Optimal Component Selection of COTS based Modular Software System under Cohesion & Coupling Quality Metrics with Budgetary Constraint

Indumati¹, Vijay M. Ghantasala² and P. C Jha³

¹,³Department of Operational Research, University of Delhi, Delhi
²DST Worldwidesserices, Hyderabad

ABSTRACT

Software systems, especially knowledge based systems become increasingly large and complex in nature which is quite a challenging task for the software industry. Software complexity and software quality measurements are two important tasks needed to be performed for development of good software. Modular software system serves as a better solution for software due to complexities and size of the functional requirements. Several components altogether work to construct each module. So the basic problem is to select components in such a way that good quality software can be developed. Coupling and cohesion are two software quality metrics that captures at least eight out of thirteen quality characteristics. In this paper, a Bi-criterion problem is discussed in which maximization of two simultaneous objectives, functional performance & ICD of Components Based Software System is dealt with respect to multiple constraints of functionality, ICD and budget. The problem is demonstrated with a numerical illustration.

KEYWORDS

Intra-modular coupling density (ICD), Coupling, Cohesion, CBSS (Components Based Software System)

1. INTRODUCTION

Software industry is facing one of the challenging problems of development of an effective large-scale software system. Concepts and methods, such as abstract data types, structured programming, object orientation, design patterns, and modeling languages, have been employed to improve the effectiveness of software system development (Gamma, Richard, Ralph, & John, 1994). Component-based software system (CBSS) development focuses on the decomposition of a software system into functional or logical components with well defined interfaces. It allows a software system to be developed using appropriate software components. In recent years, a component-based approach to software system development has become more and more popular (Szyperski, Gruntz, & Murer, 2004). Some previous studies attempted to define criteria and develop metrics for software modularity. Abreu and Goulão (2001) proposed a quantitative approach to measure and assess the solutions of software modularity based on the criteria of minimal coupling and maximal cohesion. Coupling measures the interactions among software modules while cohesion measured interactions among the software components which are within a software module. A good software system should possess software modules with high cohesion and low coupling. A highly cohesive module exhibits high reusability and loosely coupled systems enable easy maintenance of a software system (Seker, van der Merwe, Kotze, Tanik, & Paul, 2004). (Kwong, Mu, Tang, Luo, 2010) have addressed concept of quantitative way of minimizing coupling and maximizing cohesion for selection of software components for software modules in CBSS development, they have used heuristic technique of Genetic Algorithm for the same. In this paper, we have introduced fixed budget constraint in the system. We have solved this more complex Integer Non-Linear Programming Problem directly using Bhatia et. al. approach with Lingo software (Version11) for comparatively smaller sample of components. It’s a complicated multi-objective model based on the criteria of multiple objective fulfillments in one go with maximizing ICD (i.e., minimal coupling and maximal cohesion) of software modules with budget incorporation along with functional performance and ICD constraints.

The rest of the paper is organized as follows: Section 2 discusses the criteria of software components selection for software modules in CBSS development. Section 3 describes the formulation of an optimization model for the selection of software components for CBSS development. An illustrative example and a discussion of the results are presented in Section 4. Finally, conclusions are described in Section 5.

2. CRITERIA OF SOFTWARE COMPONENT SELECTION FOR SOFTWARE MODULES

Abreu and Goulão’s approach (Abreu & Goulão, 2001) yielded the quantitative measures of cohesion and coupling. In their work, intra-modular coupling density (ICD) was introduced to measure the relationship between cohesion and coupling of modules in the design of an object oriented software system as follows:

\[ ICD = \frac{CI_{IN}}{CI_{IN} + CI_{OUT}} \]  

where \( CI_{IN} \) is the number of class interactions within modules, and \( CI_{OUT} \) is the number of interactions between classes of distinct modules. ICD was employed as a criterion to present the ratio between coupling and cohesion in CBSS development.
Referring to Eq. (1), the ratio of cohesion to all interactions within the jth module can be expressed as ICD. However, it can be found if one module contains only one component; the values of ICD for that becomes zero. To make up for the deficiency, (Mu, Tang, Luo, 2010) simply add 1 to the numerator of Eq. (1) to form another measure of ICD as follows:

$$ICD_j = \frac{(CI_{IN})_j + 1}{(CI_{IN})_j + (CI_{OUT})_j}$$

(2)

where $ICD_j$ is the intra-modular coupling density for the jth module; $(CI_{IN})_j$ is the number of component interactions within the jth module; and $(CI_{OUT})_j$ is the number of component interactions between the jth module and other modules.

