Inf Syst Front (2011) 13:171–189
DOI 10.1007/s10796-009-9190-z

Integrating constraints to support legally flexible business processes

Yuqing Sun · Joshua Zhexue Huang · Xiangxu Meng

Published online: 23 June 2009
© Springer Science + Business Media, LLC 2009

Abstract Flexible collaboration is a notable attribute of Web 2.0, which is often in the form of multiple users participating different activities that together complete a whole business process. In such an environment, business processes may be dynamically customized or adjusted, as well as the participants may be selected or attend uncertainly. So how to ensure the legitimacy of a business process for both security and business is increasingly critical. In this paper, we investigate this problem and introduce a novel method to support legally flexible business processes. The proposed Constraint-based Business Process Management Model incorporates constraints into the standard activities composing a business process, where the security constraints place restrictions on participants performing the activities and business constraints restrict the dependencies between multiple activities. By the assembly operations, business processes can be dynamically generated and adjusted with activities, that are obliged to the specified constraints. Several algorithms are presented to verify the consistency of constraints and the soundness of the generated business processes, as well as to perform the execution planning to guarantee the correct execution of a business process on the precondition of satisfying all constraints. We present an illustrative example and implement a prototype for the proposed model that is an application of property rights exchange for supporting legal business processes.

Keywords Authorization · Constraint · Business process · Flexibility

Y. Sun (✉) · X. Meng
School of Computer Science and Technology,
Shandong University, No. 27 Shanda South Road, Jinan,
Shandong, China 250100
e-mail: sun_yuqing@sdu.edu.cn

X. Meng
e-mail: mxx@sdu.edu.cn

J. Z. Huang
E-Business Technology Institute, University of Hong Kong,
Pokfulam Road, Hong Kong, China
e-mail: jhuang@eti.hku.hk

1 Introduction

Web 2.0 has several attributes that clearly set it apart from the current Web, such as content aggregation, community support, active collaboration and loose business process manipulation etc. One notable attribute is the flexibility of Web based business process, in which activities may be dynamically customized and adjusted, as well as the participants may be selected or attend uncertainly. In such an environment, the legitimacy requirements with respect to both security and business are much more critical for business processes than ever. From the view of security, the access to the activities related to sensitive information should be ensured by the authorized users so as to avoid illegal operation. From the business view, the sequences between activities should follow the business rules in an organization so as to satisfy the dependency relationships between activities and guarantee the correct execution of a business process. From the view of management, the generation of a business process is supposed to be customized as more as possible by the reuse of existing activities and constraints so as to reduce the complexity of management and deploy flexible business processes efficiently and economically (Gordon and Loeb 2006). Wholly considering above aspects is absolutely a challenge since it is difficult to satisfy the flexibility without compromising the security and manageability.

Security constraints are regarded as a fundamental mechanism to enforce high-level security requirements (Ahn and Sandhu 2000). For example, Separation of Duty (SOD) constraints restrict the access rights to sensitive tasks to multiple persons so as to prevent users from exceeding a reasonable level of authority for their position. Another example is the constraint of Binding of Duty (BOD) that restricts the users taking a certain responsibility should take another responsibility. The security constraints of Cardinality specify the upper bound and lower bound on the number of users assigned to a critical activity. On one hand, assigning too many users to a critical task may increase the chance of fraud. On the other hand, assigning too few users to a task may place too heavy a workload on those users and is not resilient to the absence of those users. Similarly, business constraints can be used to express business rules. For example, the prerequisite relationship specifies a sequential dependency of two activities such as the *grant funding* of a project being allowed only after the *project proved*. The example of the parallel relationship of activities can be thinking of the restriction between two reviewers of a project.

There are many literatures discussing about the specification of constraints on business process (Bertino et al. 1999), such as, TBAC model (Thomas and Sandhu 1997), TMAC model (Thomas 1997) etc. Some work discuss the authorization management in business process based on Web services (Koshutanski and Massacci 2005b; Yu et al. 2007). The role based access control model for WS-BPEL (RBAC-WS-BPEL) supports the specification of authorization information stating if a role or a user is allowed to execute human activities composing the process (Paci et al. 2008a). The authors also discuss the resiliency problem in the RBAC-WS-BPEL model (Paci et al. 2008b). However, these works place constraints only on pre-defined business process where all possible processes are well defined in the design stage, which is not applicable in many cases. Especially in the Web2.0 era, more active and flexible business processes are motivated to create at runtime and dynamically adjusted according to the (temporary) change of business requirements. In such cases, if we still adopt the conventional method to specify the constraints after each business process is created, it would be inefficient and error-prone. So how to enable the constraints automatically applied to the dynamically created business processes and to ensure its validity are increasingly meaningful.

Considering the flexibility of business process, the often selected methods are to specify a loosely or partially coupled workflow model, and to allow the modification of the model and the selection of an appropriate path in a per-instance basis, taking into account of the prevailing circumstances (van der Aalst and Berens 2001). The dynamic flexibility is generally achieved by the adjustment of a part of an ongoing instance (Mangan and Sadiq 2002). However, none of them takes security constraints into consideration, which unavoidably lets the processes risk various threats. Overall, to the best of our knowledge, there is no previous work on a full solution to the legitimacy problem of such flexible business processes.

In this paper, we investigate this problem and propose a novel approach to support legally flexible business process by integrating constraints into the customization and execution of business processes. The proposed Constraint-based Business Process Management Mode (CBPM for short) introduces a new concept of *standard activity* that represents the basic activity composing business processes. The high-level security and business requirements are expressed as constraints on each activity and on the relationships between activities. The participants are dynamically assigned the authorizations by being evaluated with their owned attributes (such as credentials) against the qualification requirements of the roles performing an activity. The flexibility of business process is implemented by the assembly operations on predefined standard activities and its legitimacy is ensured by the verification of business constraints. The soundness of such flexible business processes is proved by the proposed business process logic net.

We present a few algorithms to verify the consistency of constraints, as well as to perform the execution planning to guarantee the correct execution of a business process on the precondition of satisfying all constraints. To facilitate the model implementation, we propose a system architecture and develop a prototype of an property right exchange application for supporting legal business processes. The experiments show that our model is effective in managing the legally flexible business processes and in controlling the security of the management system that is accessed daily by many users.

The paper is organized as follows. The next section presents some preliminary knowledge. The proposed CBPM model and its formal specification are presented in Section 3. The assembly operations and business process customization are discussed in Section 4. In

Section 5, we discuss the system architecture to implement the model and present a prototype. The related work is presented in Section 6. Finally, conclusions and some considerations about the future research are given.

