An Experimental Study of Algorithm for Selection of Components in Component Based Software Development (CBSD) Environment

Jyoti Sharma and Arvind Kumar
SRM University, Delhi-NCR, Sonipat − 131029, Haryana, India; kaushik.jyoti27@gmail.com, k.arvind33@gmail.com

Abstract

Objectives: Component Based Software Development (CBSD) approach is used to provide reusability and it is also used to increase productivity, reliability. The research paper discusses the optimized way of selecting those components which are having the complexity greater than or equal to 5. Calculation of specific software complexity is by Optimal Component Selection (OCS algorithm). Estimation of component reusability is also calculated using this approach. Method: when the components are gathered according to the requirement template then cyclometric complexity will be calculated and if it is comes to be higher than 5 then OCS is applied to find the specific complexity or to give that whether the component is reusable or not. Findings: Now a day's people prefer the CBSD approach for the production of software because it is considered to be better to use the reusable components instead of developing the software from the scratch. The proposed idea will support this adaptation by considering the components having complexity greater than or equal to five. Applications: The proposed idea is validated against the C and java coded components and it can be improved by developing a framework that will work for all the components taken from online repository.

Keywords: Component Based Development Environment (CBSD), Cohesion and Optimal Component Selection (OCS), Component Selection, Reusability

1. Introduction

Today the software applications are large and very difficult to develop so due to this it has been packaged encompasses with a variety of advantages or features. The benefits of CBSD are to produce the quality software with a greater reliability, reusability, productivity with low cost and less number of efforts. The component based development process is having the following stages viz. Requirement engineering, Analysis and design, Implementation, Test, Release and Maintenance.

1.1 Requirement Engineering

Requirement engineering is the first step of a Software development process. According to the requirement document template the components are selected from the component repository and other sources as shown in Figure 1.

![Figure 1. Constructing software from component repository.](image)

1.2 Analysis and Design

The analysis phase is just to analyse or to check the particular component that whether it is according to requirements or not. Design phase is used to check the
difference between the required design and the possible design of a component and this analysis has a great impact on the system quality properties.

### 1.3 Implementation

For the component based system the implementation part requires the less amount of effort as compared to the non-component based system. It is the code which is generated automatically and helps the components to connect and establishes a communication between these components.

### 1.4 Test

Test phase is all about the testing of the components according to functionality required for that component. This test phase cannot be considered as a very important phase because different components are in different purposes and also they behave differently in various conditions.

### 1.5 Release

Release phase is just to deliver the final software product to customer.

The categorization of paper is as follows: Section 2 describes the literature review part for the CBSD. Component selection framework has been described in section 3. Section 4 makes readers aware of working of referenced algorithm. Section 5 provides the results and discussion. Section 6 concludes the given paper, followed by references.

Several researches have been performed by researchers regarding the component selection, component integration, component architecture out of which few are referenced. The method in which the component selection is done by using the various types of clustering like fuzzy c-means clustering and subtractive clustering. The new component based software development process for improving the reusability factor. In proposed a system in which there is a way of defining the component architecture and this identifying architecture is used in the component selection decision for the software production. Explores the need of the concept of reusability and he strongly recommended that CBSD is a good approach for software development. Proposed search engine interface for selection of components. It helps to identify the components but did not use any approach to retrieve those components. Propose integration of components using type collection. It was based on java but does not support other software like Visual basic. A novel model for the component selection is to maintain compatibility among them. They used genetic algorithm approach to select multiple components at same time. Describe Object Oriented Software Development life cycle that consists of various processes.

![Diagram of Component Selection Framework](image)

**Figure 2.** Layout of component selection framework.

Made use of genetic algorithm to retrieve components from repository but did not produce results on integrated environment systems. Proposed the approximation algorithm for component selection using the NP Complete
and greedy approach\(^2\). Propose a SDLC model which is useful for developing the software by the use of existing components rather than considering the component from the very scratch. They emphasizes on the reusability factor used in component based software engineering.

2. Component Selection Framework

Figure 2 gives outline view of system for selecting components in an optimal manner after calculation of cyclometric complexity.

![Figure 3. Home screen.](image)

![Figure 4. Uploading of a component.](image)

It starts according to the requirement documentation template.

It is accompanied by creation of software repository or database which is called as the component repository having the external cots components as well as the internal in-house components.

After the system and software design the calculation of cyclometric complexity, advance matrix calculation, number of independent paths is calculated as shown in Figure 3-7.

- Cyclometric complexity is directly related to programming complexity of module or component. If the calculated cyclometric complexity is greater than 10 then the structure of module or component is considered to be complex and it may be discarded but if the value is equal or greater than 5 then the structure of module is considered to be less complex and it may be tested again for the purpose of component selection.
- If the cyclometric complexity comes out to be equal or greater than 5 then the OCS algorithm
An Experimental Study of Algorithm for Selection of Components in Component Based Software Development (CBSD) Environment

will be applied. After applying algorithm, various parameters like DUM (Direct connection between classes and methods) etc. are being calculated.

