Volume 3, Issue 8, August 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Development of a Software Repository for the Precise Search and Exact Retrieval of the Components

Amandeep Bakshi, Seema Bawa
Department of Computer Science and Engineering
Thapar University, India

Abstract- The recent progress in the component based engineering has offered many facilities in the field of software reuse. However, this advancement has also brought the challenge such as designing repositories and developing efficient search of software components. But the problem that is generally encountered is the efficient search and the exact retrieval of the software components from the software repositories as they are not properly represented yet. Thus, this paper deals with the construction of the software repository, construction of the operational semantic keyword based retrieval technique for evaluating the best results and at the end comparison between the techniques for retrieval of the software components is made on the basis of precision and recall.

Keywords-Software Repository, Keyword based search, operational semantic search, signature based search, component library

I. Introduction to Software Reuse Repository

Software Reuse Repository is simply a component library which stores the reusable components and must have the characterization of the assets that are included within. In order to make a effective use of a software repository, a reuser must have a clear understanding of its contents, so as to determine that whether his needs are likely to be met by the library [4]. Repositories are used as mechanisms to store, search and retrieve components. But finding and reusing appropriate software components is often very challenging particularly when faced with a large collection of components and little documentation about how they can and should be used. Many software component repositories have been developed often extending the approaches used for software libraries [6]. The software reusable component is defined as "any component that is specifically developed to be used and is actually used in more than one context" [7]. This does not just include code, other products from the system lifecycle can also be reused such as specifications, requirements and designs. Components in this case can be taken to include all potentially reusable products of the system lifecycle including code, documentation, design, requirements, architecture etc.

II. Problems occurs in Software Repository for storing and retrieving components

Consider a large university library. Tens of thousands of books, periodicals, and other information resources are available for use. But to access these resources, a categorization scheme must be developed. To navigate this large volume of information, librarians have defined a classification scheme that includes a library of congress classification code, keywords, author names and other index entries. All enable the user to find the needed resource quickly and easily. Similarly in component library, tens of thousands components are stored. How users can finds out the appropriate one. The solution to this problem is classifying the software libraries in a particular order and use effective retrieval technique. Another Problem rises with the existence of these software assets or components are the presence of a storage structure or a repository that would contain these components and also provide for an easy search and retrieval of the same. The repository has to be constructed and maintained in order to maximize the use of reusable components. In addition to construction of the repository, some retrieval technique also needs to be implemented that utilizes minimal repository structure to effectively support the process of finding software components. Once the repository has been constructed then the information that needs to be stored in it should also be structured in a way such that it should be digitalized, normalized and archived.

After the assets have been properly stored in the repository, the next problem that arises is finding of the appropriate asset by the user and proper query formation for the same purpose. This has to be tackled by applying proper classification schemes which also form an important part of the retrieval process. The classification is important in order to tackle the search problem that arises due to difference in the users thought process and the final query formation as they help in organizing the assets in the repository in an organized manner.

The retrieval of any component is measured according to abstract performance measures such as precision and recall. Hence, the process that needs to be followed for retrieval should guarantee high performance measures. A number of retrieval techniques are in place and the most appropriate one can be chosen according to the requirements and the available resources. Hence, it reduces the effort, time and other resources that go into building the components again and again.

III. Assessment Criteria for Storage and Retrieval Methods

The basic criteria for comparing the various storage and retrieval methods are [9]:

A. Precision: The precision of a retrieval algorithm is the ratio of the relevant retrieved assets over the total number of retrieved assets; this number ranges between 0 and 1. Under the hypothesis that all library assets are visited (example, exhaustive navigation) we get perfect precision (=1) whenever, the matching condition logically implies the relevance criterion. This can be achieved in particular by letting the matching condition be false, which means that no assets are returned (hence no irrelevant assets are returned). When concrete data are not available to quantify precision, we may instead assign values in a discrete five value rating [very low (VL), low (L), medium (M), high (H), very high (VH)] where, VH refers to perfect precision and VL refers to poor precision.

B. Recall: The recall of a retrieval algorithm is the ratio of the relevant retrieved assets over the total number of relevant assets in the library; this number ranges between 0 and 1. Under the hypothesis that all library assets are visited (example, exhaustive navigation) we get perfect recall (=1) whenever, the matching condition logically implies the relevance criterion. This can be achieved in particular by letting the matching condition be false, which means that no assets are returned (hence no irrelevant assets are returned). When concrete data are not available to quantify recall, we may instead assign values in a discrete five value rating [very low (VL), low (L), medium (M), high (H), very high (VH)] where, VH refers to perfect precision and VL refers to poor precision.

The aim of any search program is to maximize precision and recall, so as to provide the effective retrieval.

IV. Effective search and retrieval methods

There are various methods for searching and the retrieval of the components for the effective retrieval. The various methods are discussed as below:

A. Text Based Search Method

In this case, the entire surrogate is searched and the components found valid according the given search are displayed.

B. Keyword Based Search Method

In this case the user will only input the keywords which according to him are relevant and shall provide with the required components, once the keywords are filled in, the system checks for only the keywords present in the surrogate and once the relevant keywords are found, the system shall display the related components.

C. Signature Based Search Method

The user can also base his search on the basis of any particular signature of the components. This type of search helps to provide with components with approximately the exact match from the requirement.

D. Operational Semantic Based Search Method

In operational semantic keyword based search, the user will give as input i.e. component name and its prototype, then the relevant component is displayed along with its input and output. In this we match the candidate asset against a user query on the basis of the candidate behavior on sample inputs. Here the search is based on the input and output. The operational semantic based search method provides the exact retrieval of the component. Hence precision and recall ratio is better in this case as compare to the above methods.

V. Problem Formulation

Every software component can be reused for some other purpose. The problem that is generally encountered while doing the literature review is the efficient search and retrieval of these components from the software library as they are not properly represented. So, our basic task is to identify the general challenges, which are encountered while storing and retrieving the components, which have some reuse potential. In order to do so, the first requirement is a user friendly software repository .So that there is maximum precision and recall of the visited and retrieved software components.

The basic aim is to develop software, which facilitates this and to provide for the maximum relevancy of the retrieved components such that, they can either be reused in the exact same manner or can be modified to do so.

There are many software's and websites which provide for the retrieval of reusable components, but, the search is not precise and the retrieval not that exact. So, the system proposes to make the search as concise and precise as possible and also facilitate the users with various retrieval methods, out of which they can select the one they deem fit for their purpose. The user should be able to perform search based on certain fixed categories by using different retrieval techniques i.e. Keyword based retrieval, signature based retrieval and operational semantic based retrieval. The basic comparison between these techniques on the basis of precision and recall should be done for improving the various gaps. Thus, this paper presents the development of the software repository. Construction of the operational semantic keyword based retrieval technique for evaluating the best results and at the end comparison between the techniques for retrieval of the software components is made on the basis of precision and recall at the end.