2 Preliminaries

Role Based Access Control and attribute based authorization We have chosen the Role Based Access Control (RBAC) model as the basis of authorization management in business process in this paper because it is suitable for specifying the security requirements for a wide range of commercial, medical, government application and moreover it is being standardized as today's most influential access control model (Sandhu et al. 1996).

In RBAC, users gain permissions through memberships in roles and there is a number of constraint mechanisms to express business and security requirements, such as mutual exclusion constraints are widely used to support Separation of Duty. Assigning permissions to roles rather than users can avoid manipulation of the grantee and withdrawal of authorizations owing to the user changes.

Attribute-based access control is a widely adopted architecture in Web-based applications, such as digital library (Adam et al. 2002), in which access control decisions can be made on the basis of various user attributes in addition to simple identity. Such authorization assignments are flexible with the qualifications and characteristics of users.

In our model, we would extend the RBAC model by encapsulating access rights and roles into activities and enacting the security constraints on them, instead of directly assigning the access rights to roles in RBAC. Such binding can effectively control the access rights to the dynamically generated business processes. We also adopt the attributes-based user-role assignments mechanism to be adapt for the dynamic participants in Web 2.0 environment.

Business process management and petri net Workflow management systems are often used to coordinate and streamline the business processes in an organization. A workflow model is the formalized description of a business process, and consists of various well-defined activities and their interdependent relationships (van der Aalst and Jablonski 2000). A workflow instance denotes a particular occurrence of the business process defined by the model. Flexibility is the ability to specify a workflow from a loosely or partially defined workflow model (van der Aalst and Berens 2001), which can be achieved in two stages: modification of a model and dynamic adjustment of a part of an ongoing instance (Mangan and Sadiq 2002).

Considering modeling a business process, Petri net has been used as a formal tool since (Zisman 1977) due to a number of attractive features in business processes management (BPM), such as the graphic notation and support for complex process constructions (van der Aalst 1996). A Petri net PN is described as a three tuple $PN =< P, T, R >$, where P and T are finite sets of places and transitions, R is a set of directed arcs between places and transitions to specify causal relations denoted as $R \subseteq (P \times T) \cup (T \times P)$. A place p is called an input place of a transition t iff there exists a directed arc from p to t. The set of input places for transition t is denoted as $\bullet t$. Similarly, we can define the sets of $t\bullet$, $p\bullet$ and $\bullet p$. There is only one source place $i \in P$ and one sink place $o \in P$ that make $\bullet i = \emptyset$ and $o\bullet = \emptyset$. Every node $x \in P \cup T$ is on a path from i to o.

The state of a Petri net is called marking, representing the distribution of tokens over all places and may change during the net execution. Transitions are active components that can change the net states according to the following *firing* rules: (1) A transition t is said to be enabled iff each input place p contains at least one token, and (2) an enabled transition may *fire*. If transition t fires, it consumes one token from each input place p of t and produces one token for each output p of t. Suppose M_1, M_2,... M_n are the different states of a Petri net, the notation of $M_1 \xrightarrow{\sigma} M_n$ means: firing the sequence of transitions $\sigma = t_1 t_2 \ldots t_{n-1}$ leads to the change of the state from state M_1, via a set (possible empty) of intermediate states $M_2 \ldots, M_{n-1}$, to state M_n. The state M_n is said reachable from M_1.

Based on Petri net, we would propose a revised business process logic net (BP-net) for the specification and analysis of a business processes. By means of BP-net, any process can be dynamically customized by the standard activities and maintain the soundness.

3 The Constraint-based Business Process Management Model

In this section, we present the overview of the proposed approach and the basic terminologies. The core of our approach is the proposed Constraint-based Business Process Management Model (CBPM for short), which would be presented in detail. Specially, the formal specification of security and business constraints of the model and the verification of their consistency are given. To ensure the security and business requirements for business processes, our approach takes the following ways:

(i) Standardization of activities: The business processes in an application domain are decomposed into a set of standardized atomic activities (called standard activity) stored in the activity depository. Each activity encapsulates the necessary permissions fulfilling it and the roles who are allowed performing it.
(ii) Security and business constraints: Security requirements are expressed as security constraints specifying on standard activities while business constraints are used to represent business requirements such as the sequential or parallel dependency relationships between activities.
(iii) Flexible business process: A business process is allowed to be dynamically customized or adjusted with standard activities by assembly operation, such as deletion or addition of an activity etc. The soundness of the assembly operations are guaranteed by the proposed BP-net.
(iv) Verification: For any customized process, the validation of associated business and security constraints is ensured by the verification algorithms, while its correct execution is enforced by the execution planning.

3.1 Overview of the CBPM model

The CBPM is an extension to the widely adopted RBAC model (shown in Fig. 1), in which each component is given below. The adopted symbols that would be used in the whole paper are summarized in Table 1.

Table 1 The symbols and semantics of CBPM model

Symbol	Semantics
U	Set of users
P	Set of permissions
R	Set of roles
RH	Set of role hierarchies
A	Set of activities
SC	Set of security constraints
BC	Set of business constraints
BP	Business process
RA	Role activity assignments
URA	User role assignments

Permissions Permissions are the approvals of users to operate on one or more protected resources in a step of a business process. A resource can be a database, a table, an attribute of a table or an item of a Web page.

Roles and role hierarchies In an organizational context, a role is a job function with some associated semantics regarding the authority and responsibility. Roles in this model are authorized to perform an activity of business process, and they are hierarchically related. Role hierarchies $RH \subseteq R \times R$ is a partial order on a role set and defines an inheritance relation among roles, written as $\succeq$. $r_i \succeq r_j$ means that users who are memberships of r_i are memberships of r_j also, while all permissions assigned to r_j are inherited by r_i (Clark and Wilson 1987).

