IDENTIFYING BUSINESS COMPONENTS FROM BUSINESS MODEL: A METHOD BASED ON FEATURE MATHCING

Meng Fanchao, Zhan Dechen, Xu Xiaofei

Abstract: Identifying reusable business components from business model is the premise of Component-Based Software Development (CBSD). In CBSD, business component is the basic unit for reuse and it provides a coarse-grained functionality. A business component typically consists of related elements that possess similar features. This paper proposes an approach to business components identification based on features matching. In our method, the concepts of feature and equivalent feature relation are presented, and the rule of judging equivalent feature relation and the algorithm of parting feature set are given. To identify reusable business components, a hierarchical clustering technique is proposed. In the process of clustering, we give the formula of calculating similarity among a set of elements that extends Sorenson Coefficient. Finally, a tool RBCET is built using this method to help reusable business components extraction from domain business model.

Keywords: domain business model; feature; equivalent feature set; reusable business component.

ACM Classification Keywords: D.2 Software – Software Engineering (K.6.3)

Introduction

With the rapid development of hardware of computer, software has become more and more complex. How to rapidly develop maintainable, extensible and adaptable software that meet the changeable requirements has become a crucial problem. Component-Based Software Development (CBSD) plays an important role in tackling software crisis and promoting the software productivity [D'Souza, 1999] [Jain H, 2001]. In CBSD, component is the basic unit for reuse and it provides a coarse-grained functionality. Identification of reusable components is the premise of CBSD [yang Fuqing, 1999], currently there are some methods have been brought forward for resolving this problem. In general, we classify those methods into two categories: structure analysis and feature matching.

The methods of identifying reusable business components from domain business model can be classified into two categories: structure analysis methods and feature matching methods. Structure analysis methods abstract domain business model as mathematical notations, such as tree structure or a graph structure. Via cluster analysis, a domain business model can be partitioned into sub-structures, and each sub-structure is taken as a candidate component. Currently the main methods of structure analysis have COMO [Lee SD, 1998], O2BC [Ganesan R, 2001], CRWD[Somjit Arch-int, 2003] and graph decomposition [Y. Chiricota, 2003] . A disadvantage of these methods is that the results of partition excessively depend on weights which are set by designer so that it is difficult to apply into practice. Feature matching methods classify similar elements according to the features of them, and these methods also depend on the similarity measures and clustering algorithms being used. Wigglers [T.A. Wiggerts, 1997] gives an overview of software clustering techniques and suggests the use of the term 'entity' to describe elements being grouped together and 'feature' to denote the attributes of these elements. The representative feature matching method is the F^3 reuse methodology proposed by AIPA [Silvana Castano, 1997,1998]. In this methodology, reusable conceptual components are constructed from schema families stored in the Design Library using descriptors. They calculate the conceptual distance between components in different schemas and cluster them according to similarity levels based on the computation of an affinity measure between components.

In this paper, a method of reusable component identification based on feature matching is proposed. In our approach, the concepts of feature and equivalent feature relation are presented, and give formula of calculating resemble degree among business elements which extends Sorenson Coefficient. As our experiments show, the method proposed can provide more promising results for component identification than the previously used methods.

Domain Business Model and Business Components

A domain buisness model is composed of a set of buisness models that d belong to same application domain. A business model consists of a buisness object model and a buisness process model.

A business object model is composed of a set of business objects and relationships between them. A business object is an object with well defined boundaries and an identity that encapsulates a business state and behavior [13]. A business state is a structure property represented by attributes while a behavior is a behavioral property represented by business operations that operate on the attributes. Business objects represent resources in a business model such as product, planning, order, material etc.

Definition 1: A business object can be defined as $BO=\{n, A, OP, R\}$, where n is name of business object, A is the set of attributes, and OP is the set of business operations that operate on attributes. R is the set of relationships between the business objects and other business objects.

