International Journal of Research in Engineering and Innovation (IJREI)
journal home page: http://www.ijrei.com
ISSN (Online): 2456-6934

A new approach for component retrieval from reuse repository using R-tool

Dr. Jasmine K.S[1], Ravi Teja[2]

[1]Department of MCA, R.V.College of Engineering, Bangalore, Karnataka, India
[2] Software Engineer, Aricent, Gurgaon, Haryana, India

Abstract

A continuing challenge for software designers is to develop efficient and cost-effective software implementations. Many see software reuse as a potential solution; however, the cost of reuse tends to outweigh the potential benefits. The costs of software reuse include establishing and maintaining a library of reusable components, searching for applicable components to be reused in a design, as well as adapting components toward a proper implementation. The study of storage and retrieval methods of software assets in software libraries gives rise to a number of paradoxes: While this subject has been under investigation for nearly two decades, it still remains an active area of research in software reuse and software engineering. The new technologies such as the internet, the world wide web and object oriented programming keep opening new opportunities for better asset packaging, better library organizations, and larger scale libraries – thereby posing new technical challenges. Also, while many sophisticated solutions have been proposed to this problem, the state of the practice in software reuse is characterized by the use of adhoc approaches. This can be explained by the observation that most existing solutions are either too ineffective to be useful or too intractable to be usable. Finally, it is difficult to imagine a successful software reuse program without a sophisticated, systematic procedure for software component storage and retrieval. In this context, a new method is suggested here for component classification and retrieval which makes use of R-Tool, an open source tool for statistical computing and graphics.

Keywords: Software reuse, Component Retrieval, R-Tool

1. Introduction

Many software organizations realized that developing the software using reusable components could dramatically reduce development effort, cost and accelerate delivery. But the non-existence of a standard searching technique for finding the suitable component and also the lack of appropriate tool in this field contributed towards the large-scale failures in their approach. From the past studies on this field, it is found that researchers are tried with different approaches to improve the adaptability of the component but very few studies had taken place in improving the efficiency of component retrieval. Fuzzy linguistic approach is familiar in the information retrieval process. Vector Space model approach is also widely accepted for document retrieval process [5]. Due to the semantic sensitivity and poor representation of large documents, it was not a promising approach to achieve its intended goal. In order to overcome these limitations, R-tool is used by making use of its statistical and analytical capability which exploits semantic connections between documents and parts thereof, and semantic connections between queries and documents. The paper is structured as follows. Section 1 starts with a discussion of what is meant by software reuse and a reusable component. Section 2 talks about present scenario in the component retrieval process. Section 3 presents methodology adopted. Section 4 mentions some concluding remarks and directions for future research in this area.

1.1. What is Software Reuse?

Software reuse at its most basic level consists of making use of any existing information, component or product when designing and implementing a new system or product.
There are differing opinions as to which activities constitute genuine software reuse. Replication of an entire software program does not count as reuse. Reuse of assets is dependent upon both similarities and differences between the applications in which the component is being used [2].

Many organizations already practice a limited form of reuse, for example, most developers have libraries of components that they have developed in previous projects, or they use standard libraries, which are available with many programming languages [5]. About 30% of the cases, it is a very ad-hoc method of reuse, and it will work very well on a small scale and it will not be suitable for entire organizations [6]. Instead, businesses need to implement a systematic reuse program in order to gain the full advantages of reuse.

