International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2018 IJSRCSEIT | Volume 3 | Issue 8 | ISSN : 2456-3307
DOI : https://doi.org/10.32628/CSEIT183836

Recent trends in Component Based software development and Efficiency analysis of Semantic search based component retrieval Technique

Vishnu Sharma[1], Prof. Vijay Singh Rathore[2]
[1]Research Scholar, Mewar University, Chittorgarh, Rajasthan, India
[2]Professor & Head, JECRC Engineering College, Jaipur, Rajasthan, India

ABSTRACT

In these days most of the software development uses preexisting software components. This approach provides plenty of benefits over the traditional development. Most of the software industries uses their own domain based software libraries where components resides in the form of modules, codes, executable file, documentations, test plans which may be used as it is or with minor changes. Due to shrinking time and high demand of software development it is necessary to use pre tested software components to ensure high functionality in software developed. Software components can be used very easily and without having the worries of errors and bugs because these are developed under expert supervision and well tested. What we have to do is just embed these components in our project. In this paper a survey got conducted over 112 software developer,testers and freelancers. In survey several issues in CBSD were identified. An efficient repository along with a component search engine is developed. All the component retrieval techniques were evaluated and compared with precise and recall method.

Keywords : CBSD-Component Based software Development, COTS-Component Off The Shelf,

I. INTRODUCTION

Component Based software development is an efficient approach to develop quality softwares . Software industries are using these components with a great extent and serving the industry with reliable software. Component Based development is not too popular among a group of software developers because with CBSE lot of the problems are faced by the developers which waste the resources and time of the developers. Best qualified component search is a very tedious task. This paper exert a pull on these kind of problem .

Component based software development life cycle

Component based development has to gone through from certain stages for successful completion. CBSE has the following phases[1]

1. Component Adaption

This step ensures that architectural mismatches are not present. Since development of each component targets diverse requirements, considering different expectations about the working environment, adaptation is aimed to make sure that divergence among these segments is minimum and it also ensures specific attributes of the components or the system. There below are a lot of adaptation techniques:

Component properties may be modified with parameterized interface with specified parameters, these are essential part of component interface; Wrappers to introduce encapsulation feature of components &it provides many new interfaces that prohibits or extend available interfaces, or adding particular properties; writing of adapters to change component interfaces for ensuring its compatibility with their interfaces of diverse components. These various ways are depending on the accessibility of the internal structure.

2. System Integration

System Integration phase is used for standard infrastructure components and application components integration. This integration gets completed at different levels: particular components level and the entire system level. Particular component integration into the new system is known as component deployment.

3. Verification and Validation

Verification and Validation are concerned with standard test and verification techniques. Components are black-box types (Internal structure is hidden) and these are sold from different vendors, hence it is very hard to discover the errors. Contractual interface specifications is very important in this situation which facilitate testing of proper input and output.

In IusWare [27], an model is devised which facilitates verification and validation activities with verifying the continuity of the prescribed model and their components. While using this model, different features of component are evaluated. These measures are transformed on values on criteria and these values are combined together for recommendation. PORE [24] defined 3 models (requirement, product and compliance) to get a negotiation between customer requirements and product features. CARE [26] introduced the world model which describes the environment, the system model represents system capabilities: both functional and nonfunctional, and the interface model this maps system goals and component goals and specifications). Component evaluation is done on the basis of these three models.

4. System Maintenance

System Maintenance is used broadly for replacement of old component with new components or addition of many new blocks or components into available systems. This approach of the maintenance method is equal to this for this development: search a proper block or component, if found evaluates it, make adaption, if essential, it integrates the same into the system.

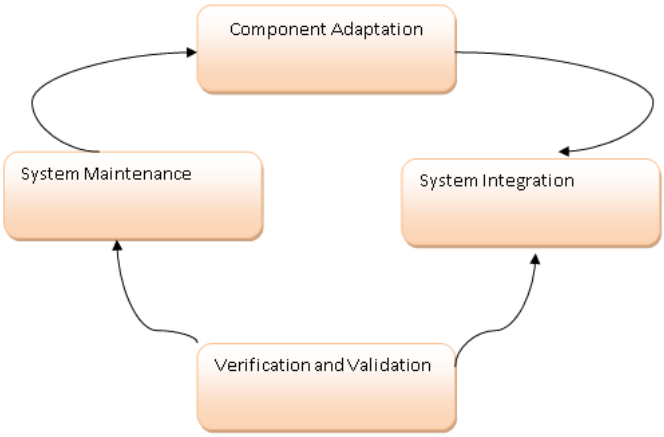


Figure 1. Component Based Development Life Cycle

II. Advantages Component Based Software Development

1. Recompense of Component Based Software Development

CBSD has various benefits that reduces the software development life cycle and enhances efficiency:

- ✓ **Improved Maintainability** – Component based software development make software easy to maintain, since regressions are condensed as components interaction is done through specific and explicit services of interaction. CBSD enhances modularity

- ✓ **Promotes Distributed Development** – Components can be developed in distributed fashion with different teams without dealing with the intricate branching and integration confront of a complex application. Specified interfaces builds a clear boundary to every component that provide freedom to work in parallel.
- ✓ **Reduced Release & Deployment Cycles** – CB software supports releases of components independently without synchronize these release cycles.
- ✓ **Easy roll-out** – CB software applications can be rolled out globally more easily, since they allow deployment of a intermingle of generic and localized components that communicate together.

2. Risks and challenges of component-based software development

Components are developed in two scenarios.
Developers either develop components for the mass market

1) For specific clients.
2) For personalized Component Library

Developing for the mass market requires developer's to research and identify business domains that would produce good returns which would justify component development for them[2]. Once the domain has been identified, their next challenge is to come up with components that have the entire domain knowledge and processes build into them which is not easy considering the ever changing business domains[3]. Keeping a pace with the changes could alone prove quiet challenging. Then there is always the risk associated with a component repository getting obsolete due to unforeseen industry trends and managing changes. The developers risk the investment made in terms of effort and finances. Going the tried and tested 'legacy' systems way by giving the legacy systems a component wrapper would also require a lot of effort[4]. The developers face the challenge of verifying that the domain knowledge and processes correctly map into their components. Developers also face the challenge of choosing the correct tools for developing components. What make the choice so difficult are factors like price, performance, middleware required etc.

Project management of CBSD projects is also challenging as compared to traditional software projects.

- ✓ Its and very tedious task to search appropriate component available with repositories. Component search techniques provides a vast range of components without any customization option[5].
- ✓ CBSD projects require proper monitoring and coordination at every step.
- ✓ Version control is of great importance considering multiple versions of components are released for various projects, coordinating and informing clients of new releases is also challenging.
- ✓ Developers need to unit test their components without knowing how their developed product will be put to use.
- ✓ Developers need to thoroughly analyze the cost and benefit before undertaking any client projects or developing for the mass markets. Intellectual [6]
- ✓ Property rights and licensing issues need to be resolved in addition to the above mentioned when developing for the mass markets.
- ✓ A reliable repository is also an issue.

3. Current status of component based development :

To know the current status of software reuse one of the eminent survey was done on world's renowned

organizations involve in maintaining component libraries and integrating existing software components and organizations which were not reusing components, or component builders, as the respondents. All the participants for this survey were from renowned software companies. We conducted a survey over 112 software developers, testers, analysts and Project Managers of world renowned companies like Erricson Infosys, Cognigent, Infogain,Apperio etc. and freelancers also. We tried to explore the latest trend in software development. We divided the whole developers in 4 separate groups based on their responses.

1. First group was of the developers use software components for software development at max.
2. Second group still believes to make complete software at their own
3. Third group was of the developers who said use of component depend on the complexity and availability of components.
4. Fourth and last group itself develops software components for reuse and use them.

Outcome of the survey

During the interaction with the respondents it was noticed that most of them are facing few problems that why they sometimes avoid using components .

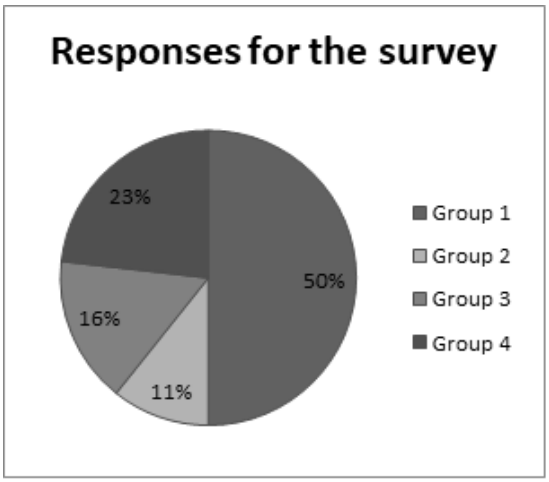
A. Problem in searching qualified component.
B. Component search retrieves plenty software components whose functionalities cannot be recognized without using it.
C. Text based search is mostly used which results lot many confusing text results.
D. None of the component repository is able to provide a component repository which can provide efficient search with minimum investment of time and efforts.

