International Journal of Applied Sciences & Engineering (IJASE) 1(2): October, 2013: 69-73

Three Phased Component Retrival Technique (TPCRT) for Best Qualified Component

Vishnu Sharma [1] and Vijay Singh Shekhawat[2]

[1]Research Scholar, Mewar University, Chittorgarh, India
[2]Shri Karni College,Vaishali Nagar, Jaipur, Rajasthan, India

Email: vishnushree2008@gmail.com

Abstract

The focus of this paper is to suggest a efficient component retrieval technique. Here a combined architecture of three search techniques from traditional (Keywords based) to latest approach (deductive search) is used to get best qualified component. This approach is useful for the software developers to get the appropriate components to develop efficient software within a short span of time. It also provides an efficient way to retrieve appropriate component from repository. The suggested design effectively supports query specification and component search. It further guides users to exploit component resources for reuse.

Keywords: Component, CBSE, Search, reuse, Three Phased Component Retrieval Technique (TPCRT), Product Line Architecture, Automated Theorem Prover (ATP), Product-line.

Introduction

In today's scenario most of the businesses are doing component based production. On the same pattern to augment the productivity, quality and efficiency and diminish efforts done on testing component based software engineering has taken its market place. Components are piece of code which is ready to embed in our software without disquieting about the failure. Developing any complex software from scratch is expensive, slow and error prone as well. If the development task is not performed in a systematic way without testing in early stages of SDLC, it may easily get out of control making it almost impossible to debug and even more difficult to modify the code. Still problem to get the best qualified component is troubling the developers. In this paper a 3 phased mode for component retrieval is introduced.

Software Component Repository

Software components enhances the capability of software The software architecture is one of the main artifacts developed during the software life cycle [1] because it determines most of the non-functional characteristics the resulting software will have, and it is also one of the most difficult documents to change once the software is deployed [2]. Component-Based software engineering is the key technology to cope with the requirements of high productivity, low maintenance cost and reliability of software products [6]. Software product lines are a trend for planned colossal reuse of software assets [3]. The most typical reusable assets are software components, but we can also reuse the product line architecture (PLA), software requirement documentation, and test cases, among others. The PLA is an important reusable asset because all software products in the family share the same design [4]. Therefore, the PLA design should be carefully approached making sure it will produce software that complies with the desired requirements.

There are three major areas in software engineering which has to be focused when considering the components for software reuse. These are described as a) classifying the components needed .b) describing the components wanted .c) finding the appropriate components. Finding the component includes a major area of search techniques and retrieval techniques. In this research paper we will try to provide a framework to get best qualified component from the repository using semantic search.

Component Retrival Techniques

Few traditional approaches to get best qualified component are :

1. **Keyword search** requires assigning to each object a number of relevant keywords or indices [5].
2. **Full-text Retrieval** : when a person wants information from that stored collection, the computer is instructed to search for all documents containing certain specified words and word combinations, which the user has specified [5].
3. **Hypertext Search:** The basic building blocks in hypertext are nodes and links. Each node is associated with a unit of information, and nodes can be of different types[5].
4. **Enumerated classification**: Enumerated classification uses a set of mutually exclusive classes, which are all within a hierarchy of a single dimension [6].
5. **Attribute value**: The attribute value classification scheme uses a set of attributes to classify a component [6].
6. **Faceted:** Faceted classification schemes are attracting the most attention within the software reuse community [6, 7].
7. **Signature matching :** Consider the signatures presented in Figures 1 and 2 for a stack of integers and a queue of integers, respectively [8]

Deduction-based software component retrieval uses pre and post conditions as indexes and search keys and an Automated Theorem Prover (ATP) to check whether a component matches. This idea is very simple but the vast number of arising proof tasks makes a practical implementation very hard. We thus pass the components through a chain of filters of increasing deductive power. In this chain, rejection filters based on signature matching and model checking techniques are used to rule out non-matches as early as possible and to prevent the subsequent ATP from “drowning.” Hence, intermediate results of reasonable precision are available at (almost) any time of the retrieval process. The final ATP step then works as a confirmation filter to lift the precision of the answer set. We implemented a chain which runs fully automatically and uses MACE for model checking and the automated prover SETHEO as confirmation filter. We evaluated the system over a medium-sized collection of components. Whenever different ways for achieving a objective keep their pros and cons than a perfect solution is, which keeps pros of all of these. It is mixed approach

Three Phased Component Retrival Technique (TPCRT)

In this research paper phased component search is proposed to get best qualified component. 3 most popular component retrieval search algorithms works together in a sequential manner. If one search is not able to get the component. Next phase is done with the search results and further filtration is done. If still objective is not achieved 3rd phase is completed.