1.2 What Can Be Reused?

The definition of a reusable component is "any component that is specifically developed to be used, and is actually used, in more than one context" [4]. This does not just include code; other products from the system lifecycle can also be reused, such as specifications and designs, and even requirements on occasion [3]. 'Components' in this case can be taken to include all potentially reusable products of the system lifecycle, including code, documentation, design, requirements etc.
There are various criteria that should be satisfied in order for an asset to be successfully reusable. These are grouped into general, functional and technical requirements [14]. General requirements focus on aspects such as compliance with relevant standards, completeness, modularity and simplicity. All components should conform to the general requirements. Functional requirements include such concerns as which business processes it will simulate or automate, and how well it does this. Functional requirements mainly concern Vertical or Domain-specific assets and tend to be very specific to each information domain. Lastly, technical requirements refer to criteria such as interoperability, portability, communication, security etc [2].
There are different levels of reuse, which can be considered [3]. At the highest level, entire applications can be reused on different platforms provided they are portable. Sub-systems can be reused within different applications, possibly within different domains; for example, a login system could be used in a database application as well as a control application. At a lower level modules or objects can be reused, and at the very lowest level single functions can be reused. This is also known as classification of the granularity of components. Fine grained is used to describe those smaller and more generic components, for example file access functions, or I/O functions. Course grained is used for the more complex components, for example user-interface packages [5]. Reusable assets can be also being built in-house, retrieved from legacy systems or can be bought from an external source [4].

2. Present scenario in the component retrieval process

Existing approaches to software component retrieval process cover a wide spectrum of component encoding methods and search or matching algorithms. The encoding methods differ with respect to their soundness, completeness, and the extent to which they support an estimate of the effort it takes to modify a component. Text-based encoding and retrieval is neither sound nor complete. Its disadvantages have been thoroughly in the information retrieval literature. Lexical descriptor-based encoding approach also suffers from a number of problems about developing and using classification vocabulary [11]. Software specific challenges include the fact that one-word or one-phrase abstractions are hard to come by in the software domain [12]. From the user's point of view, lack of familiarity with the vocabulary is also pointed out as draw back in using a component retrieval system effectively [14]. Vector space Model is also not a promising solution for component retrieval process due to its semantic sensitivity [7]. In this context, content based indexing schemata and ranking based on frequency of occurrence of component search with keywords will be a promising approach for establishing semantic connections between queries and documents.

3. Methods used

3.1. Proposed Document Retrieval Approach

R is the powerful open-source implementation of the language S. R is a very effective statistical tool and well worth the effort to learn. R is polymorphic, which means that the same function can be applied to different types of objects, with results tailored to the different object types [1].
In the proposed approach, the outstanding statistical and graphical capabilities of R is used to make a frequency count of the word search (based on the software component) and a visualization mechanism is followed which facilitates easy understanding of the method adopted.

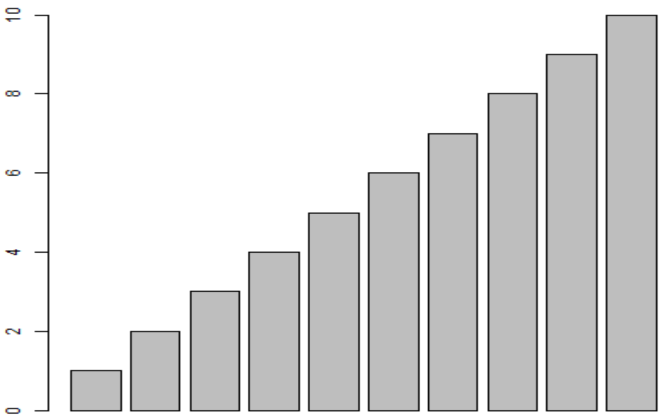


Figure 1: R visualization with simple data

R can import data from CSV files, Excel, SAS and produces the output in pdf, jpg, png formats and also table output.

Steps Adopted

Step: 1

Consider three files as input and stored as .csv files namely cc1.csv, cc2.csv, and cc3.csv (Assumption: searching for a document which addresses the 'Cloud Computing' concept in the reuse repository)