VI. Proposed Software Repository

This paper includes the implementation details for the proposed software repository by using effective retrieval techniques .VB.net and SQL server is used for the implementation. It integrates computation, visualization and programming in easy to use environment. There are various techniques, approaches and methods having their own merits and de-merits. In our implementation, we used the advanced keyword based search, signature based search and the operational semantic keyword based search for the effective search and retrieval of the software components.

A. Features of the proposed software repository

The main features of the proposed system are:

1) It contains a dedicated repository for storage of components that can be reused.
2) It contains an interface, which provides with the various criteria of reusable components for the user to confine his search.
3) The interface provides various effective search and retrieval techniques for the exact required components.
4) The Administrator manage the storage , updation and deletion of components.
5) User can easily download the executables, source code and the documentation of the exact components.
6) The software library provides the signature based search, in which the user can search the components on basis of parameters and the return types.
7) This Repository also provides the operational semantics keyword based search, in which the user can search the components on the basis of input output.
8) To compare the signature based search and operational semantic keyword based search with the traditional keyword based search on the basis of some fixed criteria i.e. precision and recall to find out the effective retrieval.

.

B. UML Specification of the Software Repository: Data Flow Diagram

The Proposed repository consists of two main modules i.e. Component Search Engine and Component Storage Management which are shown as follow.

1) Component Search engine

This Module provides the functionality of searching the appropriate component using techniques namely advanced keyword based search, signature based search and operational semantic keyword based search techniques.

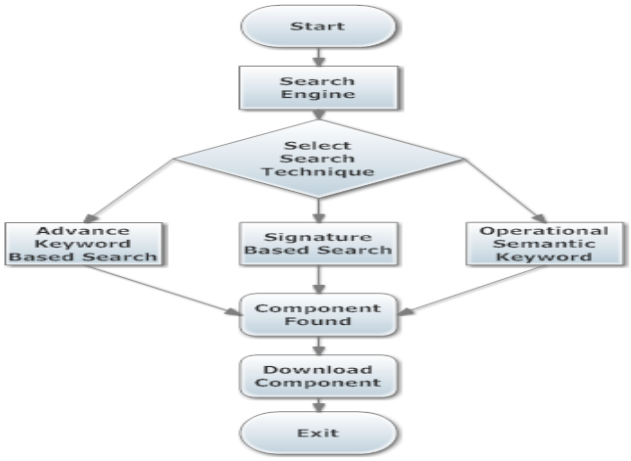

Fig 1: Flow of component search module.

2) Component Storage management

This module allows the administrator to manage the storage, updation and deletion of the components. She can also add operational keywords for the operational semantic based search.

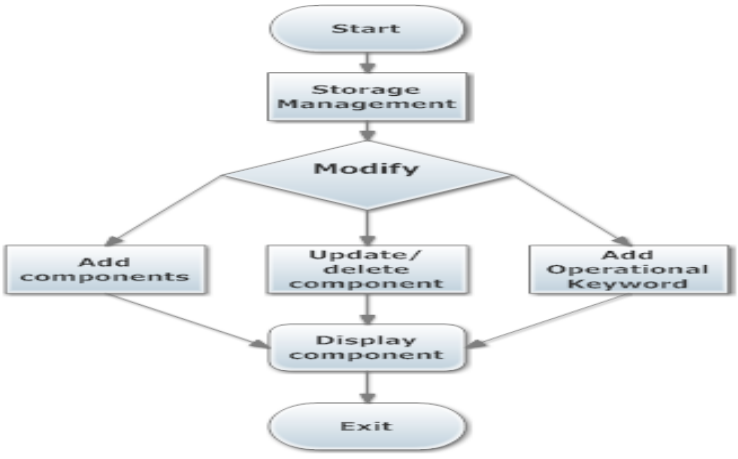

Fig 2: Flow of component storage management.

C. Working of the proposed Software Repository

Software Retrieval is a web application, the working and functionalities of its various units are described in this section.

1) User Authentication: The user has to login every time they wish to access the site and the components available. The user can either be a registered user or the administrator. Figure 3 shows the User Login area where the user has to enter his authenticated username and password to access the system.

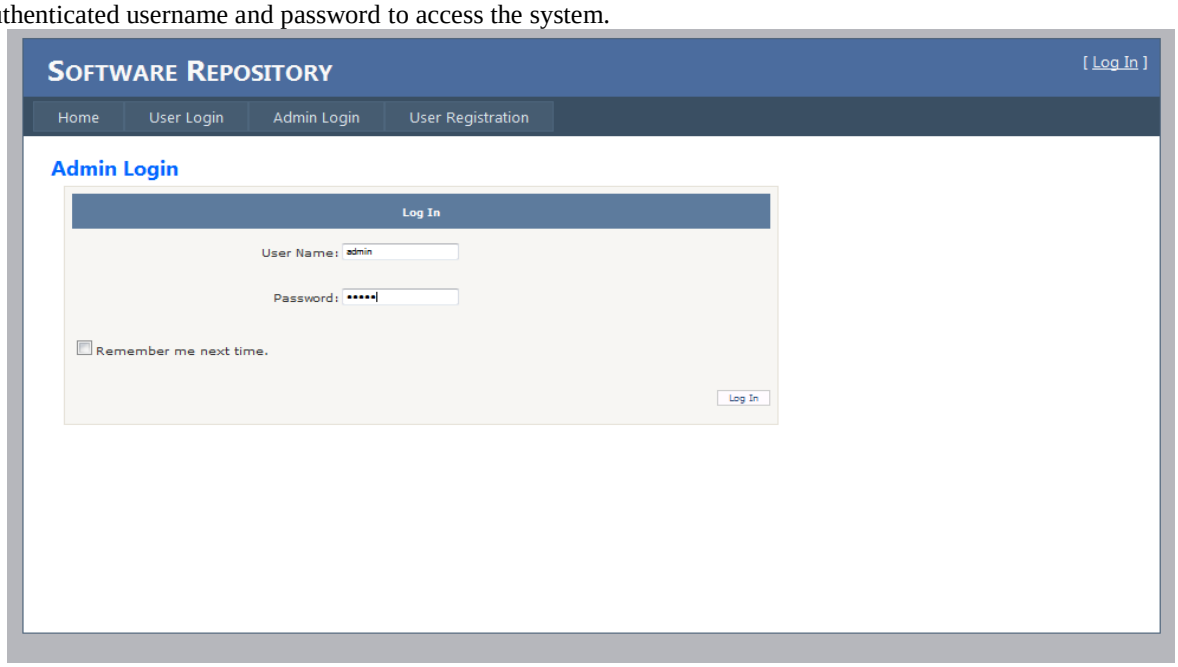

Fig 3: Snapshot of User authentication

2) User Registration: The first step for any user is to register himself so that he may be given the access rights to not only view but also download the available components.

