DMAL-BPIDE: A Distributed Multi-Abstraction-Level Business Process IDE

Xiaotian Xu, Hailong Sun, Xu Wang
School of Computer Science and Engineering
Beihang University
Beijing, China
{xuxt, sunhl, wangxu}@act.buaa.edu.cn

Abstract—How to raise the efficiency of collaboration of modelers and bring down the cost of reusing processes are key problems in modern process developing activities. In the role-oriented way to organize modelers and the single-instance-based way to organize processes, traditional practices failed to tackle the problems. This paper describes an abstraction-level-based solution which organizes modelers and processes in the context of the top-down modeling hierarchy, and introduces a distributed BPMN based business process IDE called DMAL-BPIDE, with which enterprises could regulate and optimize modeling activities, effectively organize modelers through task distributing mechanism, and facilitate the reuse of processes.

Keywords-business process; process modeling; BPM; process reuse; BPMN

I. INTRODUCTION

With the increasing popularity of SOA technology and the perfected development of the supporting platforms, web service as a distributed computational model has become one of the mainstream solutions to solve the problems of interactions among heterogeneous IT applications in the context of the internet. More and more enterprises have chosen service composition as a way to build and reuse their IT systems aiming to fulfill their core business values efficiently and agiliey as well as provide endurable benefits and flexible strategies to adapt the constantly changing business requirements [1]. The business process driven web services composition is a popular solution in which process plays a key role in the definition and constraint of IT systems. By managing the designing and reengineering of business processes, enterprises could efficiently build and modify IT systems logic/data models. Meanwhile the solution would make it easy for process modelers to focus on business logics rather than underlying technical details [2] [12].

Yet the considerable division of business domains and the complexity of designation structure and department organization in modern enterprises lead to an increasingly hierarchic and complicated process of business process development. The aim of process modeling technics thus turns from customization towards unique business instance to reuse-oriented process reengineering [3]. Hence proper organization and persistence towards excessive amount of intermediate process products should be taken in the context of process development in order to provide resources to process reuse. On the other hand, with the enhancement of globalization of enterprises, business people are regularly located at different areas; and this happens especially when they are in different business fields or separate departments. An efficient collaboration thus is in dire need to organize the modelers from different places. In such a trend, it is difficult for role-based traditional practices to manage the distribution of task and authorization appointment automatically [11] as well as to organize large amount of intermediate processes. In another word, there are still two compelling problems in the traditional development activities of business processes:

- The lack of organization of business modelers
- The difficulty in process reuse

In this paper, we introduces an abstraction-level based business process modeling method (ALBBPM) which provides the process modeling activities with an efficient way to manage the modelers and process products by defining a hierarchic abstraction-level structure of modeling process. ALBBPM also establishes a mechanism to manage the distribution of tasks and the classification of authority in the context of abstraction levels. On the basis of ALBBPM method, distributed multi-abstraction-level business process integrated developing environment (DMAL-BPIDE) is implemented. While supporting the ALBBPM method, the system also provides uniform modeling view for business modelers by adopting a distributed solution which will help to tackle the problem of collaborations in different location and thus will ease the organization of modelers.

The remainder of the paper is organized as follows: related work is discussed in section II; section III introduces process and principles of ALBBPM method and gives use scenarios; section IV describes the designing of DMAL-BPIDE, and introduces features of major modules; in section V we study a modeling case to illustrate availability of the system and improvement the solution has made compared with traditional practices; section VI concludes this work.

II. RELATED WORK

How to facilitate the collaboration of modelers and cut the cost of process reuse provoke rapt attention of studies in the field of BPM. [9] proposed an improvement to process modeling through reuse of process fragments based on $\pi$-calculus which provides only a reuse recommendation but no way to merge the fragment into processes directly. [10]
suggested that business process should be regarded as a kind of knowledge and defined a process component model for knowledge reuse, but does not provide a practical way to get the concrete process structure from the model. [13][14] presented an process interaction design solution and associated operations to support the design of business collaborations at successive abstraction levels, but the method isn’t fit for other structures in business processes.