Step: 2

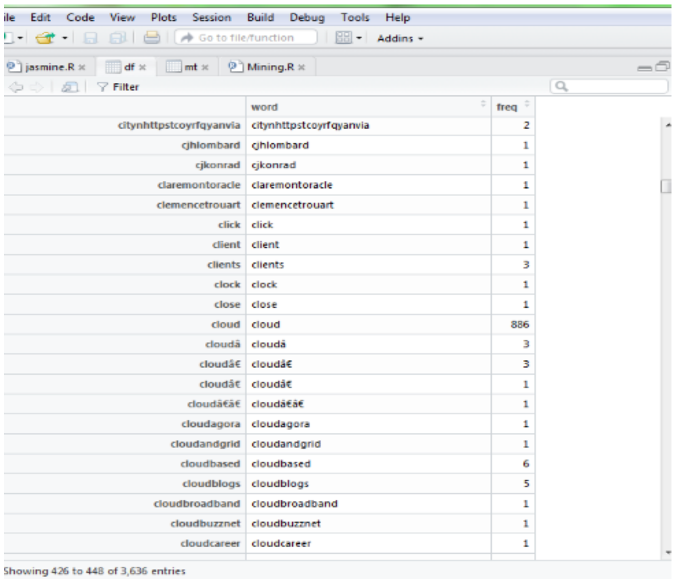
Plot the graphs which shows the frequency of occurrence of all words for the three files separately, (cc1Rplot, cc2Rplot, cc3rplot)

Step: 3

Consider also the complete word 'Cloud Computing ' and find the frequency of it.
(Actually, the words are arranged in ascending order from a to z).

Step: 4

Consider the output of the Search (the file which contains maximum count of the word 'Cloud computing') as the best result.



	word	freq
citynhttpstcoyrfqyanvia	citynhttpstcoyrfqyanvia	2
cjhlombard	cjhlombard	1
cjkonrad	cjkonrad	1
claremontoracle	claremontoracle	1
clemencetrouart	clemencetrouart	1
click	click	1
client	client	1
clients	clients	3
clock	clock	1
close	close	1
cloud	cloud	886
cloudã	cloudã	3
cloudâ€	cloudâ€	3
cloudâ€	cloudâ€	1
cloudâ€â€	cloudâ€â€	1
cloudagora	cloudagora	1
cloudandgrid	cloudandgrid	1
cloudbased	cloudbased	6
cloudblogs	cloudblogs	5
cloudbroadband	cloudbroadband	1
cloudbuzznet	cloudbuzznet	1
cloudcareer	cloudcareer	1

Showing 426 to 448 of 3,636 entries

Figure 2: Tool generated Word frequency table from cc1.csv file

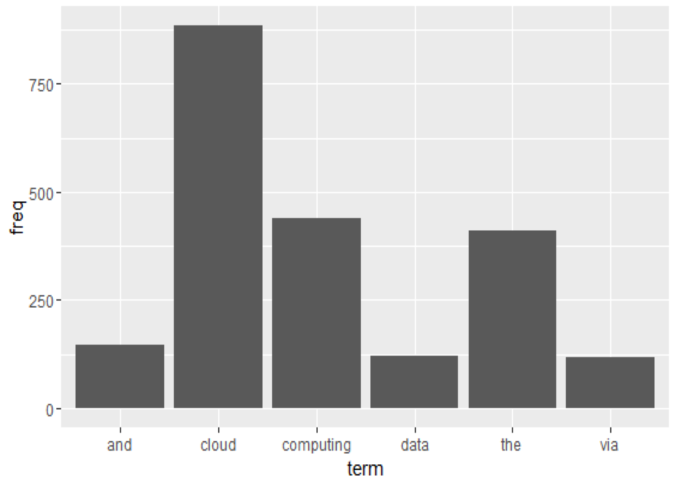
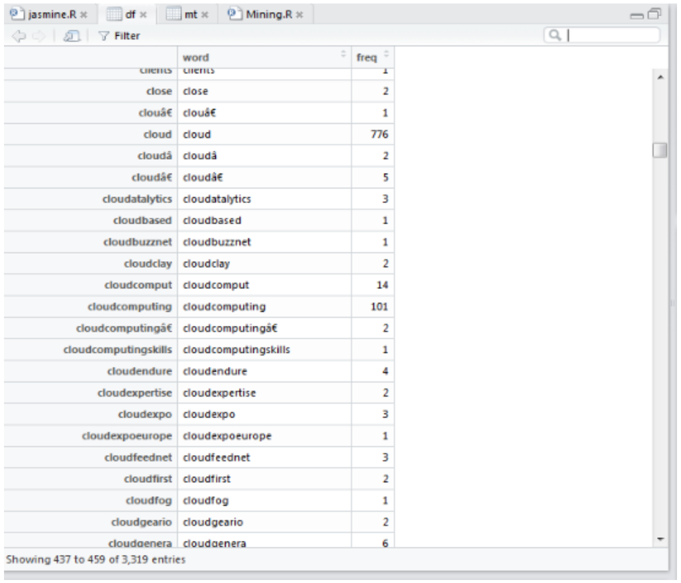