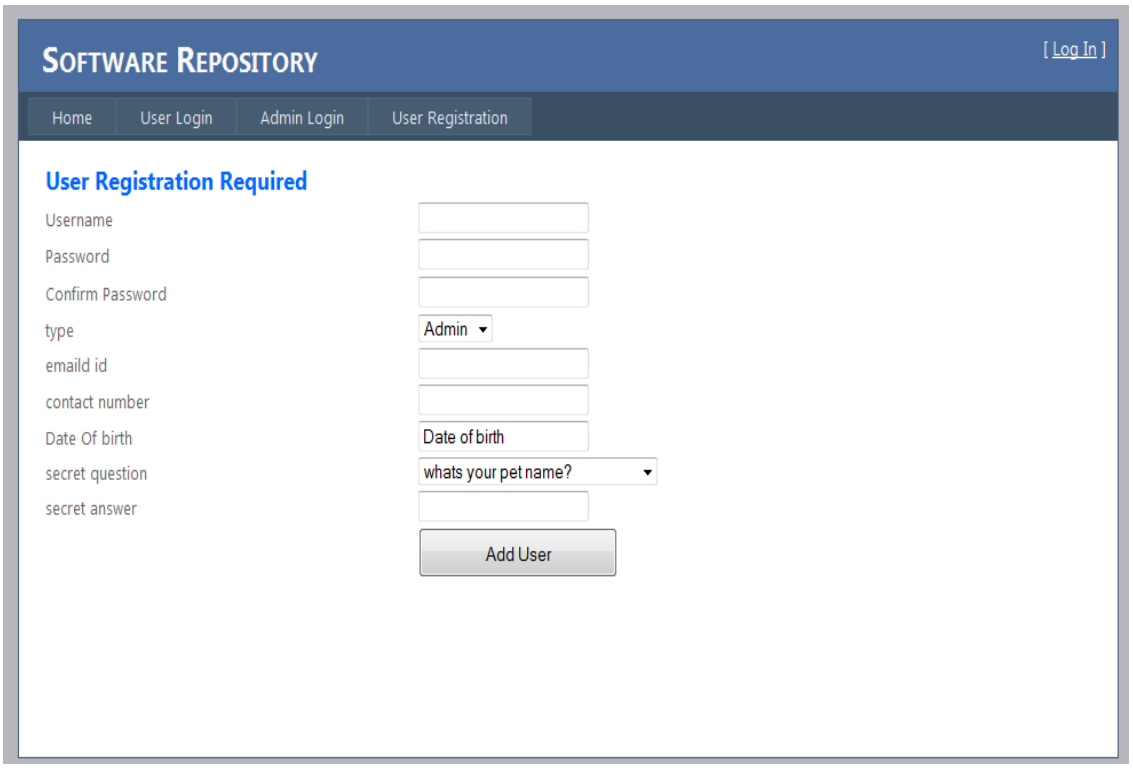

Fig 4: Snapshot of New User Registration

3) Component Search engine

Software repository mainly contains 2 modules i.e. component search engine and component storage management. User can choose component search engine module. After choosing component search engine, user can search the components by using various techniques namely advanced keyword search, signature based search and operational semantic keyword based search.

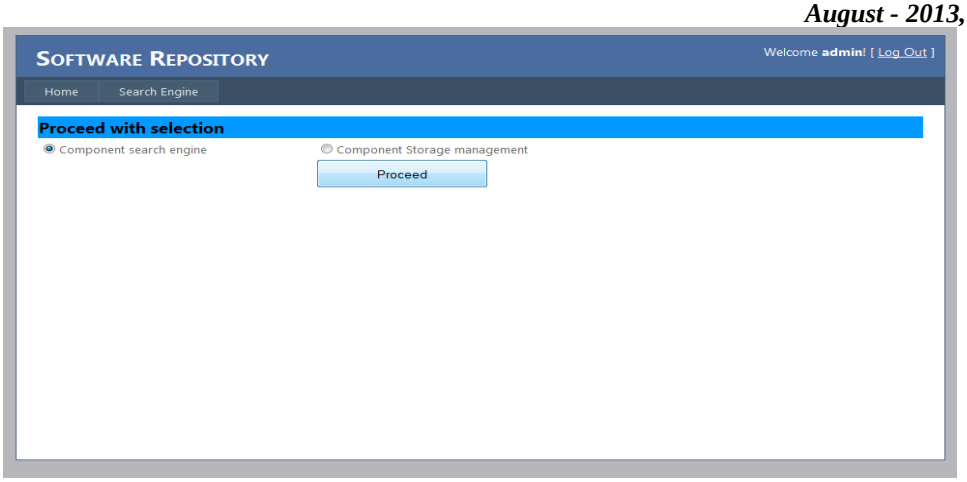

Fig 5: Snapshot of component search engine

4) Keyword Based Search

In this case the user will only input the keywords which, according to him are relevant and shall provide with the required components, once the keywords are filled in, the system checks for only the keywords present in the surrogate and once the relevant keywords are found, the system shall display all the related components as in Figure 6.

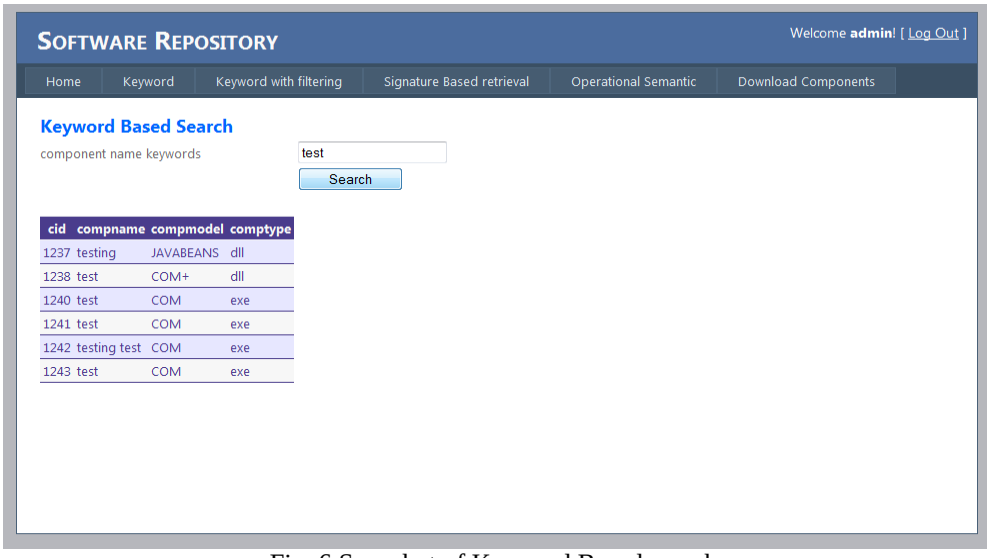

Fig. 6 Snapshot of Keyword Based search

5) Signature Based Retrieval

The user can also base his search on the basis of any particular signature of the components i.e. on the basis of prototype of parameters, number of parameters and return type. This type of search helps to provide with components with approximately the exact match from the requirement.