3. MODEL FORMULATION

One of the major problems of CBSS development is how to select software components available in markets to formulate a software module. Software components are concrete software products that contain executable program codes and can be provided by various third-party software components providers. The software components should be selected such that the interactions of the software components within a software module are maximized, and interactions of the software components among software modules are minimized.

3.1 NOTATIONS

- **M**: the number of software modules
- **N**: the number of software components
- **L**: the number of sets of software components
- **$S_{ci}$**: i=1,2,...,N, the ith software component
- **$m_j$**: j=1,2,...,M, the jth software module
- **$S_k$**: k=1,2,...,L, a set of alternative components for the kth functional requirements of a CBSS
- **$i \in S_k$**: denotes that $S_{ci}$ belongs to the kth set
- **$S_{cij}$**: the ith software component of jth software module, s.t. $S_{cij} = S_{ci} = S_{cj}$ for all $j=1,2,...,M$
- **$r_{ij}$**: i,j=1,2,...,N, the number of interactions between $S_{ci}$ & $S_{cj}$. As the coupling and cohesion are indirect relations, $r_{ij} = r_{ji}$
- **$f_{ij}$**: i,j=1,2,...,N, the function ratings of $S_{ci}$ to $S_{cj}$, f_{ij} are real numbers ranging from 0 to 1
- **H**: a threshold value of ICD of each module that needs to be set by decision makers.
- **$c_{ij}$**: i,j=1,2,...,N, cost associated with the ith component of the jth module
- **C**: Available budget that needs to be set by software development team
- **$x_{ij}$**: i,j=1,2,...,N, j=1,2,...,M, $x_{ij}$ is the binary variable. $x_{ij}$ = 1, if SC is selected for $m_j$; otherwise 0

(Kwong, Mu, Tang, Luo, 2010) have formulated two objective functions of the optimization problem but in the model discussed earlier, the components were selected without considering the cost of the software modules. In the COTS based development system, acquiring a component either developed in-house or purchased from the market there is a cost associated with that, as for any organization budgets are not unlimited. Developing the software with cost optimization in mind is an important goal of any organization. When we develop the application with finite budgets, it is necessary to utilize the resources in an optimized way without compromising on quality. In order to achieve this we need to have optimization models which will minimize the cost and also ensure that the application runs without failure. With this in mind, a new constraint on cost is proposed with an intention of minimizing the cost of selecting the modules. The cost constraint is as follows:

$$\sum_{j=1}^{M} \sum_{i=1}^{N} c_{ij} x_{ij} \leq C \quad i = 1,2,...,N, \ j = 1,2,...,M$$

(3)

3.3 ASSUMPTIONS

1) Each component has a distinct, known functional rating and functional ratings data is set by the software development team.
2) Atleast one component is supposed to get selected from each module.
3) Interaction data for components is exactly same for each module.
4) A threshold value of ICD is set by the decision makers.
5) The budget is limited.
6) Functional requirements – Exactly one software component in $S_k$ may get selected to implement the kth requirement.

$$Max ICD = \sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{r=1}^{N-r} r_{ij} \cdot x_{ij} \cdot x_{r,j}$$

(4)

$$Max F = \sum_{j=1}^{M} \sum_{i=1}^{N} f_{ij} \cdot x_{ij}$$

(5)

