Volume 4, Issue 3, March 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Multiple Search Classification of Repository

Vaneet Kaur Bhatia
CSE Department
GZS PTU Campus Bathinda
PTU Jalandhar, India

D.S Dhaliwal
CSE Department
BGC Sardulgarh
PTU Jalandhar, India

Namisha Modi
CSE Department
GZS PTU Campus Bathinda
PTU Jalandhar, India

Abstract— The software repository is used for retrieving, managing and retrieving large numbers of software components. Repository should be designed to meet the growing and changing needs of the software development organizations. Representation and storage of reusable software components in software repositories to assist retrieval is a key concern area. In this paper various assets of the component repository like component searching mechanisms and classifications such as Enumerated, Free text, Faceted and Attribute Value classifications are discussed. A New classification scheme is proposed which covers the limitations of existing classification schemes. In order to do so, the first requirement is a user friendly interface which provides the best solution for reorganization of the components in the repository so that recall and precision may be maximized. . The basic comparison of search presented in this paper on the basis of precision and recall deals with evaluating the technique for retrieval of the software components for the purpose of reuse.

Keywords— Repository, Retrieval, Component, Reusability, Query.

I. INTRODUCTION

A component is an autonomous and independent part of a system having complete functionalities. CBSE is a process that aims to design and construct software systems using reusable software components. Component-Based Software Engineering (CBSE) refers to the construction of large software systems from existing parts. CBSE uses software engineering principles to apply the same idea as object oriented programming to the whole process of designing and constructing software systems. Software reuse is a worthwhile goal because it is used to reduce software costs and improve software quality as well as programmer productivity [2]. A component-based system highlights appropriate reuse and composition of software components as its key concept. However, finding and reusing appropriate software components is often very difficult, especially when faced with a large collection of components and little documentation about how they can and should be used[1]. The benefits of software reuse are increased productivity, quality and shorter time to market [3]. The paper is focusing on the issue, that CBSE faces, is how to select the best qualifying component from available repositories. Qualification means to check how much a reusable component is according to our requirements [12].

II. FACETS OF COMPONENT REPOSITORY

A software component repository represents a potential tool for the exercise of reuse inside organizations. When the components stored in repository are well organized and managed, this potential becomes a reality. The main idea is to allow the repository to serve as a knowledge base, since a software component is the product of tasks based on the intensive use of knowledge. After analysis of the requirements we search the software components in the repository so that we can make a design with the reusable software components. A reusable software artifact like requirement document and design can be put in the software repository for future reuse. Software engineering data contains different type of documents like requirement specifications, design, testing relevant discussions, test runs, source code and bugs details. All these data sets are available in the projects own repository and developers of the same organization can use or refer them in future[6].
Reusability is a process of utilizing and applying already developed components. There are many work products that can be reused, for example source code, specifications, designs, architectures and documentation. The structure of a repository is generally regarded as key to obtaining good retrieval results. Even if the matching algorithm is Intelligent, if components are indexed or otherwise structured poorly, it will be difficult to achieve good retrieval performance [4]. Effective software reuse requires that the users of the system have access to appropriate components. Retrieval should allow users to formulate high-level queries about component capabilities and takes account of the context in which a query is performed to assist query formulation [1]. The user must access these components precisely and quickly, and if necessary, be able to modify them. To classify software it allows reusers to organize collections of components into structures so that they can be searched easily. Most retrieval methods require some kind of classification of the components. Enumerated, Free Text, Attribute Value, and Faceted classifications are the four different classification techniques that had been previously employed to construct reuse repository. This classification scheme faces various challenges. Free text classification has Limited Vocabulary, Uncertain and consists of indexing costs. Enumerated

classification has no flexible classification of components, fastest method but difficult to expand. Attribute Value classification is the slowest method and has no ordering and faceted classification is difficult to find right combination of terms, effective for domain specific rather than broad and heterogeneous collections [12].

A) Storage and Retrieval of Software Component

Successful reuse requires having a wide variety of components with high quality, retrieval mechanisms and proper classification. Effective software reuse requires that the users of the system have access to appropriate components. Effective repository retrieval interfaces are essential to the success of any component-base development system [10].One major issue is the reuse barrier because of the large and constantly changing component repositories. Programmers often fail to correctly anticipate the existence of reusable components. When developers do not believe that a component exists, they will not even make an attempt to find it. A second issue with repository retrieval systems is vocabularies and ill-defined queries submitted by non-expert users.An effective retrieval mechanism including a representation schema for indexing and a matching criterion between a query and a component is essential.