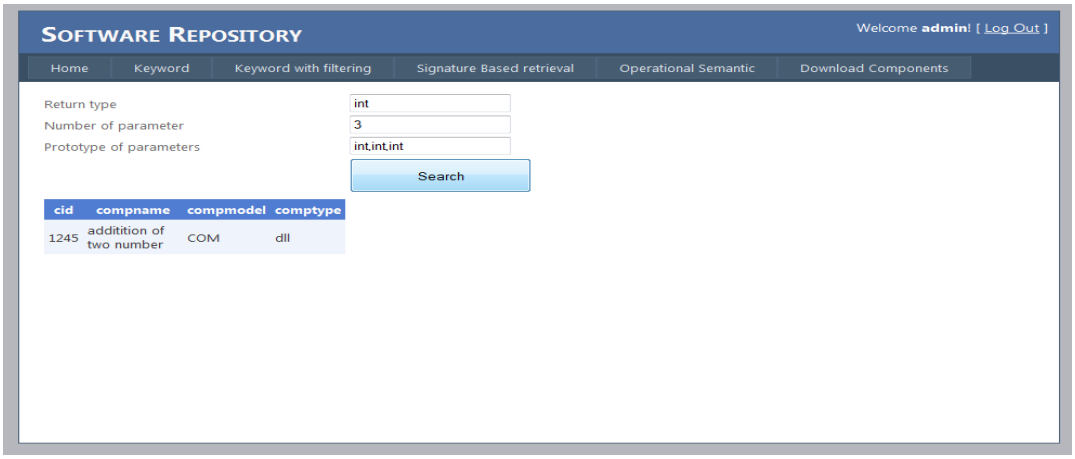

Fig.7 Snapshot of Signature Based search

6) Operational semantic keyword based search

In operational semantic keyword based search, the user will give as input i.e. component name and its prototype, then the relevant components are displayed along with its input and output.

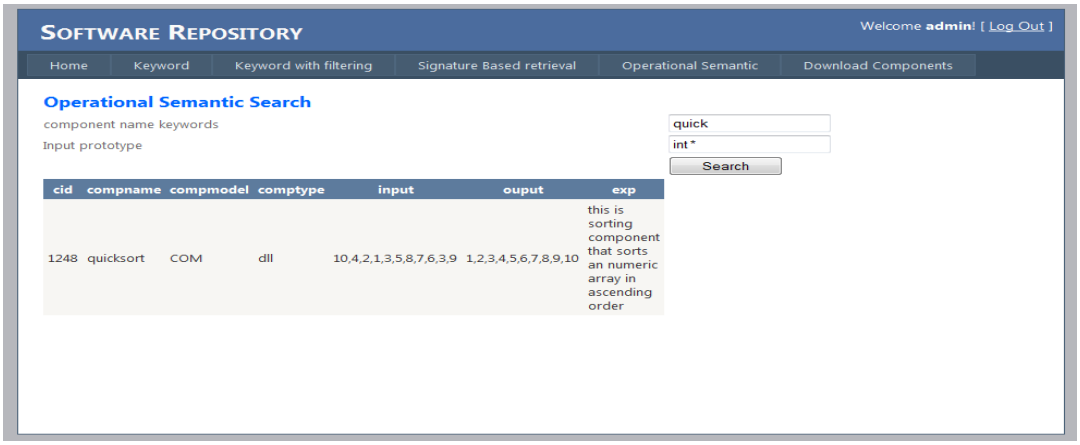

Fig. 8: Snapshot of Operational semantic keyword Based search

7) Download Components

After searching the relevant component according to the need .User can easily download the source code, executables and documentation of the component.

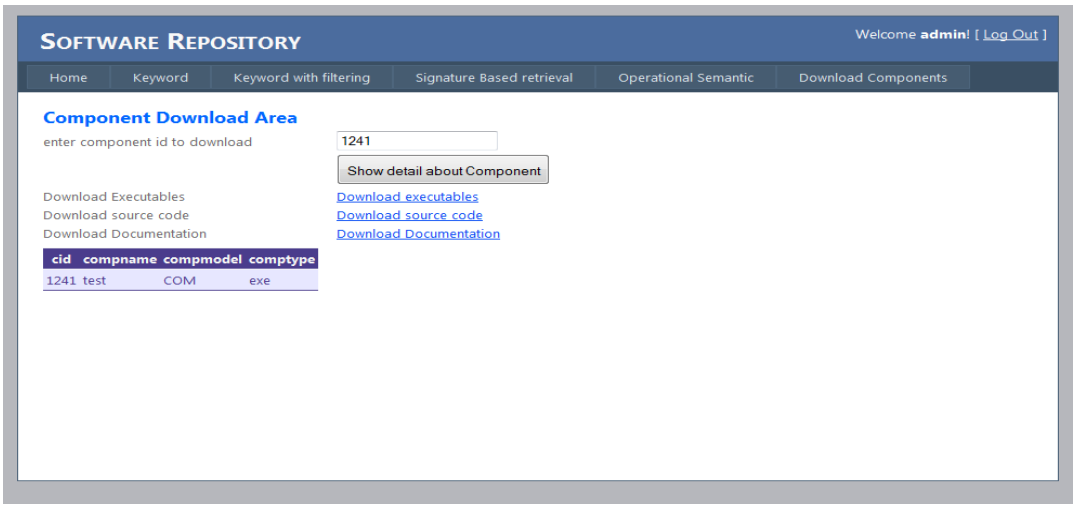

Fig 9: Snapshot for downloading the component

8) Add components

Admin can add the components according to the type of the component, component model, description, input, output and upload the respective executable, source code and the documentation.

.