Attribute can be classified into individual attribute and composite attribute. A individual attribute is defined with a name and a data type, $a=\{n, DT\}$, where n represents the name of attribute, DT represents the data type of attribute. A composite attribute is defined groups of individual attributes logically related and grouped, denote as $a=(a_1,a_2,\ldots,a_n)$, where $a_i(i=1,2,\ldots,n)$ is a individual or composite attribute.

Business object can provide business activities with business operations to satisfy their executive demands. A business operation is defined as $op=\{n , t, In, Out\}$, where n is the name of business operation. $t\in\{Create, Modify, Delete, Transform, Query\}$ is the type of business operation. In is the set of input parameters and Out is the set of output parameters. A parameter can be represented as $p=(n_p, DT_p)$, where n_p is the name of parameter, and DT_p is the data type of parameter.

A relationship between two business objects BO_1 and BO_2 can be defined as $r_{12}=(n_1, n_2, t)$, where, n_1 is the name of BO_1, n_2 is the name of BO_2, $t\in\{Generalization, Association, Dependency\}$ is the type of relationship. A generalization is a taxonomic relationship between a general business object BO_1 and a specific business object BO_2. An association specifies a link relationship that can occur between BO_1's instances and BO_2's instances. A dependency is relationship that signifies that one or more business operations in BO_1 call the business operations in BO_2 for their implementation.

A business process model can be decomposed into a set of business processes and each business process can also decomposed into business activities resulting in a two-level hierarchy the business processes model. A business process is a specific ordering of business activities across time and place, with a beginning, an end, and clearly identified inputs and outputs. These ordered business activities affect the states of business objects by creating, consuming and changing their contents. Business activity that involves business objects operating on a business state with business operation is the basic function unit of a particular business process.

Definition 2: A business activity can be defined as: $BA=\{n, In, Out, OP\}$, where n is name of business activity. In is the set of input business objects, and Out is the set of output business objects. OP is the set of business operations, and each business operation is provided by corresponding business object.

Traditionally, a software component is defined as a self-contained piece of software with well-defined interface or set of interfaces [14]. A larger-grained component called a business component focuses on a business concept as the software implement of an autonomous business concept or business process [G.Q. Huang, 1999]. Business components vary from traditional software artifacts such as code segment, class and procedure etc. Traditional software artifacts are mostly fine-grained and technical-oriented, business components, on the other hand, are more coarse-grained and provide a high-level business-oriented representation, and they can express future components and the relations of those components.

In term of the functions implemented by business components, they can be classified into: entity components and process component. In general, business objects that possess resemble features in a domain business model are capsulated an entity component. Analogously, business activities that carry out resemble tasks in a domain business model are capsulated a process component.

Equivalent Feature Relations

To identify reusable business components from domain business models, we use the elements represent business objects and business activities in domain business model, and use the features represent the characteristics of business objects and business activities. According to the definitions of business object and business activity, the features of business object include name, attributes, business operations and relationships, and the features of business activity include name, input business objects, output business objects and business operations.

To evaluate similarity between elements in different business models in a domain business model, we need to refer to the domain thesaurus containing semantic information. A thesaurus usually is sets of dictionaries, every one of which contains group of terms that are extracted the names of business elements (business objects, attributes, date type, business operation and business activities, etc) from all business models in a domain business model. Each dictionary in domain thesaurus is structured as a directed graph. Nodes of the graph represent the terms and directed edges between nodes represent the partial relations between terms. The distance between two terms reflects the semantic similarity between them. The longer the distance is, and the less the similarity is. In the following, we give the definitions of similarity relation between features.

- Name Similarity

Let n_1 and n_2 be two business objects' names, if $SIM(n_1,n_2)\geq\theta_{BO}$, where, $\theta_{BO}(0\leq\theta_{BO}\leq1)$ is a similarity threshold of the names of business objects, then n_1 and n_2 are similar, denoted as $n_1\square n_2$.

