CREATIVE MATH. & INF. **Online version at** http://creative-mathematics.ubm.ro/
17 (2008), No. 3, 505 - 510 **Print Edition: ISSN 1584 - 286X Online Edition: ISSN 1843 - 441X**

Dedicated to Professor Iulian Coroian on the occasion of his 70^{th} *anniversary*

Component selection based on fuzzy clustering analysis

CAMELIA ŞERBAN, ANDREEA VESCAN AND HORIA F. POP

ABSTRACT. Component Based Software Engineering (CBSE) is concerned with the assembly of pre-existing software components. Component selection is a crucial problem in CBSE.

The problem of selecting the best candidate from a set of available components is discussed in this paper. Fuzzy clustering analysis is used to classify the components into different groups based on the values of metrics that measure different attributes of the components. The choice of the best component is based on the obtained classifications.

1. INTRODUCTION

Component-Based Software Engineering is a branch of the software engineering discipline and covers both component development and system development with components. The main objective of CBSE is to obtain a more efficient development with shorter development times and better quality products: using an existing component, rather than developing a new one is often the quickest and cheapest method of meeting any given need.

In this paper we address the problem of automatic component selection. Informally, our problem is to select a subset of components from the available component set which satisfy the system requirements. When we select a component, there may be different components that offer similar functionalities. We aim at a selection approach that takes into consideration some attributes of the components, initially established. Thus, results the necessity to evaluate these components. As a result in this direction, software metrics are very useful being a mean to quantify those attributes considered important for the system that will be built. Fuzzy clustering analysis is used to classify the components based on the values of metrics. The choice of the best component is based on the obtained classifications.

We discuss the proposed approach as follows. Section 2 describes in details our proposed approach for component selection problem based on component evaluation using fuzzy clustering analysis. The experimental results obtained by applying the proposed approach on a component system are discussed in Section 3. Section 4 reviews related works in the area of component selection problem and fuzzy clustering. Finally, Section 7 summarizes the contributions of this work and outlines directions for further research.

Received: 29.10.2008. In revised form: 5.05.2009. Accepted: 19.05.2009.
2000 *Mathematics Subject Classification.* 68Q01, 68Q15, 03E72.
Key words and phrases. *Software metric, component-based development, fuzzy analysis.*

2. Component Selection Problem and Fuzzy Clustering Analysis

Component selection methods are traditionally done in an architecture-centric manner, meaning they aim to answer the question: "Given a description of a component needed in a system, what is the best existing alternative available in the market?". Existing methods include OTSO [15] and BAREMO [7].

Another type of component selection approaches is built around the relationship between requirements and components available for use. In [11] the authors have presented a framework for the construction of optimal component systems based on term rewriting strategies. Paper [8] proposes a comparison between a Greedy algorithm and a Genetic Algorithm. The discussed problem considers a realistic case in which cost of components may be different.

Regarding the problem of software metrics results interpretation, M. Frentiu and H.F. Pop [14] presented an approach based on fuzzy-clustering to study dependence between software attributes, using the projects written by second year students as a requirement in their curriculum. They have observed that there is a strong dependence between almost all considered attributes.

3. Proposed approach

3.1. **Component Selection Problem.** Component Selection Problem (CSP) is the problem of choosing a number of components from a set of components such that their composition satisfies a set of objectives. The notation used for formally defining CS, as laid out in [9] with a few minor changes to improve appearance is described in the following.

Denote by SR the set of final system requirements $SR = \{r_1, r_2, ..., r_n\}$, and by SC the set of components available for selection $SC = \{c_1, c_2, ..., c_m\}$. Each component c_i may satisfy a subset of requirements from SR, $SR_{c_i} = \{r_{i_1}, r_{i_2}, ..., r_{i_k}\}$. The goal is to find a set of components Sol in such a way that every requirement r_j from the set SR may have assigned a component c_i from Sol where r_j is in SR_{c_i}.

Different components may exist to satisfy the same needed requirement and our aim is to select the best available component. The evaluation of the components is a necessity to help us decide which component should be selected.

3.2. **Component evaluation using metrics.** As already mentioned before, the problem of selection of a component from a set of available components is not unique. In general, there may be different alternative components that can be selected, each coming at their own set of offered requirements. Our goal is to select the best existing component, taking into account some attributes considered important for the final system. Considered attributes are quantified using relevant metrics. Based on the interpretation of the measurements results obtained, we decide what component best satisfies the system need.