This research paper focuses on the problem of software component retrieval method.

Survey results were as under:

Table 1

Total Participants	**Group 1 (developers use software components for software development at max)**	**Group 2 (Believes to make complete software at their own)**	**Group 3 (use of component depend on the complexity and availability of components.**	**Group 4 (develops software components for reuse and use them)**
112	56	12	18	26



Reusing of component reduces the overall effort and time required to develop the whole system. We can state that reliability and timeliness may be surely obtained with the use of Components off the shelf(COTS)[6]. This survey was focused on the software developers, testers, architects. This survey provide an insight in how Component Based Software Engineering helps software reuse in current scenario, it also demonstrates scope of software components, their verification in isolation[7], and

how does users test components and evaluate their applicability prior to selecting them.

III. Popular Techniques for Component Retrieval

On the basis of the survey conducted it has been observed that still component based development is not adapted everywhere due to following reasons:

- It is not easy to find out qualified component for adaption.
- Keyword base search provides too many results so that lot of time is consumed to judge which one is compatible with the application. It totally waste the time[8].
- Signature based search also not able to attract the developers to switch towards the CBSD[9].
  So here is requirement of any other more precise component search methods

IV. Semantic Search for Component Retrieval

Semantic search is an efficient method which improves most qualified component search accuracy by understanding the component searcher's objective and the meaning of terms as they seems to be in query specification, either on the Web or a closed system, to generate more significant results[10].

In this paper we have tried to retrieve components from component repository with the frequently used techniques [11] :

V. Implementation of the Component Retrieval Framework

For evaluation of the proposed framework a Component Repository for Semantic Search (CRSS)is built with .net framework. Having functionalities as shown in figure 3

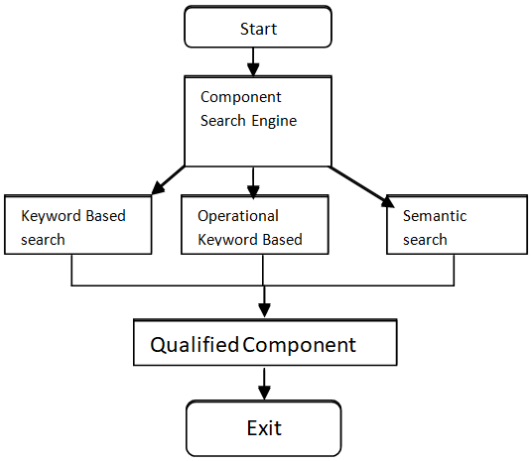


Figure 2 : Evolution of Component Search Engine

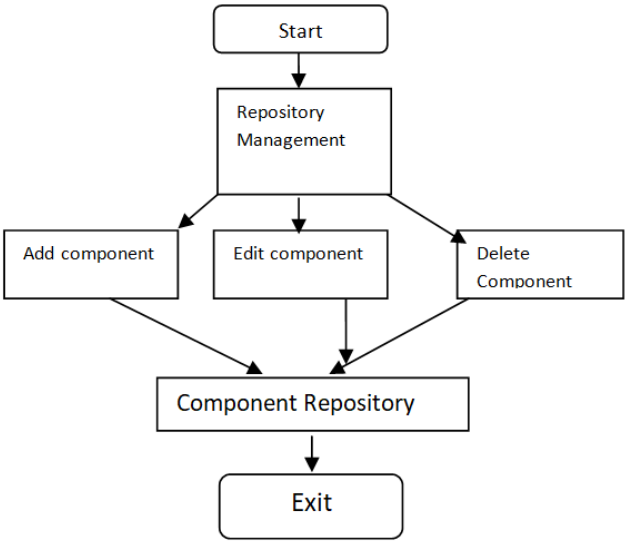


Figure 3 : Component Storage Management

Components from CRSS is retrieved with different available techniques and finding were evaluated with Precise and Recall method

VI. Evaluation of Results

Precise and Recall method :

- ✓ Precision is the number of document retrieved that are relevant and Recall is the number of relevant document that are retrieved.
- ✓ Relevance is how useful the information presented is to the topic being discussed at the moment.

✓ Precision and Recall are both extremely useful in understanding what set of documents or information was presented and how many of those documents are actually useful to the question being asked.

