Advances in Engineering Research (AER), volume 174
International Conference on Energy Development and Environmental Protection (EDEP 2018)

Business Component Identification Method Based on Semantic Similarity and the Cluster Algorithm

Li-Yan CHEN[1,a], Long TAN[2,b] and Jun LU[3,c]

[1, 2, 3] School of Computer Science and Technology, Heilongjiang University, Harbin, China

[a]Email:cly1970@126.com

Keywords: Reusable business component, component identification, semantic similarity, cluster algorithm.

Abstract. Component identification is a key problem in software reuse. In order to obtain a set of business components (BCs) with high reuse value and good reuse performance to support reuse, a BC design method based on the cluster algorithm was proposed. Through analyzing existing business models, element composite models were described to divide the domain by analyzing the conception semantics of the transaction field. The hierarchical clustering analysis technique based on the similarity degree among activities was also given. In the identification process, the concept of business element similarity which can overcome the limitation of the domain platform was given. Commonality, variability, granularity, and reuse cost were taken into account in the method. Experiment results show that the valuation and performance of reusability for the transaction component are improved effectively, especially the design in the platform independent model.

Introduction

Component-based software development has become the key technology in the development of information systems and software reuse. The identification and design of a reusable component are the prerequisites of component-based software development. The current paper proposes a business component (BC) identification model that analyzes the semantic similarity between requirement models through the hierarchical clustering analysis method in order to identify transaction components. Activities with higher similarity are grouped into a cluster and are identified as BCs.

Component identification is a major task in domain engineering. Recently, the problem of component identification has gained widespread attention considering the cost of component reuse. The general BC under the same domain can be divided into two categories: structural decomposition and feature matching. Structural decomposition starts with the structure of components and then proceeds to the function components using the principles of high cohesion and low coupling. The current representative methods which are comprehensively reviewed in the literature [1] are O2BC [2], matrix analysis method [3, 4], and cluster analysis method [5]. However, these methods cannot analyze components with variable features. They also have limitations in practical applications.

Cluster analysis collects elements with a high degree of similarity or correlation to form a pattern [6, 7, 8]. The current paper bases on the background of the same domain. Through domain analysis of the different business requests, a domain business model is given, and then the elements composing a business activity (BA) and the relationships between them are described. Based on semantics including the relations and similarities between concepts, the definitions of function specifications and similar operation relations are given, and the method of calculating the similarity among a number of activities is examined. Based on the similarities, the technique of cluster analysis is used to classify the activities in the domain business model. Those activities with higher similarity are grouped into a cluster and are identified as the function components. A formula is used in BC size to test the complexity of the BCs.

The rest of the paper is organized as follows. Section 2 presents the semantics used including the relations and similarities between concepts and gives the function specification and similar operation relations. Using the similarities identified, the technique of cluster analysis is used in Section 3 to categorize the activities according to similar operations. Those activities with higher similarity are

group into a cluster and are identified as the function components. Variable granularity and reuse cost are considered in the method.

Approaches to Systems Component

In this section we will discuss some approaches to systems component which are presented in the literature. Most of the papers on this subject are experience reports in which a specific system in a specific environment has been worked upon. Often tools have been built or customized to support the experiments but no general completely automated process has been described.

The actual clustering takes place using a hierarchical agglomerative algorithm [9]. The authors have been experimenting with several group similarity measures and report that the single linkage update rule has proved most promising. The tool supports batch clustering, interactive radical clustering where the user is asked for confirmation each time two clusters are combined, and interactive reclustering which uses a previous classification to guide the clustering. Work has been done on automatic tuning of the similarity measure and learning capabilities by use of neural networks (see also).

Also group related procedures together. They use data bindings to determine how two procedures are related. Four types of data bindings are distinguished. The resulting matrix serves as input for a hierarchical clustering algorithm.

Reusable BC Design

A. Concept Semantics

A BC is a common component combined and abstracted from a few similar activities in a common domain. The common component has variable features. A cluster of specific components may be obtained by identifying the variable points.