B) Querying and Browsing component

Querying and browsing are the two major information access mechanisms for most users. Querying is direct that users formulate a query and the system returns information matching the query. However to formulate a query is a quiet difficult task because users have to overcome the gap from the situational model to the system model. In browsing, users determine the usefulness or relevance of the information currently being displayed in terms of their task and traverse its associated links Mili et al. [5] claims that browsing is the most predominant pattern of component repository usage because most programmers often cannot formulate clearly-defined requirements for reusable components so they rely on browsing to get familiar with available reusable components in the repository.

C) Effective retrieval mechanism in retrieving

There are several retrieval techniques. Three major approaches in retrieval mechanisms text-based, descriptor-based and formal specification-based. A component is represented by their textual documents in text-based approach and information retrieval technology is used to match components to queries [15]. In descriptor-based approach, components are represented by a set of selected descriptors. The semantic relationships among those descriptors are captured in a predetermined structure that can be specified by a semantic network. Components are represented with formal specification languages in specification-based approach, and specification refinement systems are used to determine whether a component matches a query written in formal specification languages or not. Retrieval plays an important role in locating reusable components that match reuse queries.

D) Evaluation

To check whether component repository works effectively we evaluate the correctness and effectiveness by using two measures that are recall and precision. Precision and recall are two concepts that have traditionally been used to evaluate the method of retrieval software components.

$$\text{PRECISION} = \frac{\text{Ratio of relevant retrieved assets}}{\text{Total number of retrieved assets}}$$

PRECISION is the ratio of the number of relevant records retrieved to the total number of irrelevant and relevant records retrieved. It is usually expressed as a percentage.

This is a number that ranges between 0 and 1.Under the hypothesis that all the library assets are visited, 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 no assets are returned.Precision means that all the retrieved components are exact as per query submitted by a user.

$$\text{RECALL} = \frac{\text{Ratio of relevant retrieved assets}}{\text{Total number of relevant assets in the library}}$$

RECALL is the ratio of the number of relevant records retrieved to the total number of relevant records in the database. It is usually expressed as a percentage.

This is the number that ranges between 0 and1.Under the hypothesis that all the library assets are visited, we get perfect recall (=1) whenever the relevance criterion logically implies the matching condition. This can be achieved in particular by letting the matching condition be true which means that all library assets are returned. Recall means to get all the relevant components.

It is very difficult to get both high precision and high recall using the classical approach. For effective use, the keywords have to be independent, they have to span the range of the users need, and the users must be able to pick the right keywords.

III. PROPOSED SYSTEM

Information retrieval from components repository is a tedious work. The size of repository is often very large. Repository contains large number of components and its specification. Repository is a link between development for reuse, where the components are produced and for effectively reusing the components from the repository, the selection of proper retrieval technique is essential.There is a need of user friendly interface which provides the best solution for organisation of the components so that it can be reused by the way it is required. To propose a new storage retrieval method based on multiple search criteria for better retrieval, developing a repository for the effective retrieval using a new classification scheme to validate the same by calculating precision and recall of repository. Proposed classification scheme which combines the attribute value and faceted classification schemes to classify components. The components in the repository are represented by using a surrogate. A surrogate may be defined as the metadata which describes the component almost fully.