Users Users represent the participants of web applications who take on some roles to perform some human activities completing a part of a business process. In

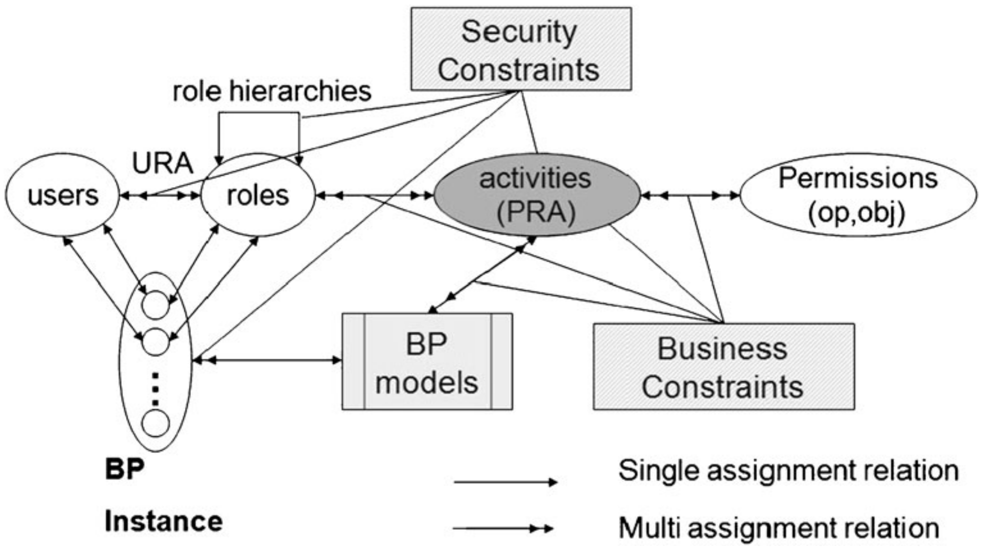


Fig. 1 The Constraint-based Business Process Management Model

our approach, the assignments of users to roles are performed against the attributes users being holding.

Activities Activity is the core concept in this model. An activity defines an atomic task of a business process that can be identified from other tasks and executed separately. It encapsulates a set of permissions with authorized roles and restricts these permissions to be valid only during execution of the activity.

By recursively decomposing a business process, the "atomic activities" can be identified, each being defined as determinable roles to perform explicit actions on specific objects, including input and output data, security and business constraints. The set of atomic activities required of a particular application domain can be achieved by analyzing all the possible business processes and abstracting the basic activity components. This can be regarded as the groundwork for the domain and the activities are then used to assemble various business processes. The following two rules specify the permission and role assignments to an activity.

Rule 1 (Permission activity mapping) *Permissions encapsulating in an activity can be effective only during the activity execution. There are many to many mapping relationships between permissions and activities. One permission can be assigned in multiple activities, while an activity can encapsulate more than one permissions.*

Rule 2 (Role activity mapping) *There are many to many mapping relationships between roles and activities. One role is allowed to assigne multiple activities, while an activity can encapsulate more than one roles.*

Constraints Security and business constraints are used to express enterprise high-level security and business requirements, which would be discussed in detail in the following section. The fine-grained dynamic security constraints are defined. For example, the dynamic Separate of Duty(DSD) sets up constraints on the invocation of user-role-activity assignments during execution.

Assembly operations and business process A business process is created by assembling activities while being obliged to the specified business and security constraints, and can be adjusted by operations, like addition and deletion, both at design and execution so as to meet flexible requirements. Assembly operations make different types of logical connections including sequential, synchronous, selective, and parallel. Details of the assembly operations will be given in Section 4.

3.2 Security constraints and consistency

We consider three kinds of security constraints in our model: mutual exclusion constraint (ME), cardinality constrain (AC), and binding of duty constraint (BD), which have been well established in previous literatures (Li et al. 2006). And we assume that constraints are specified only on activities in our model since all fundamental performances in business process are based on activities. Different constraints interact with each other and with the role hierarchies. Since in a web environment, users are allowed to separately specify security constraints, multiple constraints together may preclude one from assigning any user to a task. For example, if a ME constraint requires that no user is allowed to authorize both activities a_1 and a_2, yet a_1 and a_2 are also associated with a binding constraint (be performed by the same user), it is impossible to assign these activities to users without violating constraints. Hence, consistency checking is required so as to identify and resolve conflicts in existing constraints. Such checking is also helpful when constraints update so as to avoid the constraints unsatisfiable. In this section, we would discuss the semantics of considered constraints and their consistency.

Mutual exclusion $\mathsf{ME}(A_s, k)$ The constraint requires that no user is allowed to be authorized to k or more activities in the set A_s, where $A_s \subseteq A$ and $1 \leq k \leq |A_s|$. This most frequently mentioned constraint is a powerful means of limiting the distribution of critical permissions and widely used to support the Separation of Duty policy or to enforce the Conflict of Interest policy (Clark and Wilson 1987). This constraint can be further classified into static mutual exclusion and dynamic mutual exclusion according to whether it is enforced on business process model or instance.

Activity-cardinality $\mathsf{AC}(a, a_l, a_u)$ The constraint requires that the number of users authorizing activity a (either by direct assignment or by inheritance) must be at least a_l but no more than a_u, where $1 \leq a_l \leq a_u$, and a_l and a_u are called the *lower-bound* and *upper-bound* of the *cardinality* of activity a, respectively. If there is not an explicit declaration, we regard a constraint $\mathsf{AC}(a, 1, \infty)$ for any activity a since it should be useful in a business process. AC constraints are useful when a job responsibility or a task represented by an activity requires multiple users. The constraints can also be used to enforce resiliency policies (Li et al. 2006). For example, the constraint AC("Issue check", $2, \infty$) requires at least two users having the right "Issue check" in the system so that even if one of them is absent, the task can still be done. Furthermore, the upper-bound a_u in this constraint may be used when we want to limit the assignment of an activity due to restrictions such as resource limits or accountability.

Binding of Duty $\mathsf{BD}(A_s)$ The constraint restricts the set of activities $A_s \subseteq A$ should be performed by the same user. The purpose of binding constraint is to simplify the management of users and roles be requiring that if a user takes a certain responsibility, then he/she must also take another responsibility. For example, $\mathsf{BD}(\{InitProj, ModiProj\})$ requires that any user who *initiate a project* must have the access right to *modify the project*. We assume that any activity should be involved in only one binding constraint, otherwise we would specify all the corresponding activities into one constraint.

Example 1 Given a set of security constraints $\{c_1 = \mathsf{ME}(\{IssueCheck, InitCheck\}, 1),\ c_2 = \mathsf{AC}(audit, 1, 2)\}$, constraint c_1 limits any user taking on activity *initiate a check* should be different with the user taking on activity *issue the check*, while c_2 allows activity *audit* being assigned to at most two users.