Figure 3: Histogram of dataset1



	word	freq
clients	clients	1
close	close	2
clouâ€	clouâ€	1
cloud	cloud	776
cloudâ	cloudâ	2
cloudâ€	cloudâ€	5
cloudatalytics	cloudatalytics	3
cloudbased	cloudbased	1
cloudbuzznet	cloudbuzznet	1
cloudclay	cloudclay	2
cloudcomput	cloudcomput	14
cloudcomputing	cloudcomputing	101
cloudcomputingâ€	cloudcomputingâ€	2
cloudcomputingskills	cloudcomputingskills	1
cloudendure	cloudendure	4
cloudexpertise	cloudexpertise	2
cloudexpo	cloudexpo	3
cloudexpoeurope	cloudexpoeurope	1
cloudfeednet	cloudfeednet	3
cloudfirst	cloudfirst	2
cloudfog	cloudfog	1
cloudgeario	cloudgeario	2
cloudgenera	cloudgenera	6

Showing 437 to 459 of 3,319 entries

Figure 4: Tool generated Word frequency table from cc2.csv file

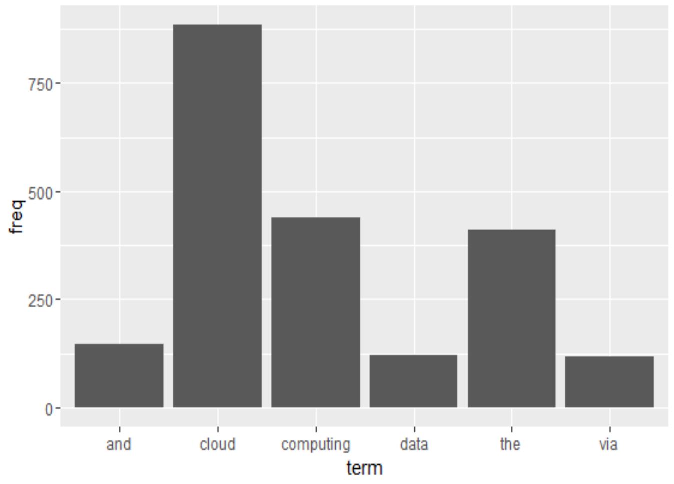


Figure 5: Histogram of datadet2