Let n_1 and n_2 be two business activities' names, if $SIM(n_1,n_2)\geq\theta_{BA}$, where, $\theta_{BA}(0\leq\theta_{BA}\leq1)$ is a similarity threshold of the names of business activities, then n_1 and n_2 are similar, denoted as $n_1\square n_2$.

- Attribute Similarity

Let $a_1=(n_1,DT_1)$ and $a_2=(n_2,DT_2)$ be two individual attributes, if a_1 and a_2 satisfy condition: $SIM(n_1,n_2)\geq\theta_A$, where, θ_A $(0\leq\theta_A\leq1)$ is a similarity threshold of attribute names, then a_1 and a_2 are similar, denoted as $a_1\square a_2$.

Let $f=(a_1,a_2,\ldots,a_n)$ and $g=(b_1,b_2,\ldots,b_m)$ be two composite attributes, if there exists a permute function T such that $T(a_1,a_2,\ldots,a_n)=(a_1',a_2',\ldots,a_n')$ and $T(b_1,b_2,\ldots,b_m)=(b_1',b_2',\ldots,b_m')$. If f and g satisfy condition: $(m=n) \wedge (a_1'\square b_1') \wedge (a_2'\square b_2') \wedge \ldots \wedge (a_n'\square b_m')$, then f and g are similar, denoted as $f\square g$.

- Business Operation Similarity

Let p_1=(n_1,DT_1) and p_2=(n_2,DT_2) be two parameters, if p_1 and p_2 satisfy condition: $SIM(n_1,n_2) \geq \theta_P$, then p_1 and p_2 are similar, denoted as p_1□p_2.Let $P_1=\{p_{11}, p_{12}, \ldots, p_{1m}\}$ and $P_2=\{p_{21}, p_{22}, \ldots, p_{2n}\}$ be two set of parameters, $SIM(P_1,P_2)= \frac{2\cdot|P(p_1 \cap p_2)|}{|P_1| \cap |P_2|}$, where, $P(bc_1 \cap bc_2)$ =$\{(p, p')| p \in P_1, p' \in P_2, p \sim p'\}$.

Let bop_1=(n_1,t_1,In_1,Out_1) and bop_2=(n_2,t_2,In_2,Out_2) be two business operations, if bop_1 and bop_2 satisfy condition: $SIM(n_1,n_2) \geq \theta_{BOP} \wedge (t_1=t_2) \wedge SIM(In_1,In_2) \geq \alpha \wedge SIM(Out_1,Out_2) \geq \beta$($\theta_{BOP}$ is a similarity threshold of the names of business operations, α is a similarity threshold of input parameters, and β is a similarity threshold of output parameters, $0 \leq \beta, \gamma, \theta \leq 1$), then bop_1 and bop_2 are similar, denoted as $bop_1 \sim bop_2$.

- Relationship Similarity

Let r=(n_1, n_2, t) and r'=(n'_1, n'_2, t') be two relationships between business objects, if r and r' satisfy condition: $SIM(n_1,n_2) \geq \theta_{BO} \wedge SIM(n'_1,n'_2) \geq \theta_{BO} \wedge (t= t)$', then r and r' are similar, denoted as r□r'.

Based on the similarity relations between features, we can define the equivalent relations between features. Let F is a set of finite features, f_1 and f_2 be two features on set F,

- If $f_1 \sim f_2$, then f_1 and f_2 have equivalence relation, denoted as $f_1 \cong f_2$.
- Equivalent feature relation is transitive, that is to say, if $f_1 \cong f_2$, $f_2 \cong f_3$, then $f_1 \cong f_3$.

Let F be a feature set, for every feature $f \in F$, equivalence feature set of f is defined as: $[f]_{\cong} = \{ f' | (f' \in F) \wedge (f' \cong f) \}$, and the partition on set F can be defined as: $F/{\cong} = \{ [f]_{\cong} | f \in F \}$.