Definition 1: A BC can be formally defined as BC=(Id,OP,R,C), where Id is the identity of the BC, OP is the set of operations that the BC contains, R is the set of relationships between business elements, and C is the set of business elements.

Definition 2: A BA can be defined as BA={Id,n,OP,R}, where Id is the identity of the BA, n is the name of the BA, OP is the set of operations the activity contains, and $R \subseteq OP \times OP$ is the set of relationships between operations.

Definition 3: The development cost of CD(C) is the cost of achieving C, which is based on the development costs of the operations involved in component C, that is,

$$CD(C) = \sum_{op \in OP(C)} CD(op) \tag{1}$$

where CD(op) is the cost of achieving the operation of op; the operation costs can be determined according to the type and complexity of the operations involved.

B. Similarity among Semantic Concepts

In domain terminology, business object elements have abstract descriptions according to their concepts and relationships, which are in turn organized by category, with each type of organization forming a hierarchical concept tree. Each node in the concept tree is a set of equivalence relations with a set of concepts.

Definition 4: Semantic distance expresses the distance between two ontologies in the semantic distance, o_1 and o_2. They are recorded as $dist(o_1, o_2)$, and their values satisfy Equation (1) requirements:

$$dist(o_1, o_2) = \begin{cases} (2^{-1-level(comF)} - 2^{-1-level(o_1)}) + (2^{-1-level(comF)} - 2^{-1-level(o_2)}) & o1 \text{ and } o2 \text{ are different concepts} \\ 0 & o1 \text{ and } o2 \text{ are the same concepts} \end{cases} \tag{2}$$

where comF is the most recent public parent from o_1 and o_2 in the concept hierarchy tree; and $level(comF)$, $level(o_1)$, and $level(o_2)$ represent nodes comF, o_1, and o_2 which are the depth of the conceptual level tree. The top-level nodes are made $level(Top)=0$, increasing the top layer below 1.

Based on the above definition, the semantic distance is normalized to satisfy the semantic distance between two semantic concepts with the farthest distance approaching infinity but no more than 1. The semantic distance determines the similarity between two concepts; the greater the semantic distance between the concepts, the smaller their similarity. Based on $dist(o_1, o_2)$, the similarity between two concepts is defined as follows:

$$Sim(o_1, o_2) = 1 - dist(o_1, o_2) \tag{3}$$

where $Sim(o_1, o_2) \in (0,1]$. The calculation of the concept of semantic similarity is based on the amount of information sharing and the graph (tree) method. The tree-based method is easy to implement and is effective, so it was used to calculate the similarity between concepts with the concept tree. To do this, there is a need to determine first the location of the tree in the hierarchy, followed by the corresponding ontology editor (such as a protégé) in order to abstract a concept hierarchy tree into a computer which in turn processes the ontology tree (mainly on the concept of parent–child class relations).

C. Division of the Operational Set

Based on the above definition, the similarity of operations can be determined by how the relation of semantic similarity of different concepts achieves the abstract semantic matching services in micro services. This includes the parameters between the two operations, both input and output parameters, and the semantic matching data sets. The final two operations involve matching the semantics of these elements and determining the weight of each.

Definition 5: Operation similarity: suppose opi= $(Id_i, n_i, t_i, In_i, Out_i, D_i)$ and opj = $(Id_j, n_j, t_j, In_j, Out_j, D_j)$ are two operations in DM. The similarity between opi and opj is then defined as

$$\begin{cases} w_n \cdot Sim(n_i, n_j) + w_{In} \cdot Sim(In_i, In_j) + w_{out} \cdot Sim(Out_i, out_j) + w_D \cdot Sim(D_i, D_j) & t_i = t_i \\ 0 & t_i \neq t_i \end{cases} \tag{4}$$

where $Sim(n_i, n_j)$ is the semantic similarity between the names of operation, $Sim(In_i, In_j)$ is the semantic similarity in the parameter name In_i and In_j, $Sim(Out_i, Out_j)$ is the semantic similarity in the parameter name Out_i and out_j, $Sim(D_i, D_j)$ is the semantic similarity between data sets D_i and D_j, $w_n, w_{In}, w_{out}, w_D$ are the weights, $w_n, w_{In}, w_{out}, w_D \in [0,1]$, and $w_n + w_{In} + w_{out} + w_D = 1$.