We would like to point out that the above specification of security constraints only presents a potential way to define security constraints. In practice, constraints are defined according to application requirements and can be extended at run time. A set of security constraints may be separately defined by different people. This often results in inconsistency and uncertainty of authorization. For example, a binding constraint $\mathsf{BD}(A_s)$ may conflict with cardinality constraints when the lower-bound of some activity in A_s is larger than the upper-bound of some other activity in A_s. Assume that we have three constraints $\mathsf{BD}(\{a', a\}), \mathsf{AC}(a, 3, 5), \mathsf{AC}(a', 1, 2)$. According to $\mathsf{BD}(\{a', a\})$ and $\mathsf{AC}(a, 3, 5)$, any user who is authorized to perform a must be authorized a' and a should be performed by at least 3 users. However, according to $\mathsf{AC}(a', 1, 2)$, the number of users assigned for a' must not exceed 2. Thus, the three constraints cannot be satisfied at the same time. Therefore, it is important to guarantee the consistency of constraints.

Definition 1 (Constraints consistency) Let SC be the set of security constraints. We say that SC is *consistent* if and only if there exists a role-activity assignment relation RA and a user-role assignment relation URA such that the state $\langle R, RH, U, RA, A, URA\rangle$ satisfies all constraints in SC.

The above definition is based on the intuition that every task represented by activity must be "enforceable" in some state that satisfies the constraints in SC. Now we study how to determine whether a set of constraints is consistent.

Lemma 1 *For a given set SC of constraints, SC is always consistent if SC contains only* ME *and* AC *constraints or only* BD *constraints.*

Proof We prove this lemma in two cases:

– only ME and AC constraints: we can construct a role-activity assignment relation RA and a user-role assignment relation URA such that each activity a is assigned to a_l roles if there is a $\mathsf{AC}(a, a_l, a_h)$ constraint associated with a, and each role is assigned to one user, and there is no role hierarchy. In this case every ME constraint is satisfied and every activity is performed by a_l users.
– only BD constraints: we can construct a role-activity assignment relation RA and a user-role assignment relation URA such that for each $\mathsf{BD}(A_s)$ constraint, we assign all activities in A_s to the same role and assign each role to a user, and there is no role hierarchy. In this case every BD constraint is satisfied and every activity is performed by at least one user. □

Lemma 2 *Given a set SC of* ME *and* BD *constraints, SC is consistent if and only if for each pair of constraints* $\mathsf{ME}(A_s, k), \mathsf{BD}(A'_s) \in SC$*, the intersection set* $A_s \cap A'_s$ *contains less than k activities.*

Proof For the "if" part, suppose $A_s \cap A'_s$ contains k activities, then any state $\langle R, RH, U, RA, A, URA\rangle$ that satisfies BD constraints in SC will ensure at least k activities in A_s are taken by the same user, which violates ME constraint. Therefore SC is *inconsistent*.

For the "only if" part, suppose SC is *inconsistent*. Let $a \in A_s$ be the "unenforceable" activity. Any state $\langle R, RH, U, RA, A, URA\rangle$ which satisfies SC will have no user authorized for a. Consider a state $st = \langle R, RH, U, RA, A, URA\rangle$ which contains one user u who is assigned with role r and r is associated with a. By definition of *inconsistency*, st doesn't satisfy SC. Suppose $\mathsf{ME}(A_s, k)$ is violated. Then u must be authorized for some k activities in A_s. Since st have satisfied other constraints, u must take all activities in A'_s and these activities much overlap more than k activities with A_s. □

Lemma 3 *Given a set SC of* RC *and* BD *constraints, SC is consistent if and only if for any* $\mathsf{BD}(A_s) \in SC$ *and any pair of constraints* $\mathsf{AC}(a, a_l, a_h) \in SC$ *and* $\mathsf{AC}(a', a'_l, a'_h) \in SC$*, where* $a, a' \in A_s$*,* $a_l \le a'_u$ *holds.*

Proof For the "if" part, suppose $a'_u \le a_l$. For any system state $\langle R, RH, U, A, RA, URA\rangle$ that satisfies SC, there are a_l users authorized for a. According to $\mathsf{BD}(A_s) \wedge a, a' \in A_s$, there are at least a_l users authorized for a' also, which violates the constraint a'. Therefore SC is *inconsistent*.

For the "only if" part, suppose SC is *inconsistent*. Let a be the unenforceable activity, namely a can not be assigned for an appropriate number of users. Consider a system state $\langle R, RH, U, RA, A, URA\rangle$ in which appropriate k roles are assigned for a, where $a_l \le k \le a_u$. According to $\mathsf{BD}(A_s)$ and $a, a' \in A_s$, k users are assigned for a' also. Since $a'_u \le a_l$, $k \ge a'_u$ must hold, which makes a' unsatisfiable. □

Now we give the computational complexity analysis of security constraints verification. For Lemmas 1 and 2, since there are totally $|SC|$ constraints need verified, checking the condition in above lemmas obviously can be done in polynomial time $O(|SC|^2)$. Lemma 3 tells us determining whether the set of AC and BD conditions is consistent can be done in polynomial time by the size of role set and constraint set. For each constraint $\mathsf{BD}(A_s) \in SC$, we need to check every pair of activities $a, a' \in A_s$ that are associated with AC constraints. Thus the computational complexity of whole process is in polynomial time bounded by $O(|SC|^3)$. Overall, we have the necessary and sufficient conditions to ensure the consistency of a set of security constraints and summarize them as the following theorem.

Theorem 1 *Given a set SC of security constraints, SC is consistent if and only if for any* $\mathsf{BD}(A_s) \in SC$*, the following two conditions hold: (1) for any pair of constraints* $\mathsf{AC}(a, a_l, a_h) \in SC$ *and* $\mathsf{AC}(a', a'_l, a'_h) \in SC$*, where* $a, a' \in A_s$*,* $a_l \le a'_u$ *holds. (2) for any constraint* $\mathsf{ME}(A'_s, k)$ *the intersection set* $|A_s \cap A'_s| < k$*. Checking whether SC is consistent is in* **P**.

3.3 Business constraints and consistency

We consider three kinds of business constraints: sequential (SR), parallel (PR) and elective (ER), which are the basic requirements of business process. The semantics are given below:

Sequential relation (SR) The sequential relation $\mathsf{SR}(a_1, a_2)$ is a partial order relation between two activities a_1 and a_2 that requires a_1 being performed before a_2. We assume if the sequential relation exists between two activities, they are always regarded as different activities. a_1 is called the preceding activity of a_2. The sequential relation is asymmetric and transitive.