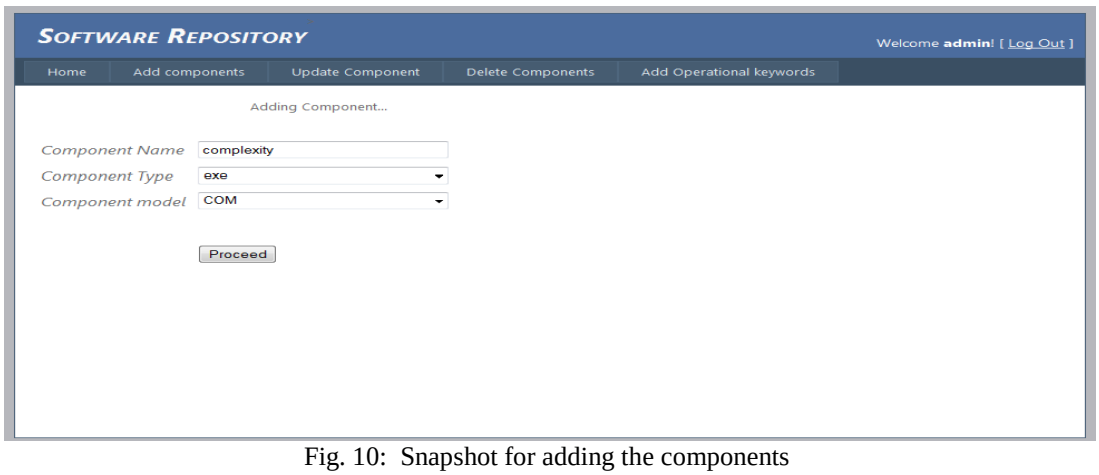

Fig. 10: Snapshot for adding the components

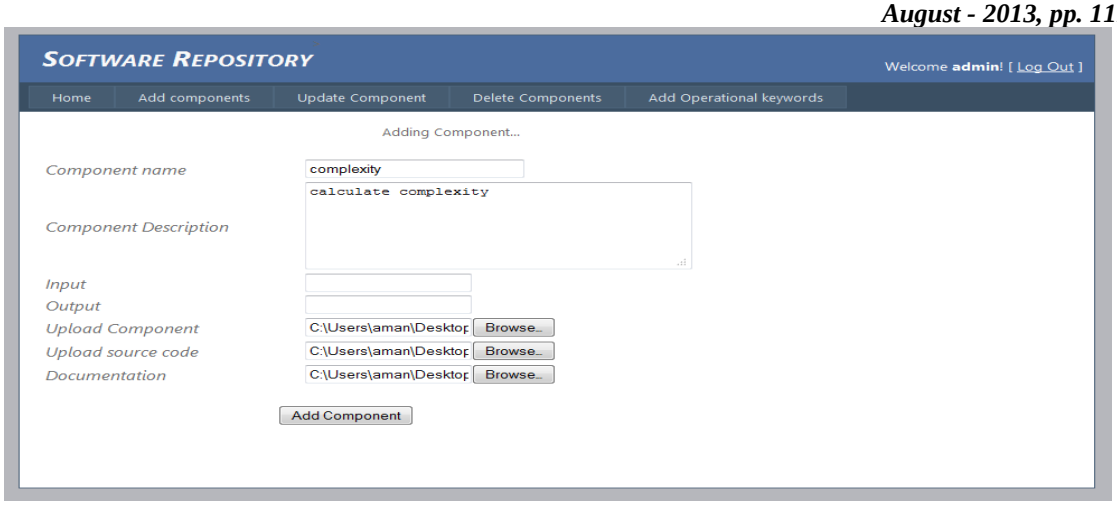

Fig. 11: Snapshot for adding the components

9) Update Components

Admin can update any present components by giving new input output, description or by uploading the new executable, source code and the documentation.

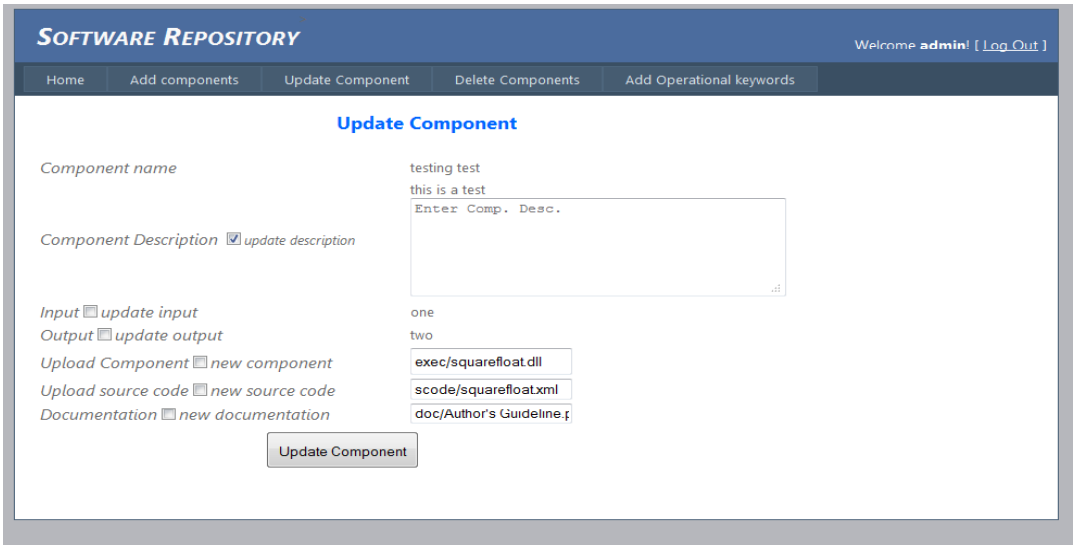

Fig.12: Snapshot of update components

10) Delete Components

Admin can delete any existing component by giving component id and component name. The appropriate component will be deleted from the software repository.