Using cluster analysis below, we select the operation of the similarity threshold θ given the similarity between operations.

Definition 6: Suppose op_i and op_j are two DM operations; if they can achieve a similar function, and their op_i and op_j are similar, they are recorded as $op_i \sim op_j$.

Definition 7: Suppose OP is a DM operation set, $op \in OP$ is an operation in OP, an OP having similar operations with op is referred to as a similar set of operations generated by op; thus, it is recorded as [op]~. That is, $[op]\sim = \{op' | (op' \in OP) \wedge (op' \sim op)\}$.

Definition 8: Suppose OP is a DM operation set. Then all different sets of similar operations on OP called the similarity division ~ are recorded as OP/~. That is, $OP/\sim = \{[OP]\sim | op \in OP\}$

According to the similarity among operations in the construct of the relations of operation similarity, the structural rules are as follows:

Rule 1. If $Sim(op_i, op_j) < \theta$ (θ is the operation similarity threshold), then $op_i \sim op_j$.

Rule 2. Suppose OP_i and OP_j are a set of similar operations; if they exist as $op_i \in OP_i$, $op_j \in OP_j$, then they meet the conditions $op_i \sim op_j$, then $OP_i \sim OP_j$.

Based on the above definition, the similarity of operations can be determined by the relation of semantic similarity of different concepts. The cluster analysis method is used to select the operation similarity, threshold θ, and to analyze the relations of operation similarity.

D. Algorithm of the Division of Similar Operation Sets

The algorithm of the division of a similar operation set is as follows:

Input: operation set OP
Output: OP/~={[op]~|op∈OP}
Algorithm description

```
TF=OP; OP/~=∅
For (opk∈TF){
  Add([opk]~,opk);  //add opk to [opk]~
  Remove (TF,opk);
/*remove opk from the collection TF*/
  for (opj∈TF){
    for (opt∈([opk]~)
      if (Sim(opj,opt)< θ){
 /*if the similarity of opj and opt is less than θ*/
          Add ([opk]~,opj);
/*add element opj to the set [opk]*/
          Remove (TF,opj);
/*remove opj  from the collection TF */
          Continue;
      }
   }
   Add(OP/~,[opk]~);   //add OP/~ to [opk]
  }
```

BC Identity

The reuse of BCs aims not only to meet the availability of the field but also to achieve high cohesion, low coupling, and easy reuse. In the current paper, the similarity of activity elements is calculated according to the given operations. Identifying a cluster of BCs depends on the similarity of the activity elements in order to obtain an activity set cluster as a reusable BC.

A. Feature Matrix

The similarity degree expresses the similarity degree of activity elements, which can be calculated according to the feature matrix. This matrix presents the similarity feature matrix of BAs. In the present study, we take the BA matrix as an example.

Definition 7(Active similarity): If A is an active set, OP (A) is the operation aggregate which contains all activities in A; an active-similar operation matrix may then be defined as MA-OP=[A,OP(A)/~]. There is an operation belonging to [opj]~([opj]~∈OP(A)/ ~), where there is a line in ai, and [opj] ~ the column in which there is disposal to 1; otherwise, it is marked as 0.

OP is in the domain model operation set, and all different similar operation sets of OP are recorded as OP~.

The reference Sorenson Coefficient provides the following formula to calculate the similarity between many activities:

$$Sim(A) = p / (p + \sum_{i=1}^{q} r(b_i)) \quad (5)$$

where p expresses that the value of all elements in each column number is 1, q expresses that each column contains the number 0, r(bi) expresses the ith column, and the proportion is 0.

If the number of active A is 2, then there is Sim(A)=2p/(2p+q), the formula for the Sorenson Coefficient, which shows that this coefficient is a special case of the formula.

