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

Dr. Jasmine K.S¹, Ravi Teja²

¹Department of MCA, R.V. College of Engineering, Bangalore, Karnataka, India
²Software 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].

Corresponding Author: Jasmine K.S
Email Id: jasmineks@rvce.edu.in

© 2017 ijrei.com. All rights reserved
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 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 a drawback 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.

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 show 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.
Figure 7: Tool Generated Word frequency table cc3.csv

Figure 8: Histogram of dataset3

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

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