Parallel relation (PR) The parallel relation $\mathsf{PR}(A_s)$, where $A_s \subseteq A$, specifies that any subset of the activities in A_s should be executed in parallel when they are assembled in the same business process.

Elective relation (ER) The elective relation $\mathsf{ER}(A_s)$ requires that any subset of the activities in A_s should be executed selectively when they are assembled in the same business process.

Similar with security constraints, above definitions only give a potential way to specify the business requirements and can be extended according to practical requirements. After a set of business constraints defined, there may exist conflict. For example, two sequential constraints $\mathsf{SR}(a_1, a_2)$ and $\mathsf{SR}(a_2, a_1)$ could not be satisfied simultaneously. Hence, consistency checking is desired for business constraints.

The purpose of consistency verification is to ensure that for a set of business constraints, there exists at least one business process model that satisfies all constraints.

Lemma 4 *Given a set BC of business constraints containing* PR *and* ER *constraints, BC is always consistent if for any pair of* $\mathsf{PR}(A_s)$ *and* $\mathsf{ER}(A'_s)$*,* $|A_s \wedge A'_s| \le 1$ *holds.*

Since only one relationship either PR or ER could exist between any pair of activities, these two activities could not appear in both PR and ER constraints simultaneously. Obviously, checking this lemma can be done in polynomial time by the size of $O(|BC|)^2$.

Lemma 5 *Given a set BC of* SR *business constraints, BC is consistent if and only if for any pair of activities* a_1 *and* a_2*, only one sequential chain exists, namely either from* a_1 *to* a_2 *or from* a_2 *to* a_1*.*

Proof To prove this lemma, we first introduce the notion of activity graph for the sequential relations. □

Definition 2 (Activity graph) An activity graph $AG = (V, E)$ is a directed acyclic graph, where V is the set of nodes denoting activities and E the set of directed edges denoting the sequential relations between activities. Each sequential relation denotes an edge (a_i, a_j), where a_i is the preceding activity of a_j.

Knowing the activity graph AG, the verification of Lemma 5 becomes testing whether there exist a circle in

AG. This is done by algorithm of Fig. 2, which returns TRUE if there is no circle in AG. If any circle is found, it returns FALSE. The computational complexity upper bound of this algorithm is $O(|V|+|E|)$, which is in polynomial time. The correctness of this algorithm can be verified as follows. If the algorithm fails, the number of marked activities must be less than |V| and there is no candidate for inclusion in V. Therefore, there is at least one node a_1 that is not marked. The AG must contain an edge (a_2, a_1), where a_2 is not marked; otherwise a_1 is a candidate for inclusion. Similarly, there must exist an edge (a_3, a_2) such that a_3 is not marked. $a_1 = a_3$ indicates that a cycle is found in the direct graph. If $a_1 \neq a_3$, there must exist a_4, such that (a_4, a_3) is an edge and a_4 is not marked. Otherwise a_3 is a candidate for inclusion. If a_4 is one of a_1, a_2, or a_3, then this is an indication that the AG has a cycle. Since AG has finite nodes, repeatedly applying this checking will eventually detect a cycle if existing. A circle in AG indicates a conflict of these dependencies.

4 Legally flexible business process

As discussed above, the flexibility of business process includes both the dynamic customization of model at design and the adaptive adjustment of instance at execution. Either way is implemented by assembling activities into a business process in our approach. More important, the legitimacy should be ensured for both security and business aspects in such flexible business processes. In this section, we would discuss the semantics of assembly operations and then validate the legitimacy of the created business processes.

4.1 Assembly operations and business process

The assembly operations are defined based on a novel data structure, called business process logic net (BP-net for short).

Definition 3 (BP-net) A BP-net $\langle T, P, R\rangle$ is a restricted Petri net, where T is a set of activities; P is a set of places in form of $(type, num)$ specifying the operation *type* on the place and the number of activities the place is connected to; and R is the set of arcs between places and activities in the net. A BP-net holds the following properties:

(1) There are only two special places i and o, where i is a source place and o is a sink place, i.e., $\bullet i = \varnothing \wedge o\bullet = \varnothing$.
(2) Any other node $x \in \mathrm{P} \cup T$ is on a path from i to o. There is no dangling activities or place in a *well defined* BP-net.
(3) In the initial state, only place i holds one token, i.e., $\mathrm{M}_0(i) = 1 \wedge \forall\, \mathrm{x}\, \varepsilon\, \mathrm{P}\colon x \neq i \Rightarrow \mathrm{M}_0(x) = 0$.

These properties ensure that a *well defined* business process can be normally executed to the end after it is initialized. An important characteristic of such "good" structured business process is to balance AND/OR-split and AND/OR-join. Clearly, the parallel activities initialized by an AND-split is AND-joined, while the selective activities initialized by an OR-split is OR-joined. Based on the proposed BP-net, we could create and update a business process by assembling activities. Note that here we only consider the logical relationships among activities in order to avoid confusion between the logical relations and the semantic relations among activities (Yuan 2005). In the following definition, we would adopt p_α and $\underline{p_\alpha}$ to denote the input and output places of an activity α, and adopt $p.type$ and $p.num$ to respectively denote the connection type and the number of the subsequent activities with a place p.

Definition 4 (BP assembly operations, BP-net-op) The BP assembly operations include the Sequential insert $\diamond$, the Parallel insert $\|$, the Selective insert $|$, and the Delete operator ∂. Given a BP-net $\langle T, P, R\rangle$ and an activity $\beta \in A$, the assembly operations are defined as follow:

Fig. 2 The algorithm to verify consistency of the sequential constraints in BC

```
Establish the activity graph $AG = (V, E)$ for a set $BC$ of sequential constraints:
For any SR$(a_i, a_j) \in BS$
        $V = V \cup \{v_i, v_j\}$, $E = E \cup \{(v_i, v_j)\}$
Let STK be a stack structure and initiated empty.
Count the $in_{degree}$ of each vertex in AG
Push the vertexes with zero $\mathrm{in}_{\mathrm{degree}}$ intoSTK.
While STK is not empty, iteratively do steps a) - step c)
        a) Pop up a vertex $v_i$ from STK, mark it.
        b) For each successive vertex $v_j$ of $v_i$, minus one from its $\mathrm{in}_{\mathrm{degree}}$.
        c) If $in_{degree}(v_j) = 0$, add $v_j$ into STK.
If all the vertexes in AG are marked, return TRUS; else return FALSE.
```