Let DBM be a domain business model, $BOS=\{BO_1,BO_2,\ldots,BO_m\}$ be the set of business objects in DBM, $BAS=\{BA_1,BA_2,\ldots,BA_n\}$ be the set of business activities in DBM. In the following, we give some symbols:

- $N(BOS)$: the set of names of all business objects in BOS, $N(BOS)/{\cong}=\{ [n]_{\cong} | n \in N(BOS)\}$.
- $A(BOS)$:the set of attributes of all business objects in BOS, $A(BOS)/{\cong}=\{ [a]_{\cong} | a \in A(BOS)\}$.
- OP(BOS):the set of business operations of all business objects in BOS, $OP(BOS)/{\cong}=\{[op]_{\cong}| op \in OP(BOS)\}$.
- $R(BOS)$: the set of relationships between business objects in BOS, $R(BOS)/{\cong}=\{ [r]_{\cong} | r \in R(BOS)\}$.
- $N(BAS)$: the set of names of all business activities in BAS, $N(BAS)/{\cong}=\{ [n]_{\cong} | n \in N(BAS)\}$.
- $IN(BAS)$:the set of input business objects' name of all business activities in BAS, $In(BAS)/{\cong}=\{[n]_{\cong} | n \in IN(BAS)\}$.
- $Out(BAS)$: the set of output business objects' name of all business activities in BAS, $Out(BAS)/{\cong}=\{[n]_{\cong} | n \in Out(BAS)\}$.
- $OP(BAS)$: the set of business operations of business activities in BAS, $OP(BAS)/{\cong}=\{ [op]_{\cong} | op \in OP(BAS)\}$.

To acquire equivalence feature set in domain business model, we give the algorithm of parting feature set which are described as following:

Algorithm 1: partition of equivalence feature set

Input: $F = \{f_1, f_2, \ldots, f_n\}$;

```
Output: $F/\cong =\{[\,f\,]_{\cong} \mid f\in F\}$;
$TF \leftarrow F$; $F/\cong \leftarrow \phi$;
for ( each $f_k \in TF$ )
{
  Add ($[f_k]_{\cong}$, $f_k$ );
  Remove ($TF$, $f_k$);
  for ( each $f_j \in TF$)
  {
    if ($f_j \cong f_k$)
    {
      Add ($[f_k]_{\cong}$, $f_j$);
      Remove ($TF$, $f_j$);
    }
  }
  Add ($F/\cong$, $[f_k]_{\cong}$);
}
```

The functions used in the algorithm are defined as follows:

- Add ($[f_k]_{\cong}$, f_k) add element f_k into set $[f_k]_{\cong}$.
- Remove (TF, f_k) delete element f_k from set $[f_k]_{\cong}$.

According to the algorithm, if we input *N*(*BOS*) (resp. *A*(*BOS*), *OP*(*BOS*),*R*(*BOS*),*N*(*BAS*),*In*(*BAS*), *Out*(*BAS*) and *OP*(*BAS*)), it can output *N*(*BOS*)/≅ (resp. *A*(*BOS*)/≅, *OP*(*BOS*)/≅, *R*(*BOS*)/≅, *N*(*BAS*)/≅, *In*(*BAS*)/≅, *Out*(*BAS*)/≅ and *OP*(*BAS*)/≅).

Similarity among a set of elements

A similarity for a given pair of elements indicates the degree of resemblance between the two elements. The metrics to calculate similarity between two elements have Jaccard coefficient, Sorensen coefficient, Russel and Rao coefficient, simple matching coefficient, Soka and sneath and Yule coefficient, etc. An approach may be well suited to one domain but not to another. For the identification of business components, the Jaccard coefficient and Sorensen coefficient metrics are more appropriate than others [S.Mancoridis, 1999]. In this paper, we extend Sorenson coefficient to calculate similarity among a set of elements. Different to the method followed by Davey and Burd [J.Davey, 2000], we use business objects and business activities as elements, and use equivalence feature sets as the attributes of the elements. In this paper, we use equivalence feature matrix calculate the similarity between a set of elements.