In the practices of collaboration developing tools, IBM Websphere Business Modeler [17] adopts the solution of BPMN-2-BPEL and uses process-central method to organize relations and resources which cares little, if any, about the hierarchic relationship between modeling roles and processes. BPM Studio [18] from Oracle supports multi-view towards different roles and definition of organizations, but division of processes and task distribution are managed manually. Other business process modeling tools like Eclipse STP BPMN Modeler [19], BizAgi Process Modeler [20] and Microsoft Visio BPMN [21] don’t support authorization of roles.

In comparison, DMAL-BPIDE places much emphasis on the organization method of modelers and processes and provides an approach to construct a uniform structure model for the realization of mapping between roles and process products in modeling cases to support the modeling activities.

III. ABSTRACTION-LEVEL-BASED BUSINESS PROCESS MODELING METHOD

A. Scenarios

As business division of an enterprise goes finer gradually, business knowledge which a separate modeler should hold is limited to some specified narrow field; and this limitation of business understanding proves to be more conspicuous when division level goes deeper. The limitation is also reflected in modeling activities: customization of an integral process of single business instance requires involvement of more experts in different fields. Under general circumstance, the business managers from higher abstraction level carry out partition of processes and hand the fragments to be refined to modelers who are familiar with related fields. We take modeling of business process as a top-down process that processes are built up by handing refining tasks continuously from modelers of higher abstraction levels to ones of lower levels. The distribution of refining task is shown in Fig. 1.

The ALBBPM method proposed in this paper adopts BPMN specification [6] as the expression form of business processes. BPMN specification which supports uniform perspective of multiple modeling roles is gradually replacing solutions like BPEL4WS [7] to become a mainstream way to support process-driven WS composition. By adopting BPMN as the business process modeling language, it is possible to provide uniform view to different modeling roles and, as a result, will facilitate business staffs in shielding IT details they are not concerned with in modeling processes. It also helps the technical developers to understand the business context of the process they are engaged with and therefore provides a smooth way to bridge the gaps between business and IT in the solution of process-driven WS composition [8].

B. Abstraction-Level Based Modeling Case

ALBBPM modeling process is the process of business process build-up and continuous refactor which is constrained by the frame of ALBBPM modeling method. We define the business process modeling case as the process products’ creation, refining, modification and other modeling and reengineering activities and accompanying system processes such as definition of roles and distribution of tasks which are within the extent of a single business instance.

Fig. 2 shows entities and the relationship between them contained in a typical business process modeling case. A business process modeling case should describe unique business instance which might constantly change with the performing of modeling activities. It is listed below some vital definitions in order to better illustrate the conception of business process modeling case.

**Definition 1 Process:** Process is defined as a tuple \( P < V, E > \), where:
- \( V \) is a finite set of activities
- \( E \) is a finite set of flow objects
- \( V, E \subseteq G_{BPMN} \) \( G_{BPMN} \) is the diagram set of BPMN

The set of processes is denoted as \( V_P \). If \( \exists u \in P(V) \), and \( u \in G_{sub} \) \( G_{sub} \) is the set of sub-process diagram, then \( P < V, E > \) is called an expandable process; If \( |V_{Task}| = 1 \) and \( \forall u \in V_{Task} \), \( u \in G_{sub} \), then \( P < V, E > \) is called an original process denoted as \( P_{origin} \), as shown in Fig. 3.
Figure 3. ALBBPM original process

**Definition 2 Extensible Unit:** Extensible Unit is defined as a tuple $U (Id, P_b, P_e)$, where:
- $Id$ is the identification of an extensible unit
- $P_b \in V_p, U \in P_b (V)$
- $P_e \in V_p, P_e$ is the expanded process of $U$

The set of extensible units is denoted as $V_U$, $P_e$ is called the expansion result of $U$. Extensible unit is the basic unit to act reformation to business processes. It is provided that the first process which contains operational semantics should be expanded from the unique extensible unit of $P_{origin}$.