Precision and recall of Keyword Based Search

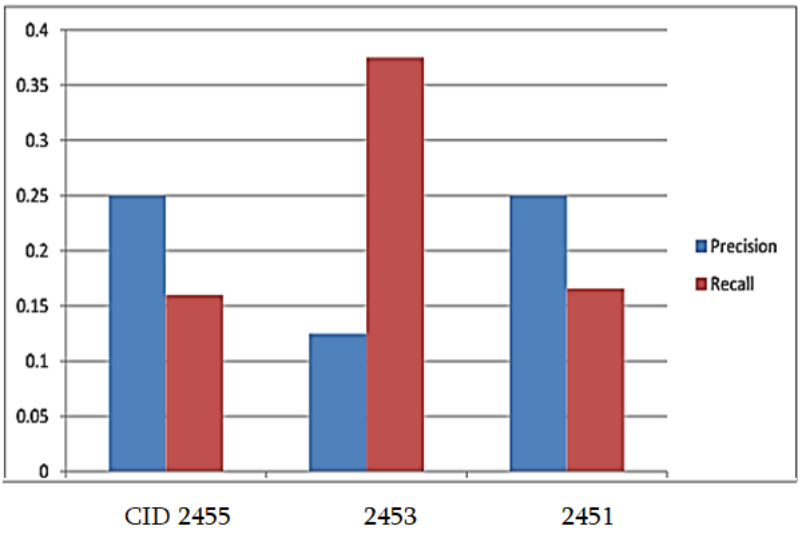


Figure 3. Graph showing the precision and recall of Keyword Based Search

Table 2. Components selected for the retrieval

CompID	Comp Name	Component Model	Component Type
2455	Calculator	Java Bean	Sorce Code
2453	Calculator	Corba	Dll
2451	Calculator	Com	Dll

Table 3. Keyword Based Search results

CompID / Method	2455	**2453**	**2451**
Precision	0.25	**0.125**	**0.25**
Recall	0.16	**0.375**	**0.166**

Results obtained for operational Semantic retrieval

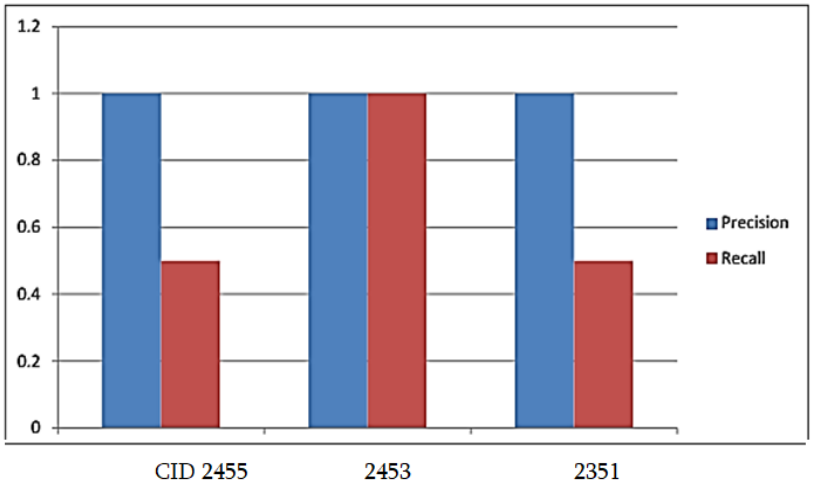


Figure 4. Graph showing the precision and recall of operational Semantic retrieval

Table 4. Components selected for the retrieval

CompID / Method	2455	2453	2451
Precision	1	**1**	**1**
Recall	0.5	**1**	**0.5**

Table 5. Result comparison on the basis of Precision and Recall

Techniques / Comp Id	Keyword Based Retrieval		Operational Semantic Retrieval	
	Precision	Recall	**Precision**	**Recall**
2455	0.25	0.16	**1**	**0.5**
2453	0.125	0.375	**1**	**1**
2457	0.25	0.166	**1**	**0.5**
Mean	0.208	0.233	**1**	**0.66**

Comparison between different component retrieval techniques (Precise)