Subject to $S = \{x_{ij}, binary \ variable \ / \ i = 1,...,N; \ j = 1,...,M$

$$\sum_{j=1}^{M} \sum_{i=1}^{N} r_{ij} \cdot x_{ij} \cdot x_{r,j} + 1 \geq H \quad j, j' = 1,2,......,M$$

(6)

$$\sum_{j=1}^{M} \sum_{i=1}^{N} c_{ij} \cdot x_{ij} \leq C$$

(7)

$$\sum_{i=1}^{N} \sum_{j=1}^{M} x_{ij} = 1 \}$$

(8)
Optimal Component Selection of COTS based Modular Software System under Cohesion & Coupling Quality Metrics with Budgetary Constraint

\[
\sum_{i=1}^{N} x_{ij} \geq 1 \quad j=1,2,\ldots,M
\]  
(P1)

Here objective (4) is the ICD function, (5) is the functional requirement of the problem. Constraint (6) refers to ICD constraint. Constraint (7) is the budget constraint which is having pre-fixed value to which budget can get extended. Constraint (8) denotes the functionality constraint and constraint (9) depict at least one software component has to be selected from a set of alternative software components for a particular software module for fulfilling a functional requirement. (Kwong, Mu, Tang, Luo, 2010) have addressed this concept of selection of software components for software modules in CBSS development without budget inclusion and they have used heuristic technique of Genetic Algorithm for the same. Here we are solving it using Bhatia et. al. technique. Let

\[
f(x_j) = \sum_{j=1}^{M} \sum_{i=1}^{N} m_{ij} x_{ij}
\]

The above problem can equivalently be written as following mathematical programming problem using Bhatia (1997).

\[
Max \ F_1(x) = f(x) - g(x)
\]

\[
Max \ F_2(x) = \sum_{j=1}^{M} f_{ij} x_{ij}
\]

Subject to

\[
X \in S = \{x_{ij} \in \{0,1\} / \text{Satisfying eq. (9) to (14)} \}
\]

\[
i = 1,\ldots,N; \quad j = 1,\ldots,M
\]  
(P2)

Using the Geoffrion’s equivalent scalarization formulation (Geoffrion, 1968) of the problem (P3) for fixed weights for the objective function is as follows:

\[
Max \, G(x_{ij}) = (\lambda * (f(x_{ij}) - g(x_{ij}))) + (1 - \lambda) * F
\]

Subject to

\[
X \in S = \{x_{ij} \in \{0,1\} / \text{Satisfying eq. (9) to (14)} \}
\]

\[
i = 1,\ldots,N; \quad j = 1,\ldots,M
\]

\[
\lambda \in \Omega = \{\lambda \in R / \lambda \geq 0\}
\]  
(P3)

The original problem was a Bi-Criteria problem. Using Geoffrion’s scalarization the problem is converted to a single objective problem by attaching weights to both the objectives.

4. NUMERICAL ILLUSTRATION

In the example a software system is decomposed into three modules, \(m_1\), \(m_2\) and \(m_3\). Ten software components (\(sc_1\)–\(sc_{10}\)) that are available in markets make up four sets of alternative software component (\(S_1\)–\(S_4\)) for each module, four functional requirements of the system are identified namely \(R'1\), \(R'2\), \(R'3\), \(R'4\).