**Definition 3 Role:** Role is defined as a tuple $R (Id, B, A)$, where:
- $Id$ is the identification of a role
- $B \in V_B$ is the business case the role belongs to
- $A \in N$ is the task distribution authority of the role

$\forall R_A, R_B \in V_C, \text{iff } A(R_A) > A(R_B), R_A \in V_{A_o}(R_B)$$V_{A_o}(R_B)$ is the set of roles to whom $R_B$ could assign tasks.

The set of roles is denoted as $V_R$. Roles are beforehand delegated certain authority according to actual department organization and personnel appointments they are involved, and are in collaboration with other modeling roles from different abstraction levels to carry out modeling activities by creating, transmitting, checking and finishing tasks.

**Definition 4 Task:** Task is defined as a tuple $T (Id, R_t, R_b, U, Tp, S)$, where:
- $Id$ is the identification of a task
- $R_t$ is the role who assigns the task, $R_t \in V_C$
- $R_b$ is the role who receives the task, $R_b \in V_C$
- $U$ is the extensible unit the task contains, $U \in V_U$
- $Tp$ is the type of the task
- $S$ is the status of the task

Task is the content unit the ALBBPM solution uses to organize the developing activities of business process such as creation, refinement and modification. As shown in Fig. 4, a task has seven statuses in its life cycle and would transform among statuses as the arrows indicated. A concrete process of task interaction will be discussed in section 2.3.

**C. Abstraction-Level and Hierarchy Structure**

The ALBBPM method organizes process products from different stage of a business process modeling case in a uniform structure by defining the abstraction hierarchy of the case. The *Spanning Tree* of a business process modeling case is defined by step as follows:
- Every node of the spanning tree should point to a unique extensible unit
- The root node of the spanning tree points to the only extensible unit in $P_{origin}$
- When a new extensible unit $U$ is created in the expansion result of the extensible unit pointed by the current node, a new child of the current node pointing to $U$ should be created and the hierarchy will extend downward to a new layer

The spanning tree generated following the rules above is defined as the hierarchy structure of a business process modeling case; every layer of the hierarchy structure is called an abstraction level of the case. Fig. 5 shows a simple hierarchy structure which contains only three layers.

As shown in Fig. 5, the hierarchy structure of case is composed of nodes which represent extensible units rather than concrete processes. It is also indicated in Fig. 5 that the first process containing real business semantic is always located at the second layer of the structure.

It can be discovered from the definition of hierarchy structure that any integral business process modeling case would generate unique spanning tree and thus would map to a unique hierarchy structure. The hierarchy structure could also mapping to the task distribution model according to Fig. 1 and Fig. 5, and this mapping could be easily realized by replacing nodes in the structure with tasks specified by the extensible units the nodes pointed to for the one-to-one correspondence between task and extensible unit. And for every task has a unique receiver, the task structure could also represent the role assignment in the modeling case. Thus the hierarchy structure is a common structure model of both processes and roles in modeling cases.

Figure 4. Life-cycle of a task

Figure 5. A simple hierarchy structure of three layers
D. Task Interaction Process

The downward extension of the abstraction level in the construction of the hierarchy model is carried out step by step by the hand-over, execution and check of tasks. Now we will discuss the detailed mechanism of task distribution and execution as well as evolution of task’s status in its lifecycle.

![Task interaction process](image)

The process shown in Fig. 6 illustrates the interactions between assigner and receiver. Process in pool 1 shows behavior of the assigner and receiver’s reaction is presented by pool 2’s process. Concrete interaction steps are as follows:

**Step 1:** \( R_A \) creates a new task \( T_0, S(T_0) = \text{New} \);

**Step 2:** \( R_A \) set the value of \( U(T_0) \) and Type(T0), then assigns \( T_0 \) to \( R_B \), \( S(T_0) = \text{Assigned} \); \( R_A \) then waits until the to-check info comes;

**Step 3:** \( R_B \) process begins after receiving \( T_0; R_B \) could decide whether to accept \( T_0 \) and assigns \( T_0 \) to another \( R_B \) if not; if \( R_B \) accepts \( T_0, S(T_0) = \text{Accept} \);

**Step 4:** After the acceptance of \( T_0, R_A \) performs specified action to \( U(T_0) \) according to \( Tp(T_0) \), and then submits to-check info to \( R_A, S(T_0) = \text{Performed} \);

**Step 5:** \( R_A \) is awaken after receives the to-check info, and checks the task \( R_B \) has just handed; if \( R_B \) has fulfilled \( T_0 \), then \( S(T_0) = \text{Fulfilled}, \) or \( S(T_0) = \text{Unqualified}, \) \( T_0 \) would be sent to \( R_B \) again for a rework until qualified by \( R_A \).

