A Design Method of Reusable Components Based on Feature Matching

Meng FanChao, zhan dechen, and Xu Xiaofei

School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China

Summary
Aim to retrieving reusable business components, an approach of business component identification based on feature matching is proposed. First a domain business model is proposed, and the main business entities in the model have business objects, business operations, events and business activities. Based on the model that two concepts of feature and equivalent feature relation are presented, and the rule of judging equivalent feature relation and algorithm of parting feature set are given. For identifying reusable business components, a hierarchical clustering technique based on graph is proposed. In the process of clustering, we give formula of calculating similarity among business entities which extends Sorenson Coefficient. To acquire high quality business components, we give quality metrics to evaluate the quality of business components.

***Key words**:*
domain business model; feature; equivalent feature relation; reusable business component

Introduction

Component-Based Software Development (CBSD) is a key technology to tackling reconfiguration of enterprise information system. With the rapid development and maturation of the standard of component model such as CORBA, DCOM and EJB, The Component-Based Software Development has been widely applied to development of enterprise information system [1, 2]. CBSD is different from traditional methodology of software development, it mainly emphasis on retrieving reusable components, and these components retrieved are assembled to implement the functions of application system [3]. Identification and design of reusable component is the premise of CBSD, currently there are some methods have been brought forward for resolving this problem. In general, we classify those methods into two categories: Identifying from domain business model and mining from legacy system. In this paper, we mainly study on identifying reusable components from domain business model [4].

Structure analysis methods abstract domain business model as mathematical notations, such as tree structure or a graph structure. Using 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 [5], O2BC [6], and CRWD [7] and graph decomposition [8, 9]. Feature matching methods classify similar entities according to their features, and these methods also depend on the similarity measures and clustering algorithms being used. Wigglers [10] 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 entities. The representative feature matching method is the F^3 reuse methodology proposed by AIPA [11, 12]. 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. A disadvantage of this method is that the results of partition excessively depend on weights which are set by designer so that it is difficult to apply into practice [13].

In this paper, a design method of reusable business component based on feature matching is proposed. In our approach, the concepts of feature and equivalent feature relation are presented. To avoid setting weight by manual, we propose formula of calculating similarity among business entities that extends Sorenson Coefficient. To acquire high quality business components, our method gives approach of selecting optimal business components.

The remainder of this paper is organized as follows. First domain business model of enterprise information system are proposed in section 2. The design method of reusable business component is proposed in section 3, and conclusion in section 4.

2. Domain Business Model Specification

A business model can be decomposed into a set of business processes and each business process can be decomposed into business activities that involve business objects operating on a business state with business operation [14]. In this paper, we are interested in analyzing similarity of business activities whose objective is to acquire reusable business activity components to satisfy a set of information systems in same domain. In the following, we will give some basic definitions.

Definition 1 Business objects represent the concepts of real world things in enterprise such as items, product and planning etc. A business object can be defined as: *bo*=(*n*, *A*), where *n* is name of business object. $A = \{a_1, a_2, \dots, a_n\}$ is the set of attributes which can be classified into individual attribute and composite attribute. A individual attribute $a_k \in A$ (k=1,2,…,n) is defined with a name, a data type and a domain of admissible values, $a_k = (n_k, T_k, D_k)$, where n_k represents name of attribute, T_k represents data type of attribute, and D_k represents scalar domains of conventional programming languages. A composite attribute can be defined groups of individual attributes logically related and grouped.

Definition 2 Business operation represents actions on business objects. A business operation can be defined as: *bop*=(*n*, *t*, *BO*), where *n* represents name of business operation. $t \in$ {*Create*, *Modify*, *Delete*, *Transform*, *Query*} represents type of business operation [15]. *BO* is the set of business objects that are operated by *bop*.

Definition 3 Business activities are driven by events that can be classified into three categories: requirement events, notification events and time events. An event can be defined as: *e*=(*n*, *t*, *D*), where *n* is name of event. $t \in \{R, N, T\}$ is type of event, and *D* is description information about event.