An equivalence feature matrix can be defined as $M=[E, F/\cong]=[m_{ij}]_{m\times n}$, where $E=\{e_1,e_2,\ldots,e_m\}$ is the set of elements, F is the set of features that belong to the elements in E, and $F/\cong=\{[f_1]_{\cong}, [f_2]_{\cong},\ldots, [f_n]_{\cong}\}$ is the set of equivalence feature sets on *F*. If there exists a feature $f \in F(e_i)$ ($1 \le i \le n$), such that $f \in [f_j]_{\cong}$ ($1 \le j \le n$), then $m_{ij}=1$, else $m_{ij}=0$.

Let $E_k=\{e_{k1},e_{k2},\ldots,e_{ks}\} \subseteq E$ be a subset of elements in *E*, $[f_j]_{\cong}=(m_{1j},m_{2j},\ldots,m_{nj})$ be a column of matrix *M*, if $m_{ij}=1$ for every $i(i \in \{k1,k2,\ldots,ks\})$, then $[f_j]_{\cong}$ is called matching type t_1 on $e_{k1},e_{k2},\ldots,e_{ks}$; if there exists $i(i \in \{k1,k2,\ldots,ks\})$ such that $m_{ij}=1$, and there also exists p ($p \neq i$, $i,p \in \{k1,k2,\ldots,ks\}$)such that $m_{pj}=0$, then $[f_j]_{\cong}$ is called as matching type t_2 on $e_{k1},e_{k2},\ldots,e_{ks}$; if $m_{ij}=0$ for every i ($1 \le i \le n$), then $[f_j]_{\cong}$ is called matching type t_3 on $e_{k1},e_{k2},\ldots,e_{ks}$. We denote as $T_1(e_{k1},e_{k2},\ldots,e_{ks})$,$T_2(e_{k1},e_{k2},\ldots,e_{ks})$ and $T_3(e_{k1},e_{k2},\ldots,e_{ks})$ the set of matching type t_1, t_2 and t_3 on $e_{k1},e_{k2},\ldots,e_{ks}$. The similarity among $e_{k1},e_{k2},\ldots,e_{ks}$ can be defined as:

$SIM(e_{k1},e_{k2},\ldots,e_{ks})=|T_1(e_{k1},e_{k2},\ldots,e_{ks})|/(|T_1(e_{k1},e_{k2},\ldots,e_{ks})|+\sum_{m_j \in T_2(e_{k1},e_{k2},\ldots,e_{ks})} r(m_j))$, where, $r(m_j)$ represents the proportion of 0 in column $[f_j]_{\cong}$. If $n=2$, then $SIM(e_1,e_2)=2a/(2a+b)$ ($a=|T_1(e_1,e_2)|$,$b=|T_2(e_1,e_2)|$)which becomes Sorenson coefficient.

Here, we give an example to explain the method of calculating the similarity among a set of elements. Table 1 gives a feature matrix that consists of five elements and seven equivalence feature sets.

- $T_1(e_3,e_4,e_5)=\{[f_3]_{\cong}, [f_1]_{\cong}\}$, $T_2(e_3,e_4,e_5)=\{[f_1]_{\cong},[f_2]_{\cong},[f_4]_{\cong},[f_5]_{\cong}\}$,$T_3(e_3,e_4,e_5)=\{[f_6]_{\cong}\}$,$SIM(e_3,e_4,e_5)=2/(2+ (2/3 + 2/3 + 1/3+2/3))=6/13$.
- $T_1(e_1,e_2,e_3,e_4,e_5)=\phi$, $SIM(e_1,e_2,e_3,e_4,e_5)=0$.

Table 1 Feature Matrix

Elements	Equivalence Feature Sets						
	$[f_1]_{\cong}$	$[f_2]_{\cong}$	$[f_3]_{\cong}$	$[f_4]_{\cong}$	$[f_5]_{\cong}$	$[f_6]_{\cong}$	$[f_7]_{\cong}$
e_1	1	1	0	0	1	1	0
e_2	0	1	0	0	1	0	0
e_3	0	1	1	1	0	0	1
e_4	0	0	1	1	0	0	1
e_5	1	0	1	0	1	0	1

Let *DBM* be a domain business model, *BOS* be the set of business objects in *DBM*, *BAS* be the set of business activities in *DBM*.