**Step 6:** \( R_B \) closes \( T_0, S(T_0) = \text{Finished} \).

E. Combination of Hierarchy

After the construction of hierarchy structure, modelers from higher abstraction levels may want to get the full picture of the brief processes they have created rather than locate and browse every sub process in hierarchy structure. It should be realized by merging sub-processes into their parent and the depth such combination will perform should be editable. We propose a practical algorithm by taking a depth-first search and the complexity is \( O(n) \) where \( n \) is the number of nodes of the spanning tree.

**TABLE I. ALGORITHM OF HIERARCHY COMBINATION**

<table>
<thead>
<tr>
<th>CombineHierarchy( )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> spanning tree T, number of layers to combine L</td>
</tr>
</tbody>
</table>

IV. DESIGN OF DMAL-BPIDE

A. DMAL-BPIDE in SOARWare

DMAL-BPIDE is a distributed business process IDE which supports ALBBP method. It is important component of the SOA Based Software Design and Production Platform [16] (SOARWare) which contains a suite of middleware and tools and basically consists of three major components including SOARBase, Service Oriented Software Production Line and Service Running Bus. The relationship between DMAL-BPIDE and SOARWare is shown in Fig. 7.

The design and implementation of DMAL-BPIDE is based on the role-based process developing toolset of SOARWare called BPIDE [8]. The new system is aimed at raising the efficiency and standardization of process modeling activities automatically and helping modelers to collaborate with each other and hence cracking with the hard nuts of reuse of processes and organization of modelers.

Fig. 8 is system architecture of DMAL-BPIDE. The abstraction level management is designed as central module which organizes the management of modeling case, process products, tasks and presentation. We describe features of some modules briefly in the following part of the section.

![System architecture of DMAL-BPIDE](image)

B. Functional Components of DMAL-BPIDE

Abstraction level management is the core module of DMAL-BPIDE depending on which other system modules organize functions. It is mainly in charge of the generation and evolution of hierarchy structures of modeling cases and supports the modules of presentation and persistence. In an instance of process modeling, the module should maintain unique hierarchy structure of the modeling case and organize
other module to respond according to proper strategies when the hierarchy structure is changed (new request of process refining, modification towards existing processes etc.).

![Figure 8. DMAL-BPIDE system architecture](image)

Modeling case management manages definition of roles, evolution of cases and combination of hierarchy structures in modeling activities. It controls the creation, maintenance and destroy of modeling roles and realize the constraint of user behavior by maintain the user’s role authority files of different modeling cases. It also provides an implementation of hierarchy combination algorithm in section 2.3.3 to help modelers of higher levels to browse full picture of processes.

The process product management is in charge of process searching and management of process versions. It provides tools including process query tool and process-unit mapping tool which facilitate related actions towards processes. Every process developed in DMAL-BPIDE will be persistent in the business process repository in context of the modeling case in order to facilitate reuse or backtracking.

**Definition 5 Hierarchy Version:** hierarchy version is defined as a two-dimensional array $$\text{Ver}_U[\text{Id}_U][\text{Ver}_U]$$, where:

- $$U \in \text{Ver}_U$$
- $$\text{Id}_U$$ is the identification of $$U$$
- $$\text{Ver}_U$$ is the version of $$U$$

As shown in Fig. 9, the module always presented the latest version of corresponding process and can, if needed, search from the business process repository for specified history versions of the processes the user has queried.