Definition 4 A business activity can be defined as: $ba=(n, BO, E, BOP, \prec, \mu)$, where *n* is name of business activity. *BO*=*In-BO* ∪ *Out-BO* is the set of business objects, here *In-BO* is the set of input business objects, and *Out-BO* is the set of output business objects. *EO*=*In-E* ∪ *Out-E* is the set of events, *In-E* is the set of trigger events, and *Out-E* is the set of raise events. *BOP* is the set of business operations. $\prec$ is an irreflexive transitive binary relation on the set *E*. $\mu : E \rightarrow BOP$ is a mapping form *E* to *BOP*, which denote the trigger relations between events and business operations. Because a business operation can be triggered by more than an event, $\prec$ is not one-one mapping.

Business models belonging to same application domain are stored into a business model library, hence a domain business model can be represented as: *DBM*={*Lib-Id*, *BM-Set*, *Des*}, where *Lib-Id* is he identification of domain business model library, *BM-Set*={BM_1, BM_2, … , BM_n} is set of business models belonging to same application domain, and *Des* is the description information about the application domain.

3. Method of reusable business components design

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 [15]. 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 of the domain. Business components, on the other hand, are more coarse-grained and provide a high-level business-oriented representation of the domain, and they express future components and the relations of those components. In this paper, an approach will be given to help the designer to select similar business activities in domain business model and define reusable business activity components from them.

3.1 The Domain Thesaurus

To evaluate similarity between business activities in different business model in same domain, we refer to the domain thesaurus containing semantic information. A thesaurus usually is sets of dictionaries, every one of which stores the terms featuring about some aspect in a domain. In our domain thesaurus, the terms can be extracted from names of business elements (business objects, attributes of business objects, business operation, events and business activities) in 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 term distance between two terms n_i and n_j in a dictionary is defined as an associative distance as follows:

$$d(n_i, n_j) = \begin{cases} 0 & \text{if } n_i \text{ is the same as } n_j; \\ 1 & \text{if } n_i \text{ is a direct successor of } n_j; \\ d(n_i, n_k) + d(n_k, n_j) & \text{if } n_k \text{ is on the path from } n_i \text{ to } n_j; \\ -d(n_i, n_j) & \text{if } n_j \text{ is the succesor of } n_i; \\ +\infty & \text{if there does not exist a path from } n_i \text{ to } n_j; \end{cases}$$

The distance reflects a semantic similarity between two terms. The longer the distance is, and the less the similarity is. The similarity degree between two terms n_i and n_j can be defined as: $SIM(n_i, n_j) = 1/(|d(n_i, n_j)|+1)$. According to this definition, $SIM(n_i, n_j)$ is a numerical value in the range [0,1]. $SIM(n_i, n_j)$=1 denotes n_i and n_j are same concept, and $SIM(n_i, n_j)$=0 denotes n_i and n_j have no any relationship. If $0<SIM(n_i, n_j)<1$, it denotes n_i and n_j exist semantic relation, the bigger the value is, and the more similar n_i and n_j are.

3.2 Thesaurus Business Elements Similarity

In domain engineering, the meanings of features are very abroad [16, 17]. In our methodology, the features denote business elements that need to be implemented by information systems, such as attributes, business objects, business operation, and business activities etc. Table 1 gives the features of some business elements in domain business model. In this paper, we use symbol Ω to denote the set of all features in same application domain. In the following, we introduce the concept of similar feature relation.

Table 1: Features in domain business model

Business Element	*Feature*
Business object	Attribute
Business operation	Business object
Business activity	Event, Business object, Business operation, behavior characteristic

Definition 5 Let $f_i, f_j \in \Omega$ be two features, if f_i and f_j express same or similar function requirement in same domain, then f_i and f_j satisfy similarity relation, denoted as $f_i \sim f_j$.

Characteristic 1: Let $\sim$ be similarity relation on Ω, $\sim$ is reflexive, symmetric and transitive.

Definition 6 Let $F \subseteq \Omega$ be a feature set, for every feature $f \in F$, similar feature set of f is defined as: $[f]_\sim = \{f' \mid (f' \in F) \wedge (f' \sim f)\}$.

Definition 7 Let $\sim$ be similar relation on set F, partition on set F can be defined as: $F/\sim = \{[f]_\sim \mid f \in F\}$.