Phase- 1

Keyword Search

Free-text keyword approach is an automatic process. In faceted index approaches, keywords from program descriptions and documentation are extracted by experts, and keywords are arranged by facets into a classification scheme, which is used as a standard descriptor for software components. To remove ambiguities, a vocabulary is derived for each facet to make sure that the keyword matched can only be within the facet context.

a) *Building Component Library* Software Components are stored in the form of component files. Associated to each component file, an index table is maintained.

Sr Component file Keywords showing functionality

1. File1 KW11, KW12, KW13, KW14
2. File2 KW21, KW22
3. File3 KW31, KW32, KW33
4. File4 KW41, KW42, KW43, KW44, KW45

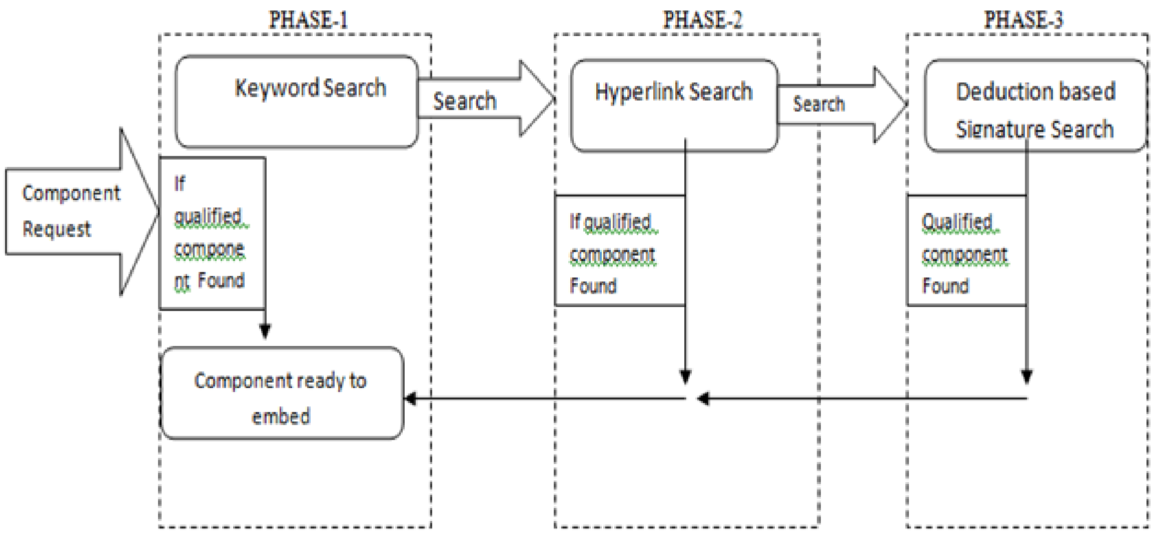
PHASE-1
PHASE-2
PHASE-3
Keyword Search
Search
Hyperlink Search
Search
Deduction based Signature Search
Component Request
If qualified compone nt Found
If qualified component Found
Qualified component Found
Component ready to embed

Fig. 1: Three phased component retrieval technique for qualified component model (TPCRT)

For searching, a search function based on keyword is used to retrieve the required component. Input to this function is a specification (multi word query) given by the users. Search function returns the component file name from the table. A link is established from returned file name to component file in the library. Faceted classification and retrieval has been proved to be very effective in retrieving reuse component from repositories, but this approach is labor intensive. Keyword search may provide more than 1 results. These results may be further send for next step if still appropriate search is not achieved.

Phase-2

With results of the first step hyperlink Search algorithms are applied. Hypertext-based search enables users move through a hypertext document by following links. It provides one of the newest forms to organize documents by using nodes and links (Conklin 1987).Nodes are associated with information blocks, and different types of links represent the different relationships between the source and destination nodes. Users are not constrained to the linear order of conventional documents any more, and navigate through documents at will, which may lead to a more flexible search process.