For example, Table 1 shows the matrix for an active-similar operation. It shows a1, a2, and a3, two kinds of quantity which are p=2 and q=3, Sim (a1,a2,a3)=2/(2+(1/3+2/3+2/3))=6/11, a3 and a4, two kinds of matched relation quantities which are p=2, q=2, respectively, and Sim (a3,a4)=2/(2+(1/2+1/2))=2/3.

Table 1 Matrix of active-similar operations

Operation set / Activity set	op(A)/~						
	[op1]/~	[op2]/~	[op3]/~	[op4]/~	[op5]/~	[op6]/~	[op7]/~
ba1	1	0	1	1	1	0	1
ba2	0	0	1	0	1	0	1
ba3	1	1	0	0	1	1	0
ba4	0	1	0	0	0	1	0
ba5	1	1	1	0	0	1	0

The similarity degree between business activities can be expressed using an operating diagram between business activities. As shown in Figure 1, the figure can be defined as an undirected weighted graph G={V,E}, where V={v1,v2,…vn} is the set of vertices, with each vertex representing a business operation, E={(vk,vj,w)| vk,vj∈V, (vk≠vj), w=RD(vk,vj), (w>0)} is the set of edges, with each edge representing a BA, and w=RD(vk,vj) for vertex vk, and vj represents the similarity degree between BAs. Table 1 can be expressed into Figure 1.

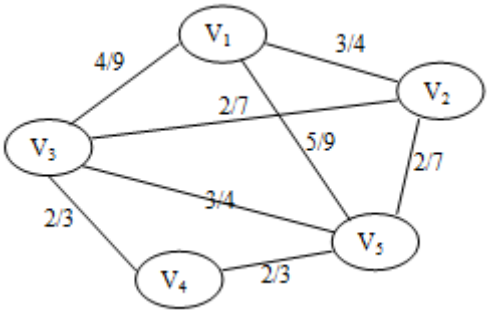

Fig. 1 Relationship among BAs

B. BC Identify Algorithm

Ensuring the activities of every concept component is a key challenge in component identification separate from the orient component analysis of the domain model. To improve the reuse degree of components and to decrease the development cost of the software system, the activities of the domain model are grouped into an active cluster. This cluster is then optimized and transformed to the component concept.

The present study presents an activity-based component identification method of similarity degree. The idea of the method is to have the activities of each active cluster separated from DM. Each activity cluster is used in the hierarchical clustering analysis, and those with the greatest similarities to each selection and go beyond the specified limit are merged. This is done until all of the clusters are less than the specified lower limit or do not meet two clusters above the conditions.

Algorithm description:

Input: DM=(A,R), activity similarity threshold θ, lower limit of active clustering S, cluster activities included in the maximum number of similar sets of operations N.

Output: DM is divided as P={DM1,DM2,…DMn},DMK=(Ak,Rk) is the activity cluster of DM. P meets the following three conditions:

(1) $\sum_{k=1}^{n} A_K = A$

(2) $A_i \cap A_{j=\varnothing}$ （$1 \le i, j \le n, i \ne j$）

(3) $R_K = \{(a_i, a_j) \mid a_i \in A_K, a_j \in A\}$

$p = \varnothing$

```
for (ak∈A){
  construct the initial cluster DMK=(Ak,Rk), where Ak={ak};
  add(P,DMk);
}
while (|P|≥S){
  List=GetMaxSubDM(P,θ,N);
  if (SizeOf(List)==0) break;
  else
    MergerSubDM(P,List);
}
```

where (P,DMk) will be added to DMk in partition P.

GetMaxSubDM(P,θ,N): set the similarity from P as the largest, which is even bigger than the assigned upper limit θ. The operation set's quantity is smaller than the assigned upper limit N.

Table 1 shows a similar activity matrix, and Figure 1 is the corresponding graph G of the operational elements. The identification result of reusable BCs will be given using Table 1 and Figure 1 as shown in the example. In this case, based on the hierarchical clustering algorithm, we set the similarity threshold θ=0.5, the number of activities cluster limit S=1, and the maximum number of operation N=4. Figure 2 shows the partition graph of activities.