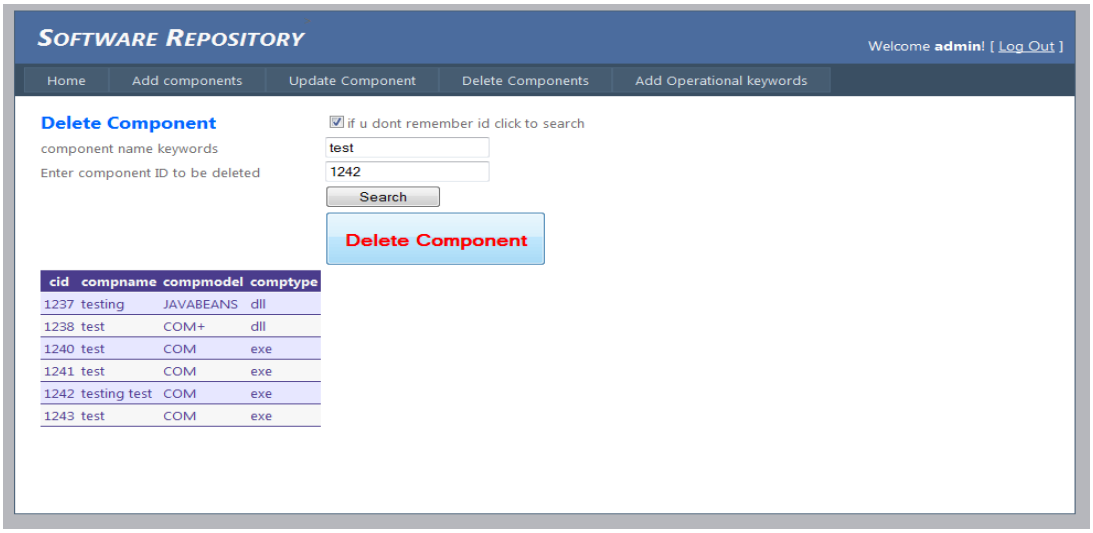

Fig.13: Snap shot for delete components

11) Add operational Keywords

Admin can also add the new operational semantic keyword by giving component id for the operational semantic keyword search.

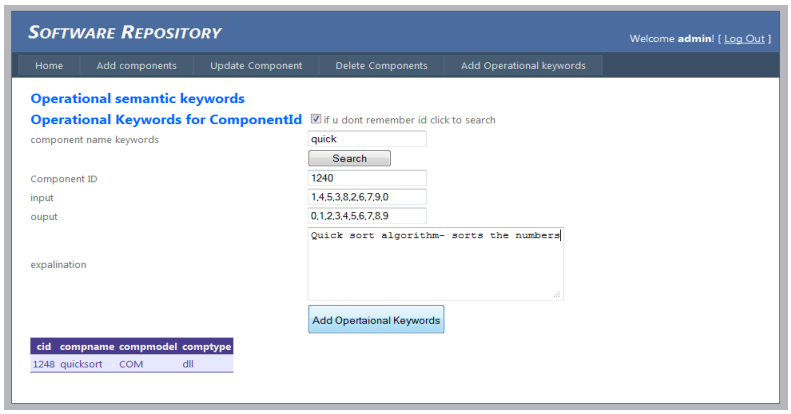

Fig.14: Snap shot for add operational keywords

VII. Experimental Results

A) Result obtained by Precision and Recall ratio of Keyword Based Retrieval.

Suppose there are many components which are stored in the software repository. For example we want to search the component of "addition" i.e. addition of two numbers or addition of two strings. Suppose we search this component from the software repository and compared the result on the basis of precision and recall. The three components have been randomly selected from the software repository which is shown in below table I.

Table: I Randomly Selected Components

Component ID	Component Name	Component Model	Component Type
1254	Addition	Corba	dll
1253	Addition	Java bean	Source code
1257	Addition	Com+	dll

The following result is obtained by the Keyword-based Search on the basis of precision and recall as shown in below table II.

Table II: Result obtained for Keyword Based search

Comp ID / Comp Criteria	**1254**	**1253**	**1257**
Precision	0.25	0.125	0.25
Recall	0.16	0.375	0.166

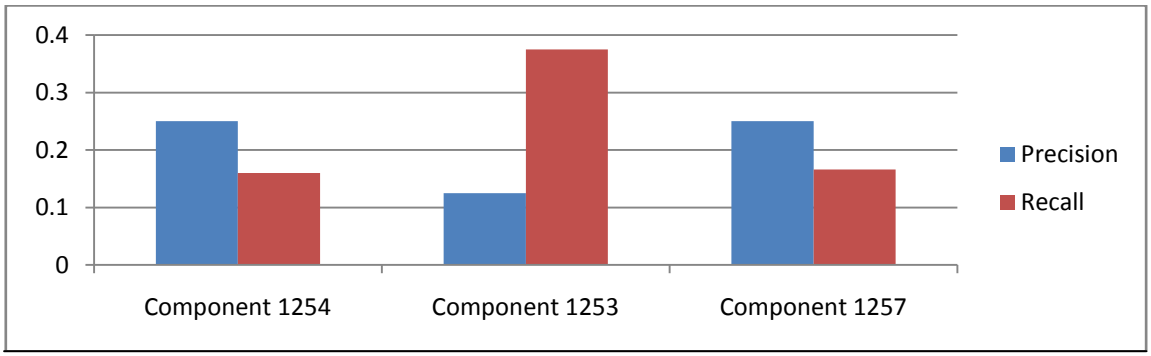

Fig. 15: Graph shows the Precision and recall of Keyword based search

B) Result obtained by Precision and Recall ratio of Operational Semantic Based Retrieval.

The following result is obtained by the Operational Semantic Based Retrieval on the basis of precision and recall as shown in below table III.

Table III: Result obtained for Operational Semantic retrieval

Comp ID / Comp Criteria	**1254**	**1253**	**1257**
Precision	1	1	1
Recall	0.5	1	0.5

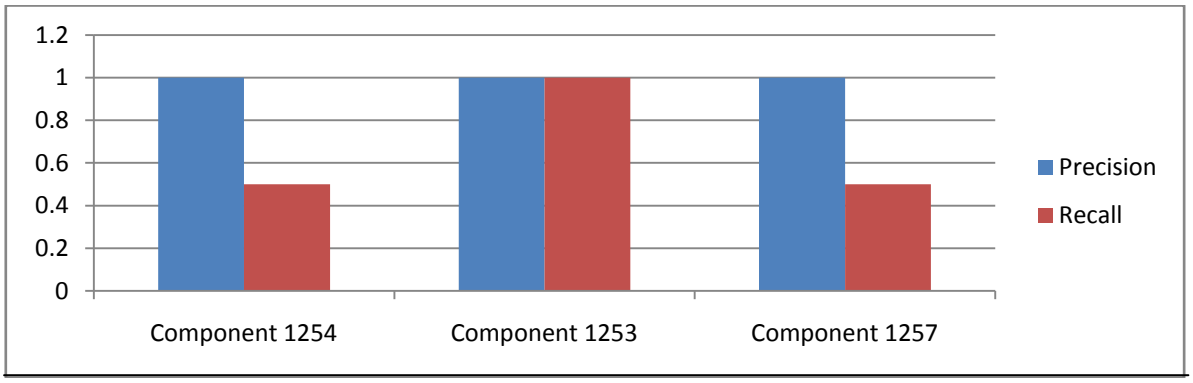

Fig 16: Graph shows the Precision and Recall of Operational Semantic Based Retrieval

C) Comparison between Keyword based retrieval and operational semantic based retrieval on the basis of precision and recall.