The similarity among a subset of business objects $BOS_i=\{BO_{i1},BO_{i2},\ldots,BO_{ik}\} \subseteq BOS$ can be defined as: $SIM(BOS_i)=w_N \cdot SIM_N(BOS_i)+w_A \cdot SIM_A(BOS_i)+w_{OP} \cdot SIM_{OP}(BOS_i)+w_R \cdot SIM_R(BOS_i)$, where, $SIM_N(BOS_i)$ is the name similarity among $BO_{i1},BO_{i2},\ldots,BO_{ik}$, $SIM_A(BOS_i)$ is the attribute similarity among $BO_{i1},BO_{i2},\ldots,BO_{ik}$, $SIM_{OP}(BOS_i)$ is the business operation similarity among $BO_{i1},BO_{i2},\ldots,BO_{ik}$, and $SIM_R(BOS_i)$ is the relationship similarity among $BO_{i1},BO_{i2},\ldots,BO_{ik}$. w_N (resp. w_A, w_{OP} and w_R) is the weight of name(resp. attribute, business operation and relationship), $w_N + w_A + w_{OP} + w_R =1$, $0 \le w_N , w_A , w_{OP} , w_R \le 1$. we can set

The similarity among a subset of business activities $BAS_i=\{BA_{j1},BA_{j2},\ldots,BA_{js}\} \subseteq BAS$ can be defined as: $SIM(BAS_j)= w_N \cdot SIM_N(BAS_j)+w_{In} \cdot SIM_{In}(BAS_j)+w_{Out} \cdot SIM_{Out}(BAS_j)+ w_{OP} \cdot SIM_{OP}(BAS_j)$, where, $SIM_N(BAS_i)$ is the name similarity among $BA_{j1},BA_{j2},\ldots,BA_{js}$, $SIM_{In}(BAS_i)$ is the input similarity among $BA_{j1},BA_{j2},\ldots,BA_{js}$, $SIM_{Out}(BAS_i)$ is the output similarity among $BA_{j1},BA_{j2},\ldots,BA_{js}$, and $SIM_{OP}(BAS_i)$ is the business operation similarity among $BA_{j1},BA_{j2},\ldots,BA_{js}$. w_N (resp. w_{In}, w_{Out} and w_{OP}) is the weight of name(resp. input, output and business operation), $w_N + w_{In} + w_{Out} + w_{OP} =1$, $0 \le w_N , w_{In} , w_{Out} , w_{OP} \le 1$.

Business Component Identification Algorithm

To identifying reusable business components from domain business model, we use the hierarchical algorithm to group these elements whose similarity is more than a threshold given to a business component. Hierarchical algorithms start from the individual elements, gather them into small clusters which are in turn gathered into larger clusters. At each step, the two clusters that are closest to each other are merged and the number of clusters is reduced by one.

Algorithm 2:

Input: $E=\{e_1,e_2,\ldots,e_m\}$;θ is the similarity threshold to choose the high reusable business components; s is the size threshold for the result of clustering.

Output: $P(E)=\{E_1, E_2,\ldots,E_n\}$ that satisfies condition: (1) $E_1 \cup E_2 \cup \ldots \cup E_n=E$ ($n \leq s<m$);(2) $E_i \cap E_j=\phi$ ($1 \leq i,j \leq n$; $i \neq j$).