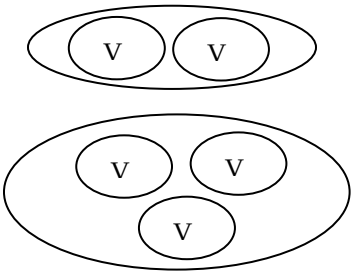

Fig. 2 Recognition results of the reusable BCs

C. Granularity Calculation of BCs

To ensure that the identified BCs are not too complicated, computing the granularity of the BCs by quantitative calculation is necessary. The formula to calculate BC granularity using the feature matrix is based on the circle complexity measure method proposed by McCabe.

$$G(BA) = \sum_{i=1}^{k} (w_i \times sum(I_i)) \tag{6}$$

where k is the number of feature kinds contained by the business activities, sum (Ii) is the number of the ith kind feature, and wi is the complexity weight of the ith kind feature.

Normally, the number of BCs in a medium-size enterprise information system is between 30 and 100. When the BC granularity exceeds the reasonable range, the value of θ can be adjusted to obtain a moderate granularity.

Conclusions

The current paper proposes a hierarchical clustering technique considering the reuse degree of the domain model. The business model is categorized and packaged into components using the principles of high cohesion and low coupling. With the appropriate granularity, the component model addresses

the problem of lack of behavioral semantics and improves the reuse rate and reuse efficiency in component identification. The business model is an attempt to formalize the design and fabrication process of component development. It can be improved further in a number of directions. First, the derived components should represent the optimal design according to the proposed model. Second, the assumption might be relaxed to include a set of legacy systems and existing components in the domain. In fact, Vitharana and Jain[10] have proposed a methodology for component fabrication, but this has not been applied to large-scale object models. Future studies can be automatically classified when the business model components are complete.

Acknowledgement

Supported by Scientific Research Fund of Heilongjiang Provincial Education Department (NO: 11553069)

Supported by the project of National Natural Science Fund under Grant No.81273619 and No.81373537

References

[1] RAINER K. Atomic architectural component recovery for program understanding and evolution[D]. Stuttgart, Germany: University Stuttgart, 2000

[2] Ganesan R., Sengupta S. “O2BC: A technique for the design of component-based applications”, In Proceedings of the 39th International Conference and Exhibition on Technology of Object-Oriented Language and Systems, Santa Barbara, California, pp46-55, 2001

[3] Luo Jing, Zhao Wei, “A Decomposition Method for Object-Oriented Systems Based on Iterative Analysis of the Directed Weighted Graph”, Journal of Software, vol. 15, no. 9, pp1292~1300, 2004

[4] Lee S. D., Yang Y. J. “COMO: A UML-based component development methodology”, in: Proceedings of the 6th Asia Pacific Software Engineering Conference, Takamatsu, pp54-63, 1998

[5] Tang Zhaohui, Liu Heng, Chen Qing etc.The Key Technology Research of Feature-Based Component-Based Software Design.proceedings of the 32nd Chinese Control Conference, pp6464-6468, 2013

[6] Chung-Horng Lung, Marzia Zaman, Amit Nandi. “Applications of clustering techniques to software partitioning, recovery and restructuring”, Journal of Systems and Software, vol. 73, no. 2, pp227-244, 2004

[7] Wang Zhongjie, Zhan Dechen, Xu Xiaofei. “Component Granularity Optimization Design Based on Business Model Stability Evaluation”, Chinese Journal of Computer, vol. 29, no. 2, pp.239-248, 2006

[8] Peng Xin, Zhao Wenyun Liu Yiming, “Feature Model and Component Semantics Based Conceptual Architecture Design”, Journal of Software, vol. 17, no. 6, pp1307-1317, 2006

[9] Y.Li, Z. Bandar. “An Approach for Measuring Semantic Similarity between Words Using Multiple Information Sources”, IEEE Transactions on Knowledge and Data Engineering, vol. 15, no. 4, pp871-882, 2003

[10] Padmal Vitharana, Hemant Jain, Fatemeh “Mariam” Zahedi, “Strategy-Based Design of Reusable Business Components”, IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews, vol. 34, no. 4, 2004