<table>
<thead>
<tr>
<th>Functional Requirements</th>
<th>Sets of alternative components</th>
<th>Software Components</th>
<th>Module m1</th>
<th>Module m2</th>
<th>Module m3</th>
</tr>
</thead>
<tbody>
<tr>
<td>R'1</td>
<td>(S_1)</td>
<td>(Sc_1)</td>
<td>1</td>
<td>0.68</td>
<td>0.51</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Sc_2)</td>
<td>1</td>
<td>0.22</td>
<td>0.63</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Sc_3)</td>
<td>0.97</td>
<td>0.77</td>
<td>0.19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Sc_4)</td>
<td>1</td>
<td>0.23</td>
<td>0.57</td>
</tr>
<tr>
<td>R'2</td>
<td>(S_2)</td>
<td>(Sc_5)</td>
<td>1</td>
<td>0.22</td>
<td>0.45</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Sc_6)</td>
<td>1</td>
<td>0.45</td>
<td>0.94</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Sc_7)</td>
<td>0.99</td>
<td>0.25</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Sc_8)</td>
<td>0.95</td>
<td>0.34</td>
<td>0.99</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Sc_9)</td>
<td>1</td>
<td>0.57</td>
<td>0.45</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cost associated with components in ()</th>
<th>Module m1</th>
<th>Module m2</th>
<th>Module m3</th>
</tr>
</thead>
<tbody>
<tr>
<td>(c_{11})</td>
<td>1099</td>
<td>1068</td>
<td>1010</td>
</tr>
<tr>
<td>(c_{21})</td>
<td>1060</td>
<td>1022</td>
<td>1063</td>
</tr>
<tr>
<td>(c_{31})</td>
<td>1000</td>
<td>1015</td>
<td>1019</td>
</tr>
<tr>
<td>(c_{41})</td>
<td>1005</td>
<td>1023</td>
<td>1020</td>
</tr>
<tr>
<td>(c_{51})</td>
<td>1055</td>
<td>1094</td>
<td>1093</td>
</tr>
<tr>
<td>(c_{61})</td>
<td>1000</td>
<td>1022</td>
<td>1045</td>
</tr>
<tr>
<td>(c_{71})</td>
<td>1000</td>
<td>1045</td>
<td>1094</td>
</tr>
<tr>
<td>(c_{81})</td>
<td>1094</td>
<td>1025</td>
<td>1000</td>
</tr>
<tr>
<td>(c_{91})</td>
<td>1064</td>
<td>1034</td>
<td>1030</td>
</tr>
<tr>
<td>(c_{101})</td>
<td>1000</td>
<td>1057</td>
<td>1045</td>
</tr>
</tbody>
</table>

Table 1 Functional ratings of software components

In Table 1, the function ratings describe the degrees of functional contributions of the software components towards the software modules. The function ratings range from 0 to 1 where 1 refers to a very high degree of contribution while 0 indicates zero degree of contribution. Table 3 shows the degrees of interaction among the software components. The range of the degrees is 1–10. The degree ‘1’ means a very low degree of interaction while the degree ‘10’ refers to a very high degree of interaction.. The detailed data set we are using here are given in Table 1, Table 2, Table 3 & Table 4.

In this example, \(\lambda = 0.5\) is taken as the weight assigned to objective function. Equal weightage is given to both the objectives. Let \(C = 23500\), \(H\) is set as 0 for all the module constraints.
Table 3 Interactions among software components for all three modules $m_1$, $m_2$ & $m_3$

The optimal solution of the problem (P3) is an optimal solution for the problem (P1). The normalized data set for fixed weights approach (Geoffrion, 1968) for the objective function ICD is as follows:

<table>
<thead>
<tr>
<th></th>
<th>$S_1$</th>
<th>$S_2$</th>
<th>$S_3$</th>
<th>$S_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_1$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_2$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_3$</td>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_4$</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_5$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_6$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_7$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_8$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_9$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_{10}$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Table 4 Normalized interactions data set

Solving Problem P1 for the above mentioned data set, we get the solution as follows:

<table>
<thead>
<tr>
<th></th>
<th>$S_1$</th>
<th>$S_2$</th>
<th>$S_3$</th>
<th>$S_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_{c1}$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_{c2}$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_{c3}$</td>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_{c4}$</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_{c5}$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_{c6}$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_{c7}$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_{c8}$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_{c9}$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$S_{c10}$</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

DISCUSSION AND CONCLUSION

In this paper, a methodology of selecting software components for CBSS development is described. A modified way of measuring the cohesion and coupling of software modules along with satisfying budgetary constraint is given. To obtain an optimal/near optimal solution for the selection of software components, Bhatia et. al. approach is used.

REFERENCES