- It leads to final selection of components having complexity always less than 5 and those components can be considered as the optimal components.
- The last step is the integration of components and then release.

A. Optimal Component Selection (OCS) Algorithm

![Figure 7. Showing the independent number of paths in a component.](image)

3. Implementation

The Figure depicts the working of our proposed approach with OCS algorithm.

Platform used: Microsoft Visual Basic 2010

4. Results and Discussion

Consider a single component named triangle.c for uploading on developed framework. Below is snippet of one of classes C1 for showing the complexity calculations.

**CheckCust.java**

```java
#include<stdio.h>
#include<conio.h>

int main(int argc, char **argv)
{
    clrscr();
    for(i=0;i<3;i++)
    {
        printf("enter the side[%d]:",i+1);
        scanf("%d",&aiside[i]);
    }
    if(aiside[0]+aiside[1]>aiside[2]&&aiside[0]+aiside[2]>
    aiside[1]&&aiside[1]+aiside[1]>aiside[0])
    printf("valid triangle");
    else
    printf("invalid triangle");
    getch();
}
```

4.1 Calculations

Here, total numbers of components or packages or header files in the given example are = 2.

So the value of \( m = 2 \)

\[
NDIUC = m (m-1)/2
\]

\[
= 2(2-1)/2
\]

\[
= 1
\]

\[
DUM = 4 \text{(as number of direct connection between classes and methods is 3)}
\]

\[
LC2P = m \times DUM/NDIUC
\]

\[
= 2 \times 4/1
\]

\[
= 8
\]

\[
IUM \text{ (Set of all the indirect connection between classes and method is 1)}
\]

Therefore \( IUM = 2 \)

\[
HC2P = m \times [DUM U IUM]/NDIUC
\]

\[
= 2 \times [4 U 2]/1
\]

\[
= 12
\]

Now, value of \( LC2P = 8 \)

Value of \( HC2P = 12 \)

Therefore the low value of \( LC2P \) leads to selection of that particular component.

5. Conclusion and Future Scope

The given paper makes readers aware of optimal component selection in phase of software development. If the cyclometric complexity calculation gives a value greater than or equal to five then the component cannot be immediately rejected, the OCS algorithm can be applied for the better results of component selection that whether the component may be selected or not. It is followed by
its implementation and validation by taking single classes into consideration. The results show higher value of HC2P which is indication of optimal component selection.

As future scope, it can be extended for the components having cyclometric complexity greater than 10 which is McCabe cyclometric complexity number.

6. References

1. Kaur J, Tomar P. Validation of Software Component Selection Algorithm based on Clustering. Indian Journal of Science and Technology. 2016 Dec; 9(45):1−4. Crossref.
2. Gill NS, Tomar P. X Model: A New Component-Based Model, International Journal of Engineering and Technology. 2008 Aug; 1(1-2):1−10.
3. Mancebo E, Andrews A. A Strategy for Selecting Multiple Components. In: SAC March '05: Proceedings of the 2005 ACM Symposium on Applied Computing; New York. p. 1505–10. Crossref.
4. Pressman RS. Software Engineering: A Practitioner's Approach, 7th Edition, McGraw Hill. 2006.
5. Khemakhem, Drira SK, Jmaiel M. SEC: A Search Engine for Component Based Software Development. Proc. ACM Symposium on Applied Computing; Dijon: France. 2006. p. 1745−50.
6. Washizaki H, Hoshi D, Yoshiaki F. A Flexible Connection Model for Software Components, IEICE-Transactions on Information and Systems. 2006 Apr; 4:1421−31. Crossref.
7. Tang JF, Mu LF, Kwong CK, Luo XG. An Optimization Model for Software Component Selection under Multiple Applications Development, European Journal of Operational Research. 2011 Feb; p. 301−11. Crossref.
8. Parthipan S, Velan SS, Babu C. Design Level Metrics to Measure the Complexity Across Versions of AO Software, IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCT); 2014. p. 1708–13. Crossref.
9. Dixit A, Saxena PC. Software Component Retrieval using Genetic Algorithms. In Procedure: International Conference on Computer and Automation Engineering; 2009. p. 151–55. Crossref.
10. Habibi J, Haghpanah N, Kargar M. Approximation Algorithms for Component Software Selection. 14th Asia Pacific Software Engineering IEEE Conference; 2007. p. 159–66.
11. Sharma J, Kumar A. Optimal Component Selection Algorithm for Appropriate Software Component Selection, Asian Journal of Mathematics and Computer Research. 2017 Feb; 15(2):71–80.
12. Jain P. Towards the Adoption of Modern Software Development Approach: Component based Software Engineering, Indian Journal of Science and Technology. 2016 Aug; 9(32):1−5. Crossref