![Figure 9. Example of hierarchy version](image)

Task management module manages organization, edit permission, distribution and lifecycle of tasks. By classify tasks of different status, the module will invoke presentation module to organize information and valid operation of tasks in different view. The function of category reminding is provided to help users to find out tasks to be engaged in.

The module of presentation management provides presentation of entities and organization of perspectives. It requires edit permission of current user and then presents the proper visual part of processes according to the result, and thus achieves the purpose of differential presentation.

V. CASE STUDY

To illustrate the availability of DMAL-BPIDE, we study a typical business process developing case of an enterprise’s annual carnival. Senior business manager from company’s headquarters first create a modeling case of an enterprise’s annual by using the quick actions provided by DMAL-BPIDE, and then build up the first brief process by refining the extensible unit of the auto-generated original process.

As shown in Fig. 10, the upper process is an original process which is auto-generated according to the name of the modeling case, and the other is the process refined from the upper one. The manager then creates new task by selecting the to-be-refined diagrams and taking the ‘create task’ action. The system will then replace the selected diagrams into sub-process elements and pop up the task editing view. The view in left bottom gives a list shows business cases the manager is involved in and processes he has created in every case.

![Figure 10. DMAL-BPIDE model refinement perspective](image)

V. CASE STUDY

To illustrate the availability of DMAL-BPIDE, we study a typical business process developing case of an enterprise’s annual carnival. Senior business manager from company’s headquarters first create a modeling case of an enterprise’s annual by using the quick actions provided by DMAL-BPIDE, and then build up the first brief process by refining the extensible unit of the auto-generated original process.

As shown in Fig. 10, the upper process is an original process which is auto-generated according to the name of the modeling case, and the other is the process refined from the upper one. The manager then creates new task by selecting the to-be-refined diagrams and taking the ‘create task’ action. The system will then replace the selected diagrams into sub-process elements and pop up the task editing view. The view in left bottom gives a list shows business cases the manager is involved in and processes he has created in every case.

![Figure 11. DMAL-BPIDE Task Assignment Perspective](image)
history hierarchy structure, or refining the current extensible unit. She/he could reuse processes from a company. After receiving the task, the modeler should will be sent to the specified receiver through intranet of the business roles to whom the manager could assign tasks. Task will then be sent to the assigner to have a check.

As shown in Fig. 11, the manager should edit the information of the assigning task in the task editing view. DMAL-BPIDE will generate information of extensible unit contains in current task automatically and provide a list of business roles to whom the manager could assign tasks. Task will be sent to the specified receiver through intranet of the company. After receiving the task, the modeler should perform the refining action towards process specified by the given extensible unit. She/he could reuse processes from a history hierarchy structure, or refining the current extensible unit by manually modeling and distributions of new refining tasks. Task will then be sent to the assigner to have a check.

As shown in Fig. 12, the manager checks performed task after receiving the to-check info and gives feedback to task executor indicating whether it is qualified. After finishes certain stage of the modeling case, the manager could combine the process hierarchy showed in the right-top view of the perspective with the assistance of the hierarchy combination tool to get a full picture. The process shown in Fig. 12 is result of two-layer combination of the hierarchy which already presents clear business semantic. The manager could hence hand the process to technical developers for an orchestration to bind concrete execution info to the process.

VI. CONCLUSION AND FUTURE WORK

In this paper, we introduced an abstraction-level based business process modeling method called ALBBPM and presented a distributed IDE supporting the method called DMAL-BPIDE. By using DMAL-BPIDE enterprises could control and improve the process development and cut the cost of process reuse. The paper also studied a modeling case to illustrate the availability of DMAL-BPIDE.

We plan to add features to DMAL-BPIDE such as the auto-recommendation of matching processes and auto-modification of processes driven by changes in processes from higher abstraction levels which will provide better support for the business process development.

ACKNOWLEDGMENT

This work was partly supported by China 863 program under Grant No. 2007AA010301. And we thank Zhongrui Che for his contribution to the development of DMAL-BPIDE.

REFERENCES

[16] Hailong S, Xudong L: SOARWare: A Service Oriented Software Production and Running Environment. APWeb 2010