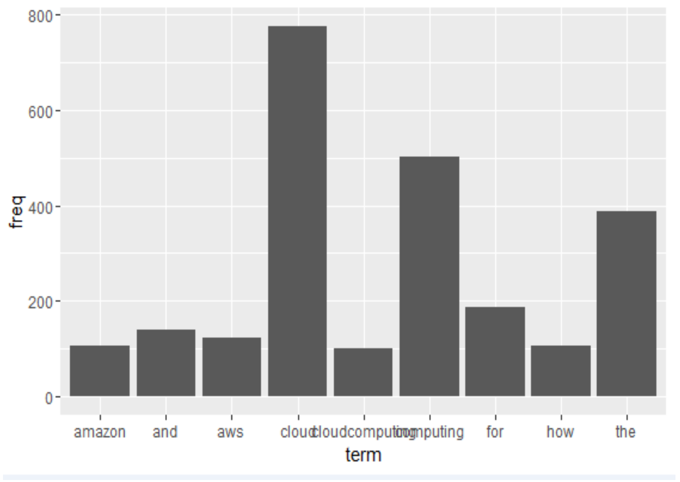
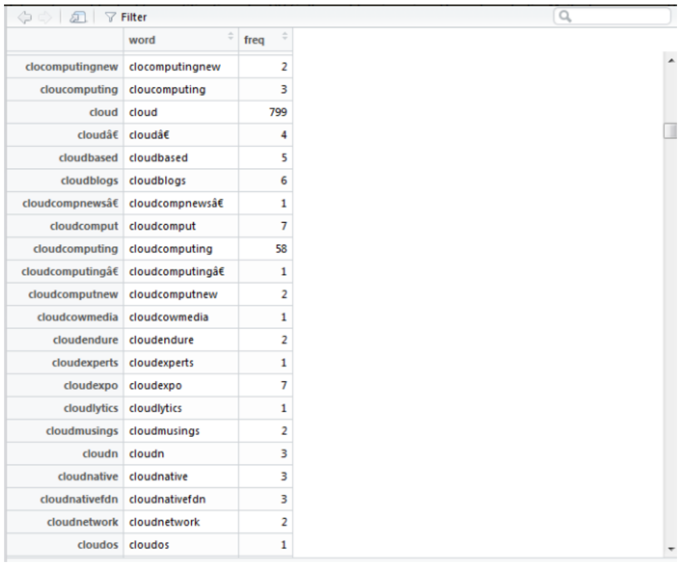


Figure 6: CC2RPlot



	word	freq
clocomputingnew	clocomputingnew	2
cloucomputing	cloucomputing	3
cloud	cloud	799
cloudâ€	cloudâ€	4
cloudbased	cloudbased	5
cloudblogs	cloudblogs	6
cloudcompnewsâ€	cloudcompnewsâ€	1
cloudcomput	cloudcomput	7
cloudcomputing	cloudcomputing	58
cloudcomputingâ€	cloudcomputingâ€	1
cloudcomputnew	cloudcomputnew	2
cloudcowmedia	cloudcowmedia	1
cloudendure	cloudendure	2
cloudexperts	cloudexperts	1
cloudexpo	cloudexpo	7
cloudlytics	cloudlytics	1
cloudmusings	cloudmusings	2
cloudn	cloudn	3
cloudnative	cloudnative	3
cloudnativefdn	cloudnativefdn	3
cloudnetwork	cloudnetwork	2
cloudos	cloudos	1

Figure 7: Tool Generated Word frequenncy table cc3.csv

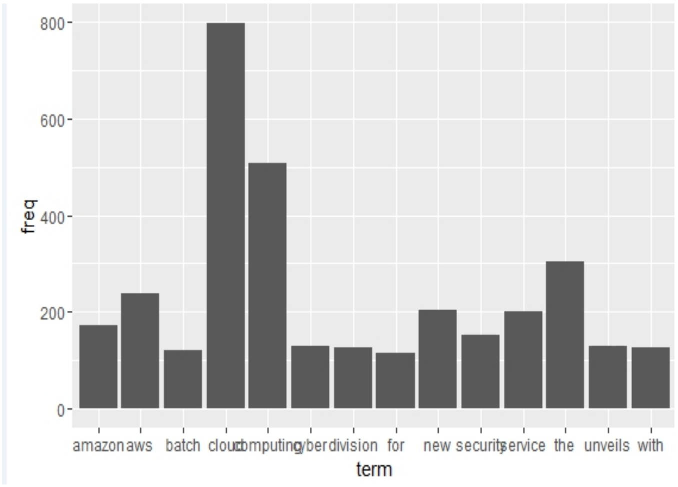
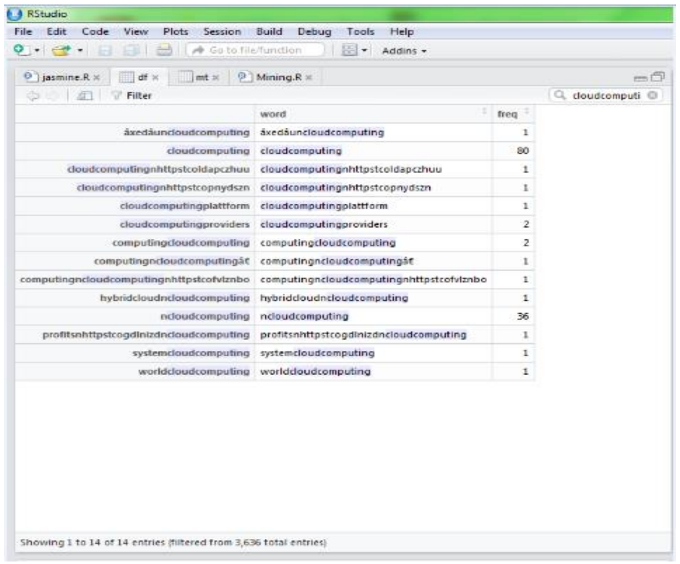