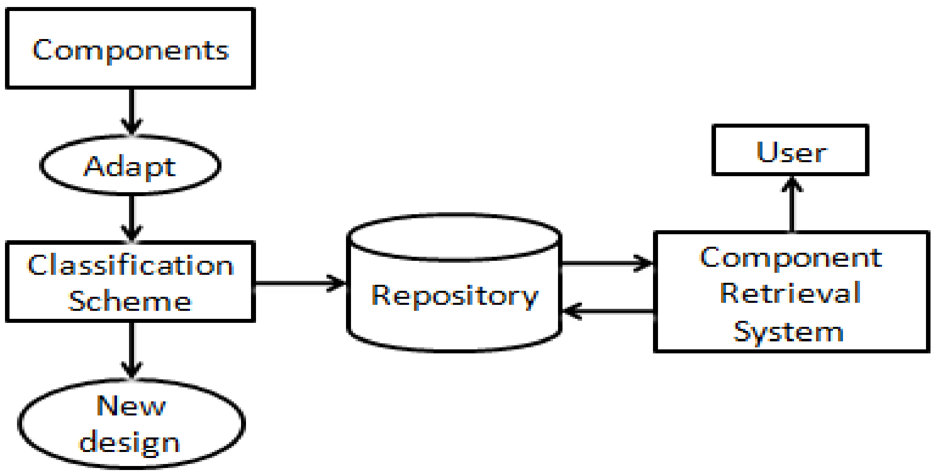

Fig. 1: Architecture of Proposed System

It contains a dedicated repository for storage of components that can be reused. Each component has a separate surrogate which represents it in the repository. It contains an interface, which provides with the various categories and subcategories of reusable components for the user to confine his search. The interface also provides with the keyword based search, in which the user is free to search on his own will based on the type of surrogate he prefers to search from. The system implements the use of a dynamic surrogate for each type of search and retrieval mechanism, which the surrogates of the component keep on changing according to the choice of the user. A separate administrative unit is also present; in order to manage various categories, subcategories and user added criteria.
The proposed component repository system comprises of easy classification according to multiple search criteria in the repository and selection refinement.
In this repository system the two types of search criteria has been considered first is query based search in which the different values of categories are searched according to the user and a person may also select some list of attributes from the interface. The user selects the category and result is displayed on this basis.
Second is keyword based search in which the user when desirous of searching a component may enter some keywords. The query formation module should accept all the keywords entered and form the query using those keywords.
A sample size of 20 components has been taken and on this basis the average precision and recall is calculated. Precision and recall are two concepts that has been used to evaluate the method of retrieval software components.

$$\text{PRECISION} = \frac{\text{Ratio of relevant retrieved assets}}{\text{Total number of retrieved assets}}$$

Precision means that all the retrieved components are exact as per query submitted by a user.

$$\text{RECALL} = \frac{\text{Ratio of relevant retrieved assets}}{\text{Total number of relevant assets in the library}}$$

Recall means to get all the relevant components.
This search is being categorized in 3 types. For both query based search and keyword search these categories have been considered.

1) Search for various applications
2) Search for various games
3) Search for various browsers

Comparing the value of the precision and recall from fig.2 for the two searches for all the cases of applications, games and browsers we found that the keyword based search shows most encouraging results. The results obtained in the module are formatted so that the user can clearly understand the functionality of component before choosing one. All the results are taken and are displayed along with their details. Now the user can select his choice of component to download or save a component in the location specified by the user.