The main objective of CBSE is that of obtaining a more efficient system with shorter development time and better quality products. This reason argues that cost, reusability and functionality are the most important attributes for the final system. In the following we define metrics for these attributes.

Cost Metric. The cost of a component (C), is defined as the overall cost of acquisition and adaptation of that component. We aim at finding a solution that minimizes the total cost of the system.

Reusability Metrics. Reusability is of great importance to any software product. In what follows we will establish metrics that emphasize some aspects related with reusability of a component. An exhaustive presentation of these metrics is not the purpose of this article, we consider some reusability metrics closely tied to our problem.

In [2] Hoek et al. proposed metrics to assess service utilization in component assemblies. We use in this article two of these metrics: Provided Services Utilization Metrics(PSU), defined as the ratio of services provided by the component which are actually used (Equation 3.1 - left side) and Required Services Utilization (RSU) Metric, witch is similar with PSU metric but for required services (Equation 3.1 - right side).

$$PSU_X = \frac{P_{actual}}{P_{total}} \qquad RSU_X = \frac{R_{actual}}{R_{total}} \tag{3.1}$$

where: P_{actual} = number of services provided by component X that are actually used by other components and P_{total} = number of services provided by component X; R_{actual} = number of services required by component X that are actually provided by the assembly and R_{total} = number of services required by component X.

A value near to 1 for PSU metric indicates a poor reusability, while a value near to 0 indicates that the component could be reusable if the system evolve and adds other requirements. For RSU metric, a value near to 1 is a good reusability indicator.

Functionality Metric. Functionality metric for a component represents the ratio between the number of required services provided by the component and the number of system required services. A component with metric value near to 1 offers several functionalities of the system.

3.3. **The Issue of Threshold Values.** After computing the metrics values, the next step is to give a relevant interpretation of the measurements results obtained. Following a classical approach we have to set thresholds values for metrics that we use. A threshold divides the space of a metric value into regions. Depending on the region a metric value is in, we can make an informed assessment about the measured entity. For example, if we measure the reusability of a component with possible values in the [0..1] range and we define 0.7 as being the threshold with good reusability, then all measured components whose reusability values are above that threshold can be quantified as being reusable. This simple example raises a set of questions: how did we come up with a threshold of 0.7 in the first place? Why not 0.5? And, is a component with a reusability value of 0.68 not reusable compared to a component having a reusability value of 0.7? Would such a threshold still be meaningful in a population where the biggest reusability value is 0,5?

As a conclusion, the accuracy of the results obtained is questionable. In order to overcome this limitation, we propose an alternative approach for the problem

of setting up the software metrics threshold values using fuzzy clustering analysis. This allows us to place an object in more than one group, with different membership degrees.

3.4. **Fuzzy clustering analysis.** Clustering is the division of data set into subsets (clusters) such that, similar objects belong to the same cluster and dissimilar objects to different clusters. Many concepts found in real world do not have a precise membership criteria, thus there is no obvious boundary between clusters. In this case, fuzzy clustering is often better as an object belongs to more that one clusters with different membership degrees.

The fuzzy clustering generic algorithm (Fuzzy n-means algorithm) used to determine the fuzzy partition of components set that we analyze is described in [3]. This algorithm has the drawback that the optimal number of classes corresponding to the cluster substructure of the data set, is a data entry. As a result in this direction, hierarchical clustering algorithms produce not only the optimal number of classes (based on the needed granularity), but also a binary hierarchy that show the existing relationships between the classes. Thus, a second partition of the components set is realize using Fuzzy Divisive Hierarchic Clustering (FDHC) algorithm [6].

4. Case Study

In order to validate our approach we have used the following case study. The set of requirements $SR = \{r_0, r_3, r_4, r_7, r_9, r_{12}\}$ and the set of components $SC = \{c_0, c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8, c_9\}$ are given. Table 1 contains for each component the provided services (in term of requirements for the final system) and the cost. Table 2 contains the values of the metrics computed based on the set of requirements and the set of component requirements dependencies. The dependencies are not specified in this paper due to space limitation.