- Sequential insert $\alpha \diamond \beta$, where $\alpha \in T$, indicates that β is sequentially assembled after α. The results of sequential operation are: $P = P \cup \{p_\beta\}$, $T = T \cup \{\beta\}$ and $R = R \cup \{(\alpha, p_\beta), (p_\beta, \beta)\} \cup \{(\beta, q)|q \in \underline{p_\alpha}\} - \{(\alpha, q)|q \in \underline{p_\alpha}\}$, $p_\beta.type = Sequential$ and $p_\beta.num = 1$.
- Parallel insert $\alpha \parallel \beta$, where $\alpha \in T$ with $p_\alpha.type \neq Selective$, indicates that β is assembled in parallel with α. The results of this operation are: $P = P$, $T = T \cup \{\beta\}$, $R = R \cup \{(p, \beta)|p \in p_\alpha\} \cup \{(\beta, q)|q \in \underline{p_\alpha}\}$, $p_\alpha.type = Parallel$ and $p_\alpha.num = p_\alpha.num{+}1$.
- Selective insert $\alpha \mid \beta$, where $\alpha \in T$ with $p_\alpha.type \neq Parallel$, indicates that β is selectively assembled with α. The results of this operation are: $P = P$, $T = T \cup \{\beta\}$, $R = R \cup \{(p, \beta)|p \in p_\alpha\} \cup \{(\beta, q)|q \in \underline{p_\alpha}\}$, $p_\alpha.type = Selective$ and $p_\alpha.num = p_\alpha.num + 1$.
- Delete operation $\partial(\gamma)$, where $\gamma \in T$, indicates that activity γ and its corresponding places being deleted. The results of deletion include the common part, namely $T = T - \{\gamma\}$, $R = R - \{(p, \gamma)|p \in p_\gamma\} - \{(\gamma, q)|q \in \underline{p_\gamma}\}$, and the different part, namely:
  - Case $p_\gamma.type = Sequential$: $P = P - \{p_\gamma\}$, $R = R \cup \{(\bullet p_\gamma, \underline{p_\gamma})\} - \{(\bullet p_\gamma, p_\gamma)\}$.
  - Case $p_\gamma.type = Parallel \vee Selective$: $p_\gamma.num = p_\gamma.num - 1$. If $p_\gamma.num = 1$, then $p_\gamma.type = Sequential$.

Definition 5 (Business process) A business process is a BP-net with the initial status {i, o, NULL, (i, NULL), (NULL, o)} and can be extended progressively by means of assembly operations on activities while being obliged to security and business constraints. An instance of business process is initialized at marking i and the binding of users to roles and activities are active in the process of instance execution.

From the assembly of a business process, we can see that it supports the four types of workflow routers proposed by the Workflow Management Coalition, namely sequence, parallel, selection and repetition. The assembly operations introduced above can perform the first three types. For repetition, it is solved during the workflow execution because it is related to each instance of a generated business process and governed by its semantics. We would point that the assembly operations can be enforced on both business process model and instance, which makes the created business processes flexible for both in design stage and at run time. The assembled BP-net may be not succinct. It can be predigested with the method in Yuan (2005), which is beyond the discussion in this paper.

Now, we need to proof the soundness of a generated business process, namely the assembly operations could make a customized business process *logically* correct execution. We would first define the soundness of a business process represented by the BP-net by means of the soundness of a Petri-net (van der Aalst and van Hee 2004), and then prove the proposed assembly operations maintain the BP-net sound.

Definition 6 (Soundness of a BP-net) A BP-net $= < P, T, R >$ is sound if and only if:

(i) For every marking M reachable from marking i, there exists a firing sequence leading from marking M to o. Formally, $\forall M$, $(i \xrightarrow{*} M) \Rightarrow (M \xrightarrow{*} o)$, (completion).
(ii) o is the only reachable marking from i with at least one token in place o. Formally: $\forall M (i \xrightarrow{*} M \wedge M(o) \geqslant 1) \Rightarrow (M = o)$, (proper completion), and
(iii) There is no dead activity in the business process. Formally, $\forall t \in$T, $\exists M$, M'$(i \xrightarrow{*} M \xrightarrow{t} M')$

The first property states that starting from the initial state, it is always possible to reach the state o with one token in place o, namely, the state o is reachable eventually. The second property states that in the moment that a token is put in place o, all other places must be empty. This property is also called as proper termination. The third property states that there is no dead activity in the initial state i. These three properties can ensure that a business process can eventually terminate correctly and there is only one terminate state. Moreover, there is no dead task.

Lemma 6 *Let a* BP-net $N = \langle T, P, R \rangle$ *be sound. The* $N' =< P', T', R' >$ *is always sound if it is transferred from* N *only by the sequential insert operations.*

Proof Without losing generality, let $\alpha \in T$, $\beta \in$A, and p_α and $\underline{p_\alpha}$ denote the input and output places of α, respectively. $\alpha \diamond \beta$ indicates that the sequential operation assembles activity β after α to the N. Let p_β be the input place of β. We prove the following three properties of soundness.

(i) Soundness property 1
Since N is sound, there exists a firing sequence $\sigma = t_1 t_2 \ldots t_n$ that changes the state of the N from marking $[i]$ to $[p_\alpha]$, then to $[o]$. According to the definition of the *sequential insert*, activity β has

only one input place p_β following α. After firing α, p_β is reached and β is fired. Namely, firing sequence $\sigma' = t_1t_2 \ldots t_n\alpha$ results in the marking $[p_\beta]$ and leads to the firing of β. Formally, let $\sigma = t_1t_2 \ldots t_n$, $t_i \in$T and $([i] \xrightarrow{\sigma} [p_\alpha]) \Rightarrow ([p_\alpha] \xrightarrow{*} [o])$. In *N*', $\sigma' = t_1t_2 \ldots t_n\alpha$ makes $([i] \xrightarrow{\sigma'} [p_\beta]) \Rightarrow ([p_\beta] \xrightarrow{*} [o])$.

(ii) Soundness property 2
Let M_α and M_α' be the two markings before and after firing α on *N*. Let M_β and M_β' be the two markings before and after firing β on *N*'. We only need to prove that M_β and M_β' are reachable markings from [*i*] and M_β'(*o*)=[*o*]. Because *N* is sound, there exists one firing sequence σ=$t_1t_2 \ldots t_n$ that makes $[i] \xrightarrow{\sigma} [p_\alpha]$. In *N*', given σ' =$t_1t_2 \ldots t_n\alpha$, we have $[i] \xrightarrow{\sigma'} [p_\beta]$. After firing β, we have $[i] \xrightarrow{\sigma''} [\underline{p_\alpha}]$ where σ"=$t_1t_2 \ldots t_n\alpha\beta$. In this process, no arc induces to place *o*, that is, M_β'(*o*) = [*o*]. So property 2 is proved.