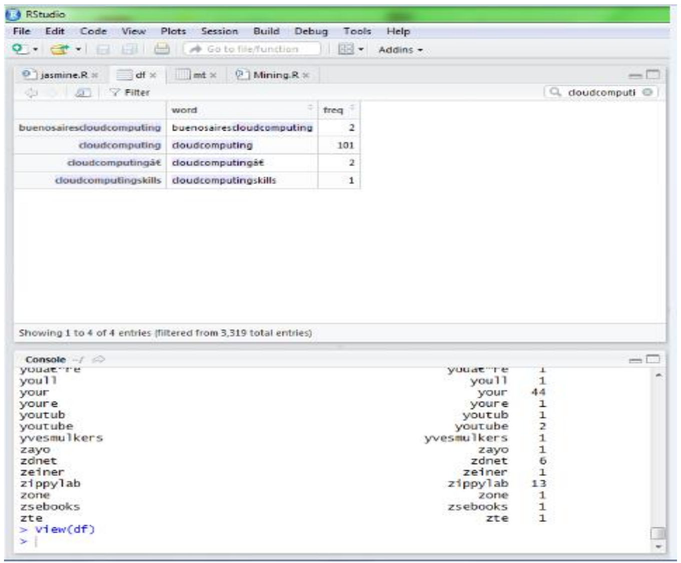
Figure 8: Histogram of dataset3



	word	freq
âxedâuncloudcomputing	âxedâuncloudcomputing	1
cloudcomputing	cloudcomputing	80
cloudcomputingnhttpstcoldapczhuu	cloudcomputingnhttpstcoldapczhuu	1
cloudcomputingnhttpstcopnydszn	cloudcomputingnhttpstcopnydszn	1
cloudcomputingplattform	cloudcomputingplattform	1
cloudcomputingproviders	cloudcomputingproviders	2
computingcloudcomputing	computingcloudcomputing	2
computingncloudcomputingâ€	computingncloudcomputingâ€	1
computingncloudcomputingnhttpstcofvlznbo	computingncloudcomputingnhttpstcofvlznbo	1
hybridcloudncloudcomputing	hybridcloudncloudcomputing	1
ncloudcomputing	ncloudcomputing	36
profitsnhttpstcogdlnizdncloudcomputing	profitsnhttpstcogdlnizdncloudcomputing	1
systemcloudcomputing	systemcloudcomputing	1
worldcloudcomputing	worldcloudcomputing	1

Showing 1 to 14 of 14 entries (filtered from 3,636 total entries)

Figure 9: Tool Generated 'Cloud Computing' word frequency table



	word	freq
buenosairescloudcomputing	buenosairescloudcomputing	2
cloudcomputing	cloudcomputing	101
cloudcomputingâ€	cloudcomputingâ€	2
cloudcomputingskills	cloudcomputingskills	1