Hypertext-based search, navigates the repository by following the predefined links. The hypertext technique can be combined flexibly with other search techniques as a complement to empower information retrieval process. For example, the prototype ORCA and AMHYRST designed for an integrated CASE environment (ICE) (Isakowitz and Kauffman 1996) presents a proposal to combine automated classification and hypertext in a tool that provides navigational capabilities for repository objects [9].

Hypertext-based search enables users move through a hypertext document by following links. It provides one of the newest forms to organize documents by using nodes and links (Conklin 1987).Nodes are associated with information blocks, and different types of links represent the different relationships between the source and destination nodes. Users are not constrained to the linear order of conventional documents any more, and navigate through documents at will, which may lead to a more flexible search process. However, such a free structure of search is subject to the potential that users get lost in the detail of information that can be accessed (Nieslen 1990).

To avoid user’s missing, a constraint to limit users’ selection should be deployed in a search process. Search techniques continue to advance, integrating new and traditional searching methods. An effective search tool should accommodate continually expanding collections, a characteristic of most information repositories. We have reviewed four search techniques. The first three, keyword search, full text search, and classification based search, are index based search in a structured repository, while the last one, hypertext-based search, and navigates the repository by following the predefined links. The hypertext technique can be combined flexibly with other search techniques as a complement to empower information retrieval process. For example, the prototype ORCA and AMHYRST designed for an integrated CASE environment (ICE) (Isakowitz and Kauffman 1996) presents a proposal to combine automated classification and hypertext in a tool that provides navigational capabilities for repository objects. The automatic indexing feature of a full text search has proven that it can be widely used in text-intensive documents like articles and books. Meanwhile, as keyword search and classification based search predefine the keywords or catalogues used as an index to describe concepts relevant to the domain of discourse, they introduce semantic information absent in full text search. Each search technique has advantages and is appropriate in specific circumstances. A suitable search technique should be proposed based on the features of information repository.

Phase-3

This is a optional phase if component is not searched after phase 2 than this is the closing phase . Zaremski and Wing proposed a component retrieval method based on matching the signatures of the operations. This method describes the behavior of an operation by a formal specification language Larch/ML. However, this method only describes the behavior based on the terms appearing in the operation's signature. This method does not allow the semantics of a component to be described completely. In fact, research on retrieval methods has always focused on theories for verifying the match between a specification and a query. Retrieval methods rarely address the question of how to describe the semantics of a component completely. In this phase deductive search of Artificial Intelligence is applied. NORA/HAMMR may be used as a filter pipeline through which the candidates are fed. This pipeline typically starts with signature matching filters. They check whether candidate and query.

References

Guru C.V., Rao and P.Niranjan **"**An Integrated Classification Scheme for Efficient Retrieval of Components". Journal of Computer Science 4 (10): 821-825, 2008 ISSN 1549-3636 © 2008 Science Publications

Jan Bosch. Design and Use of Software Architectures. Adopting and Evolving a Product Line Approach. Addison Wesley, first edition. May 2000.

Johann Schumann "NORA/HAMMR: Making Deduction-Based Software Component Retrieval Practical".

Len Bass, Paul Clements, and Rick Kazman. *Software Architecture in Practice.* SET Series in Software Engineering. Addison-Wesley, 2nd edition, 2003.Immense

Martin Fowler. Who Needs an Architect? *IEEE Software,* 20(5):1H3, 2003.

Niranan, P. and Guru Rao, C.V. "A mock-up tool for software component reuses Repository"

Paul Clements and Linda M. Northrop. *Software Product Lines: Practices and Patterns.* Addison Wesley, first edition, August 2001.

Rajender Nath, Harish Kumar; Building Software Reuse Library; 3rd International Conference on Advanced Computing and Communication Technology- ICACCT-08; Asia Pacific Institute of Information Technology, Panipat, India; November 08-09, 2008, pp. 585-587.

Tomas isakowitz and Robert j. Kauffman "Supporting search for reusable software objects. Center for Digital Economy Research Stern School of Business Working Paper IS-93-47

Zheying Zhang "Enhancing Component Reuse Using Search Techniques" Department of Computer Science and Information Systems University of Jyväskylä, PL 35, FIN-40351 Jyväskylä, Finland