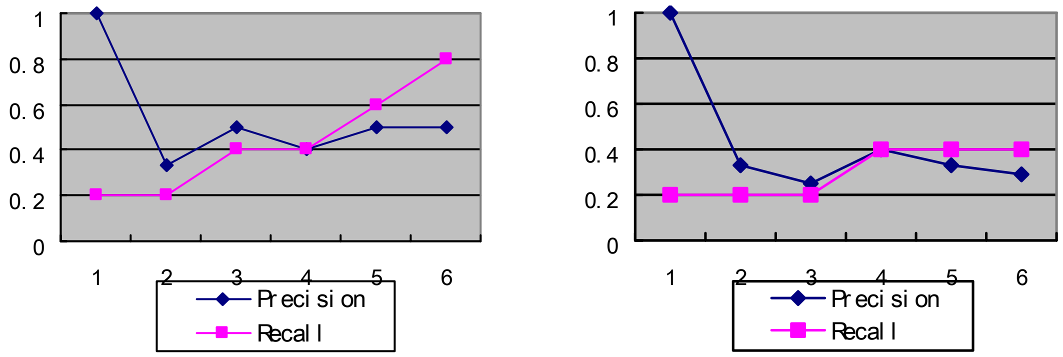
```
P(E)=ϕ;
for(i=1;i<=m;i++) {
  E_i={e_i};
  Add(P(E), E_i);
}
while(|P(E)|>s){
    Chose two clusters E_k and E_j from P(E) such that SIM(E_k ∪ E_j) is the biggest for any two clusters in P(E);
    if(SIM(E_k ∪ E_j)>θ){
      E_k=E_k ∪ E_j;
      Delete(P(E), E_j);
    }else break;
}
```

The functions used in the algorithm are defined as follows:

- *Add*($P(E)$, E_i) add element E_i into set $P(E)$.
- *Delete*($P(E)$, E_j) delete element E_j from set $P(E)$.
- According to above algorithm, if we input $BOS=\{BO_1,BO_2,\ldots,BO_m\}$, it output the clusters of business objects, each of which can be identified a entity component; if we input $BAS=\{BA_1,BA_2,\ldots,BA_n\}$, it output the clusters of business activities, each of which can be identified a process component.

Case and Experiments

In this section, we use inventory management system as example to demonstrate the proposed approach to identify reusable components from domain business model. In this case, we give three business models that represent different business requirements in inventory management domain. To evaluate the result of business components identification, we select some representative business activities from the domain business model. First we perform the component capture manually. Because the complete linkage algorithm can get the best partitions out of single, weighted, unweighted linkage algorithms [J.Davey, 2000], we compare the performance of complete linkage algorithm with that of our approach. As can be seen from figure 1, the precision and recall for the clustering algorithm is higher than form complete algorithm.



(a)Our approach (b) Complete algorithm

Fig.1: Precision and Recall

Reusable Business Component Extract Tool (RBCET)

To assist the designer to cluster similar business objects and business activities define reusable business components from domain business model, we has design and develop Reusable Business Component Extract Tool (RBCET). Figure 2 shows the process of identifying reusable business components from domain business model using RBCET.

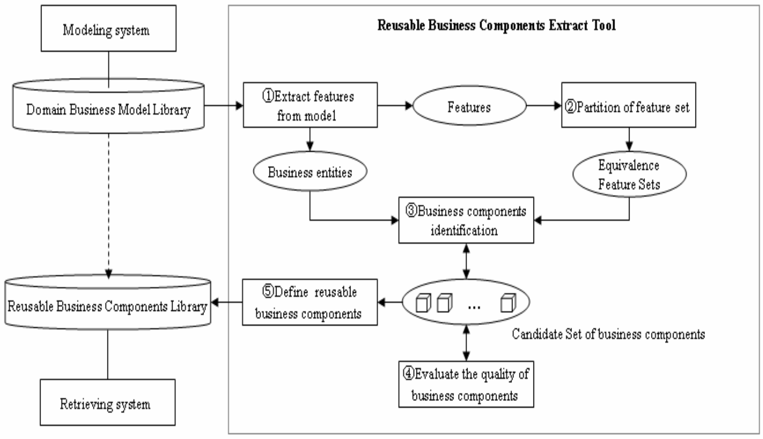

Modeling system
Domain Business Model Library
Reusable Business Components Library
Retrieving system
Reusable Business Components Extract Tool
①Extract features from model
Features
②Partition of feature set
Business entities
Equivalence Feature Sets
③Business components identification
⑤Define reusable business components
...
Candidate Set of business components
④Evaluate the quality of business components

