hasani Sadi, Eric Yu. Analyzing the Evolution of Software Development: From Creative Chaos to Software Ecosystems. *IEEE*, 2014:.1-11.
[40] Xiaohui Cui, Thomas E Potok. Document Clustering Analysis Based on Hybrid PSO+K-means Algorithm. *Applied Software Engineering Research Group, Computational Sciences and Engineering Division*. 1-8.
[41] Tong Gao, Hui Ma, I Ling Yen, Latifur Khan, Farokh Bastani. A Repository for Component-Based Embedded Software Development. 1-37.

[42] Parminder Kaur, Jarnail Singh, Hardeep Singh. Component Selection Repository with Risk Identification. *IEEE*, 2014: 524-531.

[43] Swati Thakral, Shraddha Sagar, Vinay. Reusability in Component Based Software Development - A Review. *World Applied Sciences Journal*. 2014; 31(12): 2068-2072.

[44] Adnan Khan, Khalid Khan, Muhammad Amir, MNA Khan. A Component-Based Framework for Software Reusability. *International Journal of Software Engineering and Its Applications*. Doi.10.14257/ijseia.2014.8.10.02. 2014; 8(10): 13-24.

[45] Inderjit singh, Ashima Singh. *A Survey on various Component Repositories with detail study of different Methods of Storage and Extraction of Components*. IEEE, International Conference on Advances in Electronics, Computers and Communications. 2014: 1-6.