(iii) Soundness property 3:
Obviously, the added activity to the *N* is not dead. □

Similarly, we could prove that a sound BP-net is always sound after the parallel insert, selective insert and delete operation, which are concluded as the following theorem.

Theorem 2 *If a* BP-net $N =< P, T, R >$ *is sound, any generated business process from N with assembly operations on activities is always sound.*

Proof This can be straightforwardly proved by the induction method.

Initial: The initial *N* is empty, denoted as $N_0 = < P, T, R >$ with P = {i, o}, T = {null}, R = {(i,null), (null,o)}. Obviously, it is sound.

Assume $N_k =< P_k, T_k, R_k >$ is sound after *k*-step assembly operations with activities on N_0. $N_{k+1} = < P_{k+1}, T_{k+1}, R_{k+1} >$ is generated by one of the four assembly operators. According to Lemma 6 and other related conclusions, N_{k+1} is sound. □

4.2 Validation of business process and execution planning

A business process being logically sound does not mean it could be correctly executed in practice. For example, there are two activities a_1 and a_2 in a business process and these exist a mutual exclusive security constraint ME$(\{a_1, a_2\}, 1)$ enforced on them, namely any user is not allowed to perform both activities. However, if only one user qualified for the attribute requirements of the roles (say full professor) performing these two activities, it is still impossible to normally execute the business process even though this process is logically sound. So, we still need to validate the legitimacy of a business process.

In this section, we would perform the validations about the security and business constraints satisfaction for a created business process. In practice, such verification should be performed both at design and at execution so as to make sure that the existence of the constraints will not prevent normal execution of a sound business process by authorized users. On one side, the business process should obey the business constraints associated with the activities in it, on the other side, for any user qualified for the roles that is authorized an activity, there exist at least one user-role-activity assignment of the business process that satisfies the constraints. In the following, we would firstly give the assembly rules that should be followed when customizing a business process so as to ensure its business legitimacy. And then discuss the validation of security constraints in a business process to guarantee the legitimacy of security

Property 1 (Assembly rule): *For a given* BP-net N= $< P, T, R >$ *and an activity* $a \in A$, *a can be assembled into N only if the following principles are satisfied:*

– *if these exist a constraint* SR$(a', a) \in BC$ *(or* SR$(a, a') \in BC$*) between two activities* $a, a' \in T$, *a must be sequentially assembled with a', namely there is a path from a' to a in the modified* N *(or a path from a to a').*
– *if these exist a constraint* PR$(a', a) \in BC$ *between two activities* $a, a' \in T$, *a must be assembled by the parallel operator with a', namely a and a' are in the same parallel structure.*
– *if these exist a constraint* ER$(a', a) \in BC$ *between two activities* $a, a' \in T$, *a must be assembled by the selective operator with a', namely a and a' are in the same selective structure.*

The algorithm of validating above property is presented in Fig. 3. The sequential constraint is verified by enumerating all anterior activities of a' in *T* and by checking whether *a* is in such set when SR$(a, a') \in BC$, or vice versa when SR$(a', a) \in BC$. The proof of the correctness of this part is similar with that in Algorithm 2 and is omitted here. For the verifications of the parallel or elective constraints, we back trace the BP-net from a' to reach the first parallel or elective place. If

Fig. 3 The algorithm to check the satisfaction of business constraints when inserting an activity to a BP-net

```
For each activity $a' \in T$
    check the business constraint set $BC$
    If $\mathsf{SR}(a', a) \in BC$
        enumerate all subsequent activities of $a'$ in $T$, denoted as set $Suc(a')$
        If $a \in Suc(a')$ then Return OK
    If $\mathsf{SR}(a, a') \in BC$
        enumerate all anterior activities of $a'$ in $T$, denoted as set $Pre(a')$
        If $a \in Pre(a')$ then Return OK
    Else if $\mathsf{PR}(a, a') \in BC$ or $\mathsf{ER}(a, a') \in BC$
        start from $a'$
        Do back trace
            reach a parallel (or elective) place
            If $a$ is on another branch of the same place then Return OK
        Until reach the sources place $i$.
Return BAD-INSERT
```

a is on another branch starting from the same place, the constraint is satisfied and return OK. Otherwise, the back track will continue and reach the next parallel or elective place. We would do the same checking. Such back track will stop at the source place i, which means this constraint is not satisfied and return BAD-INSERT. The proof of the correctness of this part is as follow. Since a *well defined* Petri-net allows the nesting between any two parallel or elective structures but no intersection, namely one structure is allowed to be a branch of another structure. Thus the above process of back trace would check parallel (or elective) structures from the inner to the outer. The constraint being satisfied in any structure is acceptable and the algorithm would return OK. The algorithm can be completed in polynomial time, which is bounded by $O(|BC| * |T|^2)$.

The satisfaction verification of security constraints would ensure that the user-role-activity assignments satisfy all the constraints, namely there exist at least one instance for a customized business process that can be executed successfully. This is called the execution planning. Since the Web users may change any time, such verification generally is performed on each instance after we have the available users who are eligible for the roles performing the activities of a business process. Suppose we have got the user role assignments by evaluating users' attributes against qualification requirements of each role for an activity, the user to activity relationships, the business process and the required security constraints are together called a configuration that is denoted $\langle T, P, R, U, UTQ\rangle$, where $\langle T, P, R\rangle$ is the BP-net denoting the business process, U is the set of available users and UTQ is the set of user-activity relationships in the form of $(u, a), u \in U, a \in T$ denoting user u is qualified for activity a. However $(u, a) \in UTQ$ does not mean u ultimately performing a since he is restricted by the security constraints. Just as the example given in the beginning of this section, activity a_2 would never be performed without violating the security constraint. Thus we should make the execution planning on a given configuration to make sure the given configuration can satisfy all the constraints.

Now we analyze the computational complexity of this process by means of the result in Wang and Li (2007). In their work, the mutual exclusive constraint is considered as a binary relationship between two activities, which can be regarded as the special case of our problem, namely all mutual exclusive constraints are in form of $\mathsf{ME}(\{a_1, a_2\}, 1), a_1, a_2 \in A$. The authors show that their problem is $\mathbf{NP} - complete$. Thus we could have the conclusion that checking whether a set of security constraints SC is satisfied by a given configuration is $\mathbf{NP} - complete$.