The considered case study is not a real one, our goal is to emphasize the relevance of our approach regarding fuzzy clustering analysis in component selection problem. This classification is based on the selected metrics values. In this repository there are components that offer similar functionalities with different cost and reusability. Our approach partition the initial set of components such that similar components belong to the same partition with a membership degree. Regarding the priority of the attributes considered important for the final system, we select those components that fit out purpose.

Applying the *FDHC* algorithm [6] we obtained the results in Figure 1.

We have obtained three classes, therefore we will use three clusters to further classify the available components using the values of the above stated metrics. We have run the fuzzy algorithm for three clusters. The first case taking into consideration only the PSU / RSU metrics and the second case with the Functionality and Cost metrics values. The obtained results are listed in Figure 2.

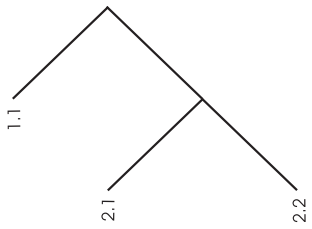
Choosing appropriate components. Based on the *FDHC* algorithm [6] we can deduce that choosing component from the class 1.1 is prioritar then choosing from classes 2.1 and 2.2. Therefore we choose the components c_2 and c_6 that offer the requirements: $\{r_0,\ r_7,\ r_9,\ r_{12}\}$. The remaind set of requirements is: $\{r_3, r_4\}$. The components that offer those requirements are from the class 2.1 and 2.2, but

Comp.	Requirements
c_0	$\{r_0, r_1, r_7\}$
c_1	$\{r_4, r_5, r_6, r_{12}\}$
c_2	$\{r_0\}$
c_3	$\{r_0, r_2, r_8, r_{10}\}$
c_4	$\{r_3, r_{11}\}$
c_5	$\{r_4, r_5, r_6, r_9\}$
c_6	$\{r_7, r_9, r_{12}\}$
c_7	$\{r_1, r_2, r_9, r_{12}\}$
c_8	$\{r_3, r_4, r_{10}, r_{11}\}$
c_9	$\{r_0, r_5, r_6, r_8, r_9, r_{12}\}$

TABLE 1. Requirements

Comp.	PSU	RSU	F	C
c_0	0.66	0.50	0.33	0.08
c_1	0.50	0.60	0.33	0.07
c_2	1.00	1.00	0.16	0.06
c_3	0.25	0.20	0.16	0.09
c_4	0.50	0.00	0.16	0.06
c_5	0.50	0.60	0.33	0.14
c_6	1.00	1.00	0.50	0.15
c_7	0.50	0.33	0.33	0.14
c_8	0.50	0.50	0.33	0.07
c_9	0.50	0.50	0.50	0.14

TABLE 2. Metrics



Class	Members
1.1	$c_2,\ c_6$
2.1	$c_0,\ c_1,\ c_5,\ c_7,\ c_8,\ c_9$
2.2	$c_3,\ c_4$

FIGURE 1. Classification tree and final partition for the set of 10 components

comp0	0.33	0.08	0.009534	0.979602	0.010863	C2
comp1	0.33	0.07	0.021609	0.952634	0.025757	C2
comp2	0.16	0.06	0.000804	0.003243	0.995954	C3
comp3	0.16	0.09	0.00335	0.013673	0.982977	C3
comp4	0.16	0.06	0.000804	0.003243	0.995954	C3
comp5	0.33	0.14	0.055393	0.897396	0.047211	C2
comp6	0.5	0.15	0.998969	0.000819	0.000213	C1
comp7	0.33	0.14	0.055393	0.897396	0.047211	C2
comp8	0.33	0.07	0.021609	0.952634	0.025757	C2
comp9	0.5	0.14	0.998985	0.000809	0.000206	C1

Functionality and Cost

comp0	0.66	0.5	0.04595	0.877104	0.076947	C2
comp1	0.5	0.6	0.014881	0.960241	0.024878	C2
comp2	1	1	0.999992	6.00E-06	2.00E-06	C1
comp3	0.25	0.2	0.019484	0.129603	0.850913	C3
comp4	0.5	0	0.018127	0.082048	0.899825	C3
comp5	0.5	0.6	0.014881	0.960241	0.024878	C2
comp6	1	1	0.999992	6.00E-06	2.00E-06	C1
comp7	0.5	0.33	0.031732	0.576785	0.391483	C2
comp8	0.5	0.5	0.002324	0.990335	0.007341	C2
comp9	0.5	0.5	0.002324	0.990335	0.007341	C2

PSU and RSU

FIGURE 2. Fuzzy clustering with three clusters considering different metrics

the best available option is component c_8 which offers both functionalities and has the best cost.