The comparison between the two retrieval techniques i.e. Keyword based retrieval and Operational semantic based retrieval show that the values obtained for Precision and Recall of the component. For instance if we compare the values obtained for the components shown in below table I, we find that the average precision of the Operational semantic based search i.e. 1. (known as perfect precision) is more than that of the Keyword based search i.e.0.208. Similarly, on comparing the values of recall we find that the value of Operational semantic based search i.e. 0.66 is more than the recall value of keyword based search i.e. 0.233 so it can be concluded that the components obtained from the operational semantic based search are more close to the user requirements than the Keyword based search.

The Comparison between both the techniques are shown in following table IV.

Table IV: Result comparison on the basis of Precision and Recall

Retrieval Techniques / Component ID	Keyword Based Retrieval		Operational Semantic Retrieval	
	Precision	**Recall**	**Precision**	**Recall**
1254	0.25	0.16	1	0.5
1253	0.125	0.375	1	1
1257	0.25	0.166	1	0.5
Mean	**0.208**	**0.233**	**1**	**0.66**

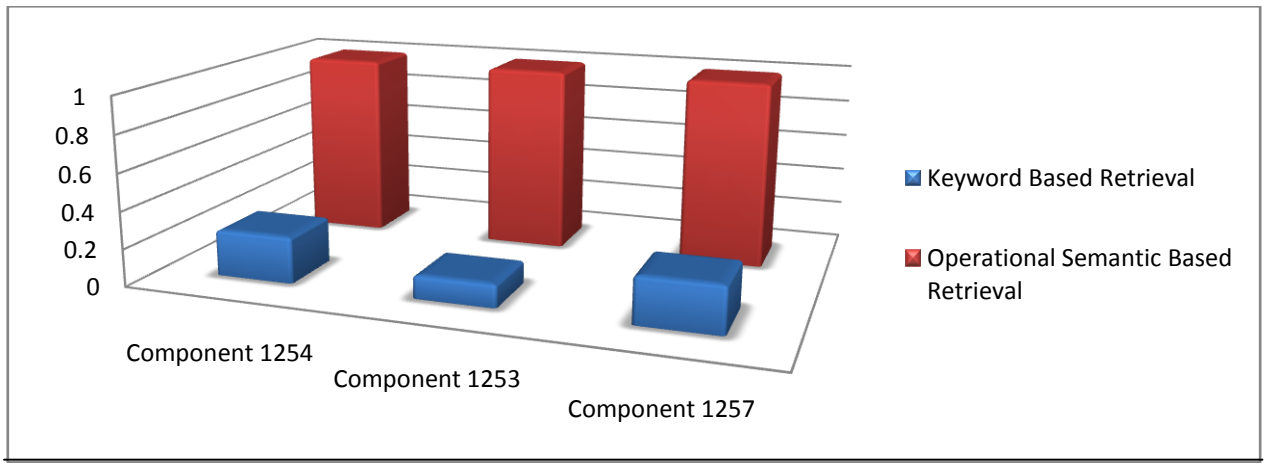

Fig. 17: Comparison on the basis of precision between the techniques

Figure 17 gives the graphical representation of the above table. From Table IV, a graph is plotted which gives the comparison of two techniques on the basis of precision. From the graph it can be inferred that the operational semantic based retrieval has a higher precision ratio (i.e. Perfect precision) as compared to the keyword based retrieval. Figure 17

gives the graphical representation of the above table. From Table IV, a graph is plotted which gives the comparison of two techniques on the basis of recall. From the graph it can be inferred that the operational semantic based retrieval has a higher recall ratio as compared to the keyword based retrieval.

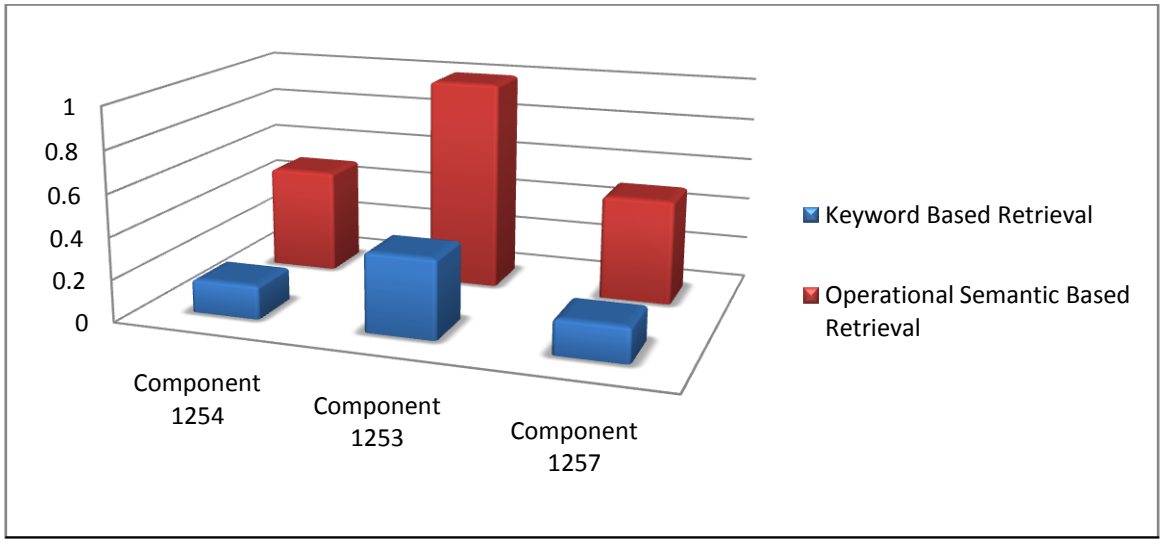

Fig.18: Comparison on the basis of recall between the techniques

Hence, it can be concluded that operational semantic based search is much better than keyword based search because operational semantic based search shows the best results for the exact retrieval and the precise search on the basis of precision and recall and it also saves the time and reduces effort for searching the exact match according to the need of the user requirements. Hence, it can be concluded that the operational semantic based search is the best among all the search criteria set up in the search engine.

VIII. Conclusions

This thesis presented a study on the ways to make the reusable components retrieved with more ease to the users by proposing and implementing an interface to do so. The system implemented also stores effectively the various reusable components in manageable and understandable categories from where the user can extract the desired component effectively and efficiently.