An intractable problem means there exist instances that take exponential time in the worst case. However, many instances that will be encountered in practice may be efficiently solvable. We would present an algorithm for the execution planning. In this process, an important thing is to lock the activity with only one eligible user for an activity on the condition of satisfying all constraints. For example, there are two constraints associated with activities a_1 and a_2, the sequential constraint $\mathsf{SR}(a_1, a_2)$ and the mutual exclusive constraint $\mathsf{ME}(a_1, a_2)$. There are two users u_1 and u_2 eligible for a_1 while only one user u_1 is eligible for a_2. In this case, if a_1 is performed by u_1 in advance, a_2 cannot be executed by u_1 again, which precludes the instance going on execution. Therefore, we should lock a_1 with u_2 and a_2 with u_1 so as to guarantee the successful execution of the whole business process.

The execution planning algorithm is shown in Fig. 4. Step 2 performs the sanitizing verification to find the impossibly satisfied cases. Step 3 verifies the BD constraints and make a binding of eligible users for the

Fig. 4 The algorithm of execution planning for a given configuration $\langle T, P, R, U, UTQ\rangle$

```
1. For each activity $a \in T$
       enumerate the eligible users of $a$ to $a.U\_set$.
2. For each AC security constraint $c \in SC$
       if $|a.U\_set| < c.a_l$ return FALSE
3. For each BD security constraint $c \in SC$
       $Inter(c) = \bigcap_{a \in c.A_s} a.U\_set$
       if $Inter(c) = \emptyset$ return FALSE;
       for each $a \in c.A_s$ let $a.U\_set = Inter(c)$
       select an activity $a* \in c.A_s$
       for each $a \in c.A_s \wedge a \in c'.A'_s$ where $c' \in SC$ is a ME constraint
           replace $a$ with $a*$
       for each $a \in c.A_s$ that also appears in an AC constraint $c'$
           let $a.a_l = max\{a'.a_l | a' \in a.U\_set\}$
4. Establish the candidate matrix $M$ for all ME constraints in $SC$
5. Return $SOD\_planning(M)$
```

activities in the same BD constraint so that the constraint can be always satisfied. Also in this step, we introduce a heuristic to reduce the AC constraints by modifying the lower bound of the AC constraints related to the same BD constraint to the maximum lower-bound since they require the same performers. Lemma 3 guarantees that such modification always reasonable. The verification of the ME and AC constraints is performed by the core sub-algorithm of execution planning, called *SOD_planning* and shown in Fig. 5. In the sub-algorithm, we introduce the notion of a candidate matrix for the set of ME constraints. Let rows be the activities involved the ME set and the columns be the eligible candidate users for them. Each entry (i, j) of the matrix is 1 if and only if user j is qualified for activity i, otherwise 0. The main idea is that we would enumerate all possible sets of user-activity assignments and check whether any of them satisfies the ME and AC constraints. To reduce the number of tried sets, we firstly lock the activities and their eligible users such that there is no other alternative user choice. Another heuristic method is to consider the precedence of the difference between eligible users with required users, say $can(i) - i.a_l$, when assigning activities to users (Please also see Fig. 5).

The practical execution time of above algorithm is highly affected by the given configuration data such as the number of activities in the business process, the security constraint set, and the number of eligible users for each activity etc. Although its computational complexity is in exponential time by $O(|SC| * |T_w|^{|U_e|})$, where $|SC|$ is the cardinality of the set of security constraints, $|T_w|$ is the the number of activities involved both in a given business process and in ME security constraints, and $|U_e|$ is the maximum number of the users eligible for the ME activities, the practical execution time is still efficient since $|U_e|$ is generally small. Some experiments were conducted to test the performance of the sub-algorithm *SOD_planning*, where the test data were randomly generated. The results are shown in Fig. 6, in which N_{SOD} denotes the average number of the elements in ME constraints. The test results

Fig. 5 The *SOD_planning* algorithm for the candidate matrix *M*

```
$satisfied = FALSE$
For each row $i$ in $M$ do
    Count the number of non-zero elements to $can(i)$
    If $can(i) < i.a_l$ return FALSE
    If $can(i) = i.a_l$ lock each non-zero entry $M(i, j)$ and row $i$
While (not $satisfied$) do
    loop: select an unlocked and unmarked row i with minimum $can(i) - i.a_l$.
        sequentially select $i.a_l$ columns for $i$ such that $M(i, j) = 1$
        mark row $i$
    until all rows are marked
    If ME constraints are satisfied under the marked assignments
        $satisfied = TRUE$
    else cancel the marks and check another set of assignments
end for
Return FALSE
```

are the average of several sets of different data. We investigated several aspects of the algorithm that might influence the efficiency. Figure 6a shows the relation of the efficiency with N_{SOD} on the fixed user number. We can see that the time of successful planning has obvious conic relation with N_{SOD} while the failure execution time is unstable although it gradually rises because the failure cases are quite different and extremes exist. Figure 6b shows that the performance time associates with the number of users when N_{SOD} is fixed. Figure 6c shows the relation between performance and the average candidate number of each element in the SoD constraints when N_{SOD} and the number of users are fixed. From Fig. 6d, we can see that the success planning time has the polynomial relation with N_{SOD} when the density of the candidates matrix is stable.

5 An illustrative example and implementation

In this section, we would present a practical example to illustrate how to use the proposed CBPM model in an application for legally flexible collaborations. We would also present the system architecture and a prototype for implementation of the model as well as the discussion about a real application in handling the property right exchange.

5.1 An illustrative example

To better illustrate how to use the proposed CBPM model for adaptive collaboration, we present a practical example in an academic institution about the flexible processes with both security and business considerations that is familiar to us. Considering the processes of different kinds of award nominations and research project approvals, they are generally promoted and defined by different offices. The participants involved in these processes may be required different qualifications in various applications and may change their roles, such as services providers or process orchestrator etc. Here are two application examples and the adopted common services(please also see Fig. 7).

Example 2 The research management office would award several project findings to some excellent young faculty or Phd students (like age less than 40). The process of examination and approval of a research project is shown as the lift side of Fig. 7. Firstly, a faculty or a PhD student with attributes of the CS filiation and age less than 40 is allowed to submit an application to the institution by invoking the activity of $<receive>submit$. Then the parallel execution of the two $<invoke>reviews$ activities are performed. After the review process is completed, the $<invoke>$