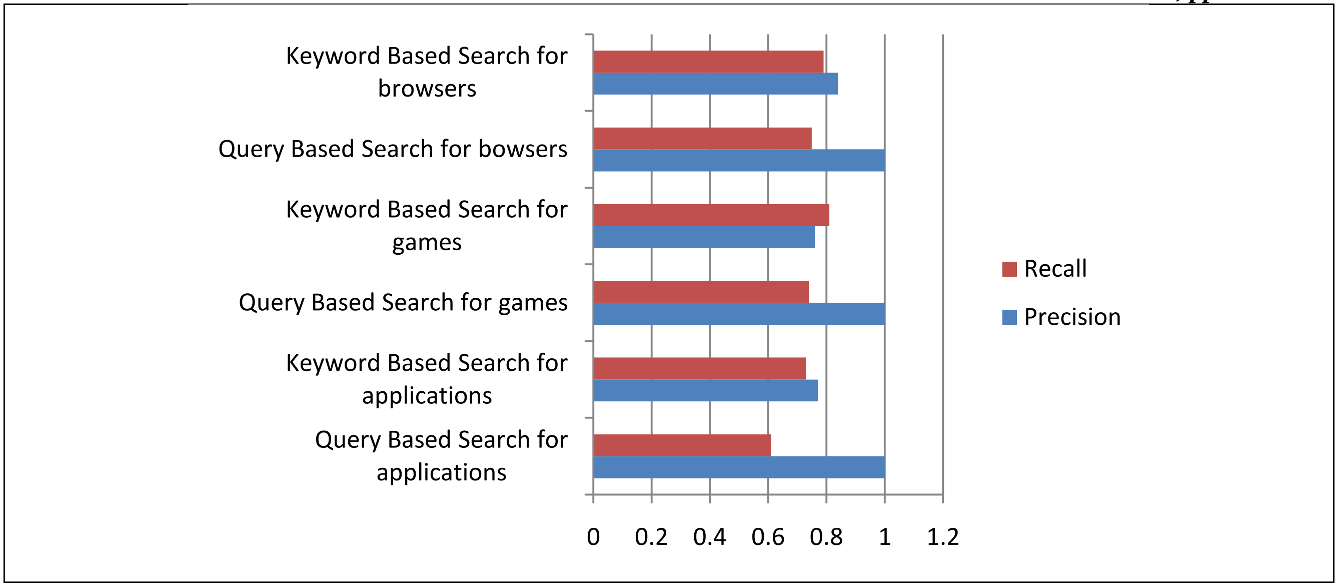

Fig. 2: Comparison graph of two searches based on all three conditions

The results obtained in the module are formatted so that the user can clearly understand the functionality of component before choosing one. All the results are taken and are displayed along with their details. Now the user can select his choice of component to download or save a component in the location specified by the user.

IV CONCLUSION

The paper presented the 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 with the help of multiple search criteria in the repository. An effective software tool with user friendly interface is designed and successfully implemented using proposed classification scheme. The system implemented also stores effectively the various reusable components in a manageable and understandable multiple categories of the search criteria from where users can extract the desired component effectively and efficiently. After implementation and various experimental results the following conclusions come to light:

1) The implementation of the multiple search criteria in the component repository system makes the search more precise and the search results are more exact.
2) Proposed classification scheme which combines the attribute value and faceted classification schemes for the multiple search criteria to classify components with the various attributes gives precise results.
3) Out of all the comparisons of the search criteria the keyword based search is better as the measures of recall and precision are more accurate.

Future work involved with this classification scheme will be to refine the scheme for Multi-Tired or Multimedia presentation of components. Also various ways of optimizing system speed and efficiency must be explored in order to keep the repository as an effective design tool.

REFERENCES

[1] J.C. Grundy, “Storage and retrieval of Software Components using Aspects”, in Proc. of the 2000 Australasian Computer Science Conference, Canberra, Australia ,IEEE CS Press, pp 95-103,2000.

[2] H. Yao and L. Etzkorn, “Towards a semantic-based approach for software reusable component classification and retrieval”, In: Proceedings of the 42nd annual southeast regional conference, pp. 110–115, ACM Press, 2004.

[3] S. Singh and S. Goel, “CBSE versus COTS Based Software Development” IJARCSSE, Vol 2, Issue 10,pp 29-32, October 2012

[4] S. Henniger, ‘Supporting the construction and evolution of component repositories’, Proceedings of the 18th International Conference on Software Engineering (ICSE'96), Berlin, Germany, 1996.

[5] Mili, S.Yacoub, E. Addy and M. Hafedh, “Toward an Engineering Discipline of Software Reuse”, IEEE Software Vol.16, No. 5, pp. 22–31, 1999.

[6] Nida Yasir, Bushra Jamil and Javed Ferzund, “PDCML: A Model for Enhancing Software Reusability”, IJSEA Vol. 7, No 1, pp.123-136, 2013.

[7] Ruben Prieto-Diaz Software Production Consortium, Herndon, VA, “Implementing faceted classification for software reuse”, ACM Press New York, NY, USA, pp. 88 -97, 1991.

[8] S. Henniger, “Supporting the construction and evolution of component repositories”, Proceedings of the 18th International Conference on Software Engineering (ICSE'96), Berlin, Germany, 1996.

[9] P. Niranjan, C. V. Guru Rao, “A mock- up tool for software component reuse repository”, International journal of software engineering and applications (IJSEA), Vol.1, No. 2, April 2010.

[10] S. Shiva, L. Shala, “Using Semantic Wikis to Support Software Reuse”, Journal of software, Vol.3, No. 4, pp.1–8, April 2008.

[11] Ye, Yunwen, “An Active and Adaptive Reuse Repository System”, Proceedings of 34th Hawaii International Conference on System Sciences (HICSS-34), pp. 9065-9075, Jan 2001.

[12] Stephen White, Michael Edwards, “Domain Engineering: The challenge, Status and Trends”, 1996.

[13] H. H .Kagdi, M. L. Collard, and J. I. Maletic, “A survey and taxonomy of approaches for mining software repositories in the context of software evolution”, Journal of Software Maintenance, pp. 77-131, 2007.

[14] N. Kaur, “Retrieving Best Component from Reusable Repository”, pp. 23, 2005.

[15] W. B. Frakes, T. P. Pole, “An Empirical Study of Representation Methods for Reusable Software Components”, IEEE Transactions on Software Engineering, Vol.20, No. 8, pp. 617–630, 1994.