After implementation and testing of the system the conclusion come to light is that:

1) Out of all the search criteria the operational semantic based search is better than the Keyword based search because the retrieved assets are more precise and the search results are more exact.
2) Operational semantic based retrieval saves the time and reduces effort for searching the exact match according to the need of the user requirements.
3) Any user can easily download the relevant retrieved component according to his/her needs.

Thus, the system has been successfully implemented and tested conforming to the objectives set in the problem statement.

IX Future Scope

The Effective Retrieval can be enhanced as:

1) The repository can be extended further by adding more components in the categories present and also by adding new and relevant components in the software repository.
2) The operational semantic based search and the retrieval technique can be further converted into the semantic and syntactic based search for making the search and retrieval more efficient than the operational semantic retrieval.
3) More work is needed to improve the similarity matching algorithm to further increase the precision and recall ratio.

References

[1] W.S.Sarma and V.Rao, "A Rough–Fuzzy Approach for Retrieval off Candidate Components for Software Re-use," Pattern Recognition Letters, Vol. 24, No. 6, 2003, pp. 875-886.

[2] P.A.González-Calero, "Applying Knowledge Modeling and Case-Based Reasoning to Software Reuse," *IEEE Proceedings* – Software, Vol. 147, No. 5, October 2000, pp. 169-177.

[3] D.Lucrédio, et al., "Component Retrieval Using Metric Indexing," *IEEE International Conference on Information Reuse and Integration, Las Vegas*, 8-10 November 2004, pp. 79-84.

[4] H.Mili/A.Mili, "Reuse Based Software Engineering: Techniques, Organization and Measurement", wiley-Interscience Publication, 2000, pp-9.

[5] Vanilson and Daniel, "Specification, Design and Implementation of a Reuse Repository", *In 31st annual International Computer software and Applications Conferences, IEEE*, 2007.

[6] Scott Henninger, "Supporting the Construction and Evolution of Component Repositories", In Proceedings of the 18th International Conference on Software Engineering, 1996, IEEE, pp-280-286.
[7] Jasmine Kalathipparambil and vasantha, "A Mixed Method Approach for Efficient component retrieval from a component repository", *Journal of Software Engineering and Applications*, 2011.
[8] Navjoti Pandhi, "reusing components using genetic components", ME thesis, May 2004.
[9] A.Mili, R.Mili and R.T.mittermeir, "A Survey of Software Reuse Libraries", *Annals of Software Engineering* 5, 1998, pp-349-414.
[10] Pfister, Joachim and Hans-Dieter Zimmermann, "Towards the Introduction of an Institutional Repository: Basic Principles and Concepts", 2008, pp- 230.
[11] John Grundy, "Storage and Retrieval of Software Components using Aspects", *IEEE*, 2001.
[12] Hanen Ben Khalifa, Oualid Khayati, Henda Hajjami Ben Ghezala, "A Behavioral and Structural Components Retrieval Technique for Software Reuse", *IEEE Computer Society*, 2008, pp-134.
[13] Yumen Ye, "An Active and Adaptive Reuse Repository System", *34th Hawaii International Conference on System* Sciences, *IEEE*, 2001.
[14] John Grundy, "Storage and Retrieval of Software Components using Aspects", *IEEE*, 2001.
[15] Jihyun Lee, Jinsam Kim and Gyu-Sang Shin, "Facilitating Reuse of Software Components using Repository Technology", 10th Asia-Pacific Software Engineering Conference, *IEEE*, 2003.
[16] Ronaldo C.Veras and Silvi.R.L.meira, "Comparative Study of Clustering Techniques for the Organisation of Software Repositories", *IEEE*, 2007.
[17] Hanen Ben Khalifa, Oualid Khayati, Henda Hajjami Ben Ghezala, "A Behavioral and Structural Components Retrieval Technique for Software Reuse", *IEEE Computer Society*, 2008, pp-134.
[18] Oualid Khayati, Jean-Pierre Giraudin, "Components Retrieval Systems", 2001, pp-56.
[19] M.R. Girardi and B.Ibrahim, "Automatic Indexing of Software Artifacts", 1994.
[20] Jingbo Zhu, Tianshun Yao: "A Knowledge-based Approach to Text Classification", *IEEE*, 2001.
[21] Divya Pandove, "Designing an interface for effective retrieval of components", ME thesis, June 2010.
[22] Panos Constantopoulus and Martin Dorr, "Component Classification in the Software Informational Base", Prentice Hall, 1995, pp-177-200.
[23] Daniel Lucredio, "A survey on Software Components Search and Retrieval", *30th EUROMICRO conference*, *IEEE*, 2004.
[24] Y.S.Maarek, D.M.Berry and G.E. Kaiser, "An Information Retrieval approach for automatically constructing software libraries", *IEEE transactions on software engineering*, 1991.
[25] S. Henninger, "An evolutionary approach to constructing effective software reuse repositories," *ACM Transactions on Software Engineering and Methodology*, vol.6, 1997.
[26] R.Prieto-Dı´Az, and P.Freeman, "Classifying Software for Reusability", *IEEE Software 4*, 1987, pp- 6–16.
[27] Wang Haitao and Chen Xing, "Study on a Component Library Model Based on the Four-Layer Architecture", International conference on intelligent networks and intelligent systems, *IEEE*, 2011.
[28] Zhu Jingbo, Yao Tianshun, "A Knowledge-Based Approach To Text Classification", *IEEE*, 2001.
[29] Chao-Tsun Chang, William C. Chu, Chung-Shyan Liu, Hongji Yang, " A formal approach to software components classification and retrieval," 264-269 Electronic Edition, *IEEE Computer Society DL*, Colin J. Hardy, Hlen M. Edwards, J. Barrie Thompson, 1997.
[30] Lei Zhang, LichaonChen, Lihu Pan, Yingjun Zhang, "A Novel Approach of Component Retrieval in Large-Scale Component Repositories", *IEEE*, 2012.
[31] Cem Kaner, "Quality Assurance in software testing", Orlando, November 2006.
[32] Jiang Guo and Luqi, "A Survey of Software Reuse Repositories", *IEEE*, 1999.
[33] GE Junwei, TAO Cong, FANG Yiqiu, "Architecture for Component Library Retrieval on the Cloud", *IEEE*, 2011, pp 536-539.
[34] P. Chen, R. Hennicker, and M. Jarke," On the retrieval of reusable software components", August 1997.
[35] Pietro Abate, Roberto Di Cosmo, "Learning from the future of Component Repositories", *ACM Transactions on Software Engineering*, 2012, pp-51.