Fig 2: The process of reusable business components Identification

Conclusion

This paper present a method of reusable business components identification based on domain business model and these components are organized in a Library called reusable business component library which can constitute a repository of the core knowledge of an enterprise in a given domain. The proposed method has been experiment on a lot of domain business model. Based on the method, the reusable business component extract tool (RBCET) has been designed and implemented to assist the designer to cluster similar business objects and business activities define reusable business components from domain business model.

Bibliography

[A.W. Brown, 1998]A.W. Brown, C. Wallnau, The current state of CBSE, IEEE Software 15 (5) (1998) 37–46.

[D'Souza, 1999] Df. D'Souza, A C Wills. Objects, Components and Frameworks with UML: the Catalysis Approach. Addison-Wesley, 1999.

[Ganesan R, 2001]Ganesan R, Sengupta S. OSBC: 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. IEEE computer Society Press, 2001, 46~55.

[G.Q. Huang, 1999]G.Q. Huang, K.L. Mak, Issues in the development and OMG-Business Object Domain Task Force, Business Object Concepts, White paper, January 1999 OMG document: bom/99-01-01.

[Jain H, 2001] Jain H, Chalimeda N, Ivaturi N, Reddy B. Business component identification - A formal approach. In: Proceedings of the 5th IEEE International Enterprise Distributed Object Computing Conference. Seattle: IEEE Computer Society Press, 2001. 183~187.

[J.Davey, 2000]J.Davey and E.Burd, "Evaluating the Suitability of Data Clustering for Software Remodularization," *The Seventh Working Conference on Reverse Engineering* (WCRE'00).2000.

[Lee SD, 1998] Lee SD, Yang YJ. COMO: A UML-based component development methodology. In: Proceedings of the 6th Asia Pacific software Engineering conference. Takamatsu: IEEE Computer Society Press, 1998. 54~63.

[P. Herzum, 2000]P. Herzum, O. Sims, Business Component Factory: A Comprehensive Overview of Component-Based development for the Enterprise, Wiley, 2000.

[Silvana Castano, 1997]Silvana Castano, Valeria De Antonellis. Engineering a library of reusable conceptual components. Information and Software Technology, 1997, 35(2): 43 ~ 57.

[Silvana Castano, 1998]Silvana Castano, Valeria De Antonellis etal. Conceptual schema analysis: Techniques and applications. ACM Trans on Database Systems, 1998, 23 (3) : 286 ~ 333.

[S.Mancoridis, 1999]S.Mancoridis, B.Mitchell, Y. Chen, and E. Gansner. Bunch: A clustering tool for the recovery and maintenance of software system structures. In *Proceedings of the International Conference on Software Maintenance*. IEEE Computer Society Press, 1999.

[Somjit Arch-int, 2003]Somjit Arch-int, Dentcho N. Batanov. Development of industrial information systems on the Web using business components. Computer in Industry 2003,50(2):231~250.

[T.A. Wiggerts, 1997]T.A. Wiggerts, "Using clustering algorithms in legacy systems remodularization," *Fourth Working Conference on Reverse Engineering (WCRE'97)*, October, 1997.

[Y. Chiricota, 2003]Y. Chiricota, F. Jourdan, and G. Melancon, "Software Components Capture using Graph Clustering," Proceedings of 11th IEEE International Workshop on Program Comprehension, 10-11 May 2003, 217-226.

[Yang Fuqing, 1999] Yang Fuqing ,Mei Hong ,Li Keqin. Software reuse and software component technology . Acta Electronica Sinica, 1999, 27(2): 68~75 (in Chinese).

Authors' Information

Meng Fanchao, Zhan Dechen and Xu Xiaofei- School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China; e-mail: Fanchao_meng@yahoo.com.cm