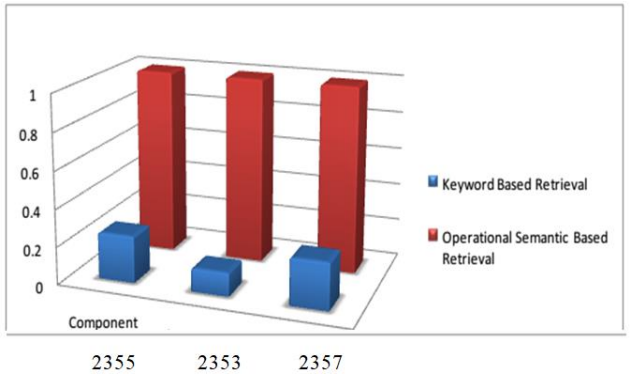


Figure 4 : Comparison on the basis of precision between the techniques(Precise)

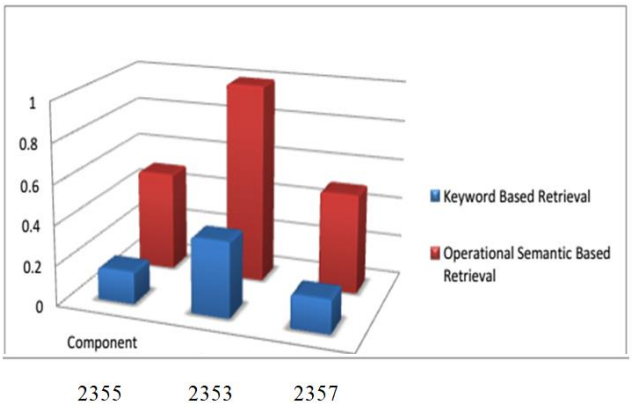


Figure 5 : Comparison between different component retrieval techniques (Recall)

VII. CONCLUSION

With the results obtained it is found that Semantic based component retrieval techniques is best for best qualified component retrieval. Semantics may be enhanced with the availability of new more specialized properties. It will not only increase the belief of the software professionals in component based development but also give a hike in software productivity. Qualified component search will reduce the time and efforts. CRSS is equally important as it will let software professional for add component and download components for better software development time.

In this paper, we have presented a survey of reusability in current scenario and different types of reuse that can reduce the overall time for development and effort and the main important cost of the application, because we need not to have to develop all the parts of the application from the scratch. A reuse plan should always be premeditated, done carefully and systematic approach should be adopted to give the higher payoff. Software reuse policy is adopted in any organization for quality improvement and productivity of the software application. The measurement of quality and productivity is done with the use of metrics. Metrics is a efficient quantitative indicator of an attribute. For utilizing the benefits of reusability, domain knowledge is mandatory. Without appropriate knowledge of what where to use the component of an application cannot provide us qualitative product with efficient time. So Domain engineering plays an significant role in software component reusability. Domain knowledge is the key concept of systematic reuse.

VIII. REFERENCES

[1]. Lakshmi Narasimhan, P. T. Parthasarathy, M. Das,"Evaluation of a Suite of Metrics for Component Based Software Engineering (CBSE)", Issues in Informing Science and Information Technology, Vol. 6, pp. 731-740, 2009.

[2]. B.Jalender, Dr A.Govardhan and Dr P.Premchand. Article: Breaking the Boundaries for Software Component Reuse Technology. InternationalJournal of Computer Applications 13(6):37-41, January 2011. Published by Foundation of Computer Science.

[3]. N. Md Jubair Basha, Salman Abdul Moiz, "Component based software development: A state of art", Advances in Engineering Science and Management (ICAESM) 2012 International Conference on, pp. 599-604, 2012.

[4]. Jian Li Dong, "Research on Heterogeneous Component Assembly Problems Based on Software Product Line", Advanced Materials Research, vol. 765-767, pp. 1324, 2013.

[5]. I. Crnkovic, B. Hnich, T. Jonsson, Z. Kiziltan, Specification, implementation, and deployment of components, Communications of the ACM 45 (2002) 35-40

[6]. A.I. Mørch, G. Stevens, M. Won, M. Klann, Y. Dittrich, V. Wulf, Component-based technologies for end-user development, Communications of the ACM 47 (2004) 59-62.

[7]. A. Bracciali, A. Brogi, C. Canal, A formal approach to component adaptation, Journal of Systems and Software 74 (2005) 45-54.

[8]. 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

[9]. 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.

[10]. 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.

[11]. Vishnu Sharma,Vijay Singh Rathore; ;Three Phased Component Retrival Technique (TPCRT) for Best Qualified Component; International Journal of Applied Sciences & Engineering (IJASE) 1(2): October, 2013: 69-73

[12]. Vitharana, Padmal. (2003). Risks and challenges of component-based software development. Commun. ACM. 46. 67-72. 10.1145/859670.859671.