5. CONCLUSIONS AND FUTUREWORK

Component Selection Problem has been investigated in this paper. A challenge is how to select those components that best satisfy the system need. In this direction, we have proposed an new approach regarding component evaluation based on fuzzy clustering analysis. In order to validate our approach we have used a

case study, presented in Section 4. We will focus our future work on three main fronts:

(1) To develop an algorithm for automatic component selection based on the obtained evaluation;
(2) To apply this approach for more case studies;
(3) Comparison with others approaches regarding the issue of thresholds values.

6. Acknowledgement

This material is based upon work supported by the Romanian National University Research Council under award PN-II no. ID_550 / 2007.

References

[1] Fox, M. R., Brogan, D. C. G. and Reynolds,P. F., Approximating component selection, in *Proc. 36th conference on Winter simulation*, Washington, 2004, pp. 429-434
[2] Hoek, A. v. d., Dincel, E., and Medvidovic, N., *Using Service Utilization Metrics to Assess and Improve Product Line Architectures*, 9th IEEE International Software Metrics Symposium (Metrics'2003), Sydney, Australia, 2003
[3] Bezdek, J., *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York, 1981
[4] George T. Heineman and William T. Councill, *Component-based software engineering: putting the pieces together*, Addison-Wesley Longman Publishing Co., 2001
[5] Zadeh, L.A., Fuzzy sets, *Information and Control* 8, 1965, pp.338-353
[6] Dumitrescu, D., *Hierarchical pattern classification*, Fuzzy Sets and Systems 28, 1988, pp.145-162
[7] Lozano-Tello, A. and Gomez-Perez, A., BAREMO: how to choose the appropriate software component using the analytic hierarchy process, in *The 14th international conference on Software engineering and knowledge engineering*, ACM , New York 2002, pp. 781-788
[8] Haghpanah, N., Moaven, S., Habibi, J., Kargar, M. and Yeganeh,S. H., Approximation Algorithms for Software Component Selection Problem, in *The 14th Asia-Pacific Software Engineering Conference*, IEEE Press, 2007, pp. 159-166
[9] Fox, M. R., Brogan D. C. G. and Reynolds, P. F., Approximating component selection, in *Proc. 36th conference on Winter simulation*, Washington, 2004 pp. 429-434
[10] Baker, P., Harman, M., Steinhofel, K. and Skaliotis, A., Search Based Approaches to Component Selection and Prioritization for the Next Release Problem, in *The 22nd IEEE International Conference on Software Maintenance*, IEEE Press, Washington, 2006, pp. 176-185
[11] Gesellensetter, L. and Glesner, S., *Only the Best Can Make It: Optimal Component Selection*, Electron. Notes Theor. Comput. Sci, 176, 2007, pp. 105-124
[12] Alves, C. and Castro, J., Cre: A systematic method for cots component selection, in *Brazilian Symposium on Software Engineering*, IEEE Press, Rio De Janeiro, 2001
[13] Alves, C. and Castro, J., Pore: Procurement-oriented requirements engineering method for the component based systems engineering development paradigm, in *Int. Conf. Software Eng. CBSE Workshop*, IEEE Press, 1999
[14] Frentiu, M. and Pop, H.F., *A study of dependence of software attributes using data analisys techniques*, Studia Univ. Babes-Bolyai, Series Informatica, 2, 2002 pp. 53–66
[15] Kontio, J., *OTSO: A Systematic Process for Reusable Software Component Selection*, Technical report, University of Maryland, 1995

Babeş-Bolyai University
Department of Computer Science
Kogalniceanu 1
400084 Cluj-Napoca, Romania
E-mail address: camelia@cs.ubbcluj.ro
E-mail address: avescan@cs.ubbcluj.ro
E-mail address: hfpop cs.ubbcluj.ro