Feature set are composed of features, those features can be divide into a set of similar feature sets. In the follows, we give the algorithm of parting feature set:

Algorithm 1: Part Feature Set
Input: $F = \{f_1, f_2, \ldots, f_n\}$;
Output: $F/\sim = \{[f]_\sim \mid f \in F\}$;
Algorithm description:

```
$TF \leftarrow F$; $F/\sim \leftarrow \Phi$ ;
for ($f_k \in TF$ )
{
  Add ($[f_k]_\sim, f_k$ );
  Remove ($TF, f_k$);
  for ($f_j \in TF$)
  {
    if ($f_j \sim f_k$)
    {
      Add ($[f_k]_\sim, f_j$);
      Remove ($TF, f_j$);
    }
  }
  Add ($F/\sim$, $[f_k]_\sim$);
}
```

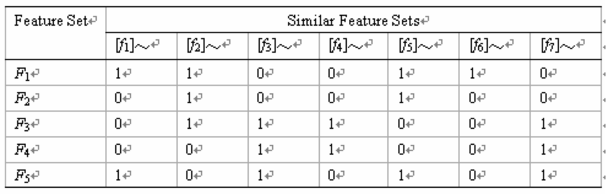
The functions used in the algorithm are defined as follows:

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

In this paper, we use feature matrix shown as table 2 to calculate the similarity between two feature sets. Different to the approach followed by Davey and Burd [18], we use feature sets as entities and similar feature sets as attributes of these entities. Let $M=[F, F/\sim]$ be a feature matrix, where $F=\{F_1, F_2, \ldots, F_m\}$ is the set of feature sets, and $F/\sim = \{[f_1]_\sim, [f_2]_\sim, \ldots, [f_n]_\sim\}$ is the set of similar feature sets which are constructed form all features in F. If $F_i (1 \le i \le m)$ exists a feature f that belongs to $[f_j]_\sim (1 \le j \le n)$, then sets m_{ij}=1, else sets m_{ij}=0.

In feature matrix, there are four match relations between two feature sets, they are 1-1 called m_1, 1-0 called m_2, 0-1 called m_3, and 0-0 called m_4. Assume their amounts are respectively a, b, c and d. There are many methods to calculate the similarity degree between two entities such as Jaccard coefficient, simple matching coefficient and soerenson coefficient etc. In this paper, we apply soerenson coefficient to calculate similarity degree between two feature sets. Let $SIM(F_i, F_j)$ ($F_i, F_j \subseteq \Omega$) be the similarity degree between feature sets F_i and F_j, according to soerenson coefficient, $SIM(F_i, F_j)=2a/(2a+b+c)$. For example, in table 2, there are five feature sets with seven similar feature sets. Based on the definition, we obtain for F_1 and F_2 that a=2, b=2, c=0 and d=3, so $SIM(F_1,F_2)$=2/3.

Table 2 Feature Matrix



Feature Set	Similar Feature Sets						
	$[f_1]_\sim$	$[f_2]_\sim$	$[f_3]_\sim$	$[f_4]_\sim$	$[f_5]_\sim$	$[f_6]_\sim$	$[f_7]_\sim$
F_1	1	1	0	0	1	1	0
F_2	0	1	0	0	1	0	0
F_3	0	1	1	1	0	0	1
F_4	0	0	1	1	0	0	1
F_5	1	0	1	0	1	0	1

Definition 8: Let F_i, $F_j \subseteq \Omega$ be two feature sets, if F_i and F_j satisfy: $SIM(F_i, F_j)>\beta$ (β is threshold of similarity degree between feature sets), then F_i and F_j satisfy similarity relation, denoted as $F_i \sim F_j$.

According to the definitions, we give the methods to judge similarity relations of attributes, events, business objects and business operations in domain business model as follows:

- Let $a_i=(n_i, T_i, D_i)$ and $a_j=(n_j, T_j, D_j)$ be two individual attributes, if a_i and a_j satisfy: $(SIM(n_i, n_j)>\alpha) \wedge (T_i=T_j) \wedge ((D_i = D_j) \vee (D_i \subset D_j) \vee (D_j \subset D_i))$, then $a_i \sim a_j$.
- Let $a=f(a_1,a_2,\ldots, a_n)$ and $b=g(b_1,b_2,\ldots,b_m)$ be two composite attributes, f and g is respectively name of attribute a and b, if a and b satisfy condition: $SIM(f, g)<\alpha \wedge (n=m) \wedge ((a_1 \sim b_1) \wedge (a_2 \sim b_2) \wedge \ldots \wedge (a_n \sim b_m))$, then $a \sim b$.
- Let $e_i=(n_i, t_i, D_i)$ and $e_j=(n_j, t_j, D_j)$ be two events, if e_i and e_j satisfy: $(SIM(n_i, n_j)> \alpha) \wedge (t_i = t_j)$, then $e_i \sim e_j$.
- Let $bo_i=(n_i, A_i)$ and $bo_j=(n_j, A_j)$ be two business objects, if bo_i and bo_j satisfy: $(SIM(n_i, n_j)>\alpha) \wedge (SIM(A_i, A_j)> \beta)$, then $bo_i \sim bo_j$.
- Let $bop_i=(n_i, t_i, BO_i)$ and $bop_j=(n_j, t_j, BO_j)$ be two business operations, if bop_i and bop_j satisfy: $(SIM(n_i, n_j)>\alpha) \wedge (t_i= t_j) \wedge (SIM(A_i, A_j)> \beta)$, then $bop_i \sim bop_j$.