Showing 1 to 4 of 4 entries (filtered from 3,319 total entries)

```
youâ€™re                                                    youâ€™re    1
youll                                                          youll    1
your                                                            your   44
youre                                                          youre    1
youtub                                                        youtub    1
youtube                                                      youtube    2
yvesmulkers                                              yvesmulkers    1
zayo                                                            zayo    1
zdnet                                                          zdnet    6
zeiner                                                        zeiner    1
zippylab                                                    zippylab   13
zone                                                            zone    1
zsebooks                                                    zsebooks    1
zte                                                              zte    1
> View(df)
>
```

Figure 10: Tool Generated 'Cloud Computing' word count table and tail words

4. Conclusion

In this paper, we have used the Statistical & graphical approach using R-tool for document retrieval. In this approach, If there are more than one documents with the same representation, every such document is retrieved based on the ranking of the word count [14]. The approach makes use of content based indexing which is a promising solution for Component retrieval from reuse repository. Future research can be aimed in this direction.

References

[1] Richard Cotton, "Learning R", O'Rielly, United Statesof America, 2013, pp.3-7.

[2] González-Calero,P.A,."Applying Knowledge Modeling and Case-Based Reasoning to Software Reuse". IEEE Proceedings – Software, vol. 147, no. 5,n2000.

[3] Sathish Kumar Soora," A Framework for Software Reuse and Research Challenges", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 10, October 2014,pp441-448.

[4] Lucrédio, D.; Almeida, E. S.; Prado, A. F., 'A Survey on Software Components Search and Retrieval', Proceedings of 30th IEEE EUROMICRO Conference, Component-Based Software Engineering Track,2004.

[5] Lucrédio, D.; Gavioli, A.; Prado, A. F.; Biajiz, M.; Yamamoto, C. H.; Almeida, E.S.; Meira, S. R. L. "Component Retrieval using Metric Indexing", proceedings of IEEE International Conference on Information Reuse and Integration (IRI), Las Vegas, Nevada, USA,2004.

[6] Sharma, WS and Rao, Vijay." A rough–fuzzy approach for retrieval off candidate components for software reuse", Pattern Recognition Letters 24(6), 2003,pp. 875-886.

[7] Zhang D.; Tsai J. J. P" Machine Learning and Software Engineering", Software Quality Journal, Issue 11,2003, pp.87-119.

[8] W.B.Frakes and B.A Nejmeh,"An information system for software reuse, Software Reuse: Emerging Technology", IEEE, CS Press,1990, pp.142-151.

[9] Y.S. Yoelle S.Maarek, D.M.Berry, and G.E. Kaiser,"An information retrieval approach for automatically constructing software libraries", IEEE Trans. Software Engineering, vol.17, no.8, 1991,pp.800-813.

[10] Y. Yang, T. Ault, T. Pierce, and C. W. Lattimer, "Improving text categorization methods for event tracking", Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'00),2000, pp65-72.

[11] Naohiro Ishii, Tsuyoshi Murai, Takahiro Yamada, Yongguang Bao,"Text Classification by Combining Grouping, LSA and kNN, icis-comsar", 5th IEEE/ACIS International Conference on Computer and Information Science and 1st IEEE/ACIS International Workshop on Component-Based Software Engineering, Software Architecture and Reuse (ICIS-COMSAR'06),2006, pp.148-154.

[12] Dhillon I. S., Fan J., and Guan Y, "Efficient Clustering of Very Large Document Collections", Data Mining for Scientific and Engineering Applications, Kluwer,2001.

[13] Ravi Teja, Dr. Jasmine K.S, " Data mining tools - An Analytical approach", International Journal of Engineering Research & Technology (IJERT),Vol.4, No.27,May2016,pp.4-8, ESRSA Publication.

[14] Jasmine K.S, Dr.R.Vasantha,"A Mixed Method Approach for Efficient Component Retrieval from a Component Repository", International Journal of Software Engineering and Applications, Journal of Scientific Research, ISSN:1945-3116,Vol.4,No.7,July 2011, pp. 442-445