In above conditions, α is the threshold of names similarity. A threshold is imposed on similarity relations to filter out names with low similarity degree.

3.3 Business component clustering algorithm

Given a domain business model, we can calculate the similarity between business activities in different business models. Intuitively, if two business activities have a lot of common features, then they are likely to express same or similar function requirement. Reusable business activities components can be defined from clusters of similar business activities by abstracting their common features. Business activity can be divided into five facets: input business object facet, output business object facet, trigger event facet, raise event facet and business operation facet. Every facet is made up of a set of features.

In this paper, we extend Sorenson coefficient to calculate similarity degree among more than two feature sets. In multi feature sets, there are three match relations: T_1, T_2 and T_3, where T_1 is represented as (1,1, …, 1), T_2 is represented as (1,1, …, 0, …, 1), and T_3 is represented with (0,0, …, 0). Assume the amount of T_1, T_2 and T_3 are respectively a, b and c, so the similarity among multi feature sets $FS_i=\{F_{i1},F_{i2}, \ldots ,F_{ik}\}$ can be defined as: $SIM(FS_i)= a/(a+ \sum_{i=1}^{b} r(b_i))$, where $b_i \in T_2$, $r(b_i)$ represents proportion of 0 in b_i. If the amount of feature sets is 2, then $SIM=2a/(2a+b)$ which is Sorenson coefficient. For example, in table 2, the amount of T_1, T_2 and T_3 is respectively 2, 3 and 3 among F_3, F_4 and F_5, so $SIM=1/(1+ (2/3 + 2/3 + 2/3 +1/3+2/3+2/3))=3/14$.

We use undirected weighted graph to store similarity between two business entities. In this graph, vertexes represent business entities, and edges that store similarity between adjoining vertexes represent their relationship. Formally, the undirected weighted graph is defined as: $G=\{V, E\}$, where $V=\{v_1, v_2, \ldots, v_n\}$ is set of vertexes, $E=\{(v_k, v_j, w) \mid v_k, v_j \in V\ (v_k \neq v_j);\ w=SIM(v_k, v_j)\ (w>0)\}$ is set of edges. According to above definition, if $SIM(v_k, v_j)=0$, there isn't edge between v_k and v_j.

To identify reusable business components, we propose a hierarchical clustering method based on graph. In this method, we use above formula proposed calculate similarity between two clusters. The algorithm for identifying reusable business component is depicted as following:

Algorithm 2: Clustering algorithm based on graph

Input: Feature matrix $M=[BE\text{-}Set, F/\sim]$; Graph $G = \{V, E\}$ storing similarity between two business entities in *BE-Set*; θ as the similarity threshold to choose the high reusable business components; s is the size threshold for the result of clustering.

Output: partition of set *BE-Set*: $P(BE\text{-}Set)=\{BE\text{-}Set_1, BE\text{-}Set_2,\ldots, BE\text{-}Set_k\}$, partition satisfy condition: (1) $BE\text{-}Set_1 \cup BE\text{-}Set_2 \cup \ldots \cup BE\text{-}Set_k = BE\text{-}Set\ (k\leq S<m)$; (2) $BE\text{-}Set_i \cap BE\text{-}Set_j=\phi\ (1\leq i, j\leq k; i\neq j)$.

Algorithm description:

```
1  $P \leftarrow \phi$;
2  for($e \in E$)
3  {
4    if ($w(e) < \theta$)
5      DeleteEdge($G$, $e$);
6  }
7  while ($|V| \geq s$)
8  {
9    $e$=GetMaxweightEdge ($G$);
10   ReConstructGraph ($G$, $M$, $e$, $\theta$);
11 }
12 $P \leftarrow V$;
```

In above algorithm, first delete these edges whose weights are lower than threshold θ in graph G. Then, according to hierarchical algorithms, once select the edge whose weight is high than others, and then group them into a new vertex. The functions used in the algorithm are defined as follows:

- DeleteEdge(G, e) delete edge e from graph G;
- GetMaxweightEdge (G) return maximum weighted edge form graph G;
- ReConstructGraph (G, M, e, θ) is used to reconstruct graph G, when combing vertex u and v that belong vertexes of edge e. When combining vertex u and v to a new vertex z, assume that there is a vertex k are linked with both vertex u or vertex v, the weight of edge (k, z) can be recalculated in term of feature matrix M. In process of each cluster, these edges whose weights are lower than threshold θ are deleted. In the following, we use business objects in Table 2.

to illustrate the process of clustering. Fig 2(a) shows the weighted graph storing all weights between two business objects. Let θ=1/2 and s=1, because weight of edge (BO_1, BO_3) and edge (BO_2, BO_5) are lower than θ, they are deleted from G. Fig 2(b) shows the graph whose corresponding edges are deleted. Fig 2(c) and 2(d) show the middle results in the process of clustering, and fig 2(e) is the last result of clustering. According to our method, it can identify two business components: BC_1={BO_1, BO_2} and BC_2={BO_3, BO_4, BO_5}.

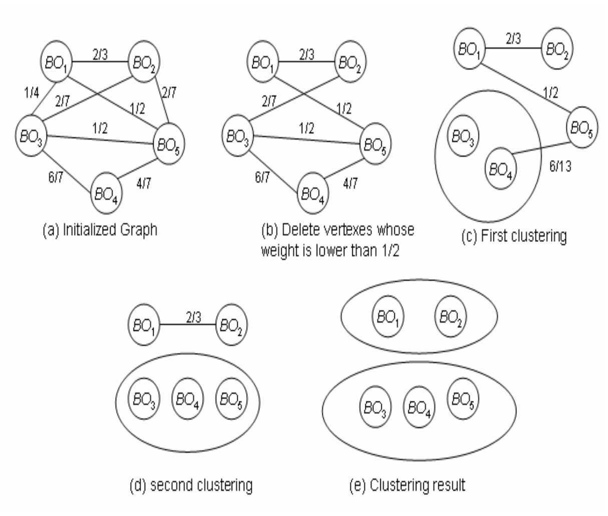


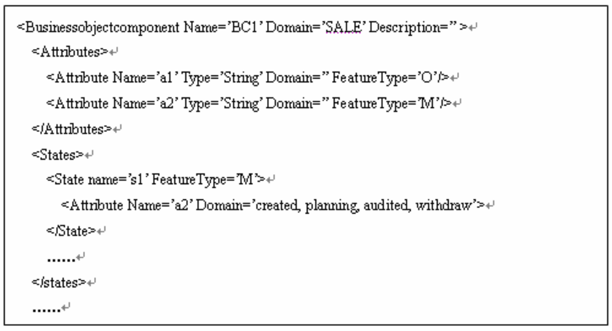
Figure 2 the process of clustering

3.4 Definition of reusable business components

The designer uses selected clusters of business entities to define reusable business components which are stored into reusable business components library for reuse in cross application systems. In general, a reusable business component includes three types of features: Mandatory feature, Optional feature and Alternative feature.

- Mandatory feature: All business models in a domain business model possess features which express common business requirements.
- Optional feature: Part of business models but not all in a domain business model possess features which express diversity business requirements.
- Alternative feature: Different business models in a domain business model have different disposal methods on same business requirement, which express changeability requirements.

XML, as a standard documentation, supports both description of business components in the phase of requirements and provides Web-native programming development. In this paper, we use XML represent business components identified above. For example, In Figure 2, we can identify two reusable business object components: BC_1 and BC_2. Figure 3 gives the representation of BC_1.



```
<Businessobjectcomponent Name='BC1' Domain='SALE' Description='' >
  <Attributes>
    <Attribute Name='a1' Type='String' Domain='' FeatureType='O'/>
    <Attribute Name='a2' Type='String' Domain='' FeatureType='M'/>
  </Attributes>
  <States>
    <State name='s1' FeatureType='M'>
      <Attribute Name='a2' Domain='created, planning, audited, withdraw'>
    </State>
    ......
  </states>
  ......
```

Figure 3 Representation of Reusable Object Component

4. Conclusion

This paper present a design method of reusable business components based on feature matching. The design method proposed has been applied into business components design of ERP [19]. Based on the method, the reusable business component extract tool (RBCET) has been designed and implemented to assist the designer to cluster similar business entities and define reusable business components. These components identified are organized in a Library called reusable business component library that constitute a repository of the core knowledge of an enterprise in a given domain. Developer can develop software in term of the definition of business components in the library and re-users can retrieve reusable component from the library.

Acknowledgments

This paper is supported by the National Natural Science Foundation of China under Grant No.60573086 and the National Research Foundation for the Doctoral Program of Higher Education of China under Grant No.20030213027.

References

[1] Df. D'Souza, A C Wills. Objects, Components and Frameworks with UML: the Catalysis Approach. Addison-Wesley, 1999.
[2] 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.
[3] Yang Fuqing ,Mei Hong ,Li Keqin. Software reuse and software component technology . Acta Electronica Sinica, 1999, 27(2): 68~75 (in Chinese).
[4] Li Gang, Jin Maozhong. A design method for reusable components. Journal of Computer Research and Development(in Chinese), 2000, 37 (5) : 609～615
[5] 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.
[6] 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.
[7] 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.
[8] 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.
[9] J. Luo, W. Zhao, T. Qin, R. Jiang, L. Zhang, and J.Sun, "A Decomposition Method for Object-Oriented Systems Based on Iterative Analysis of the Directed Weighted Graph," Journal of Software (in Chinese).
[10]T.A. Wiggerts, "Using clustering algorithms in legacy systems remodularization," Fourth Working Conference on Reverse Engineering (WCRE'97), October, 1997.
[11]Silvana Castano, Valeria De Antonellis. Engineering a library of reusable conceptual components. Information and Software Technology, 1997, 35(2): 43～57.
[12] Silvana Castano, Valeria De Antonellis etal. Conceptual schema analysis: Techniques and applications. ACM Trans on Database Systems, 1998, 23 (3) : 286～ 333.
[13] L i Gang, Jin Maozhong. A design method for reusable components[J]. Journal of Computer Research and Development, 2000, 37 (5) : 609～615
[14] WANG Zhong - jie , XU Xiao - f ei , ZHAN De – chen RO-BPM: A reconf iguration - oriented business process model. Computer Integrated Manufacturing Systems ,2004 ,10 (11) :1349 - 1355(in Chinese)
[15] Castno, Silvana; De Antonellis, ValeriaA ramework for expressing semantic relationships between multiple information systems for cooperation. Information system. 1998,23(6): 253-277
[16] Kang KC, Cohen SG, Hess JA, Novak WE, Peterson AS. Feature-Oriented domain analysis feasibility study . Technique Report, SEI-90-TR-21, Pittsburgh: Software Engineering Institute, Carnegie Mellon University, 1990.
[17] Chastek G, Donohoe P, Kang KC, Thiel S. Product line analysis: A practical introduction]. Technique Report, SEI-2001-TR-001,Software Engineering Institute, Carnegie Mellon University, 2001.
[18] J.Davey and E.Burd, "Evaluating the Suitability of Data Clustering for Software Remodularization," The Seventh Working Conference on Reverse Engineering (WCRE'00).
on Software Maintenance. IEEE Computer Society Press, 1999.
[19]ZHAN Dechen, XU Xiaofei , LI Chengyan. Research on framework of ERP system with double control from time and cost [J] . Computer Integrated Manufacturing Systems ,2002 ,8 (8) :635~ 639

Meng **Fanchao** received the B.S.and M.S. degrees from Heilongjiang University in 1997 and Harbin science and technology university in 2000, respectively. He is presently a Ph.D candidate at Center of Intelligent omputing of Enterprises, School of Computer Science and Technology in Harbin Industrial of Technology of China. His current research areas include Enterprise Modeling, ERP, and software engineering..

ZHAN Dechen He is He is a doctor, professor and doctoral supervisor of HIT, His research interest includes modern enterprise management, data and knowledge engineering , software reconstruction and reuse.

XU Xiao-fei, He is He is a doctor, professor and doctoral supervisor of HIT, His research interest includes enterprise intelligent computing, management and decision information system, ERP, supply chain management, E-commerce and business intelligent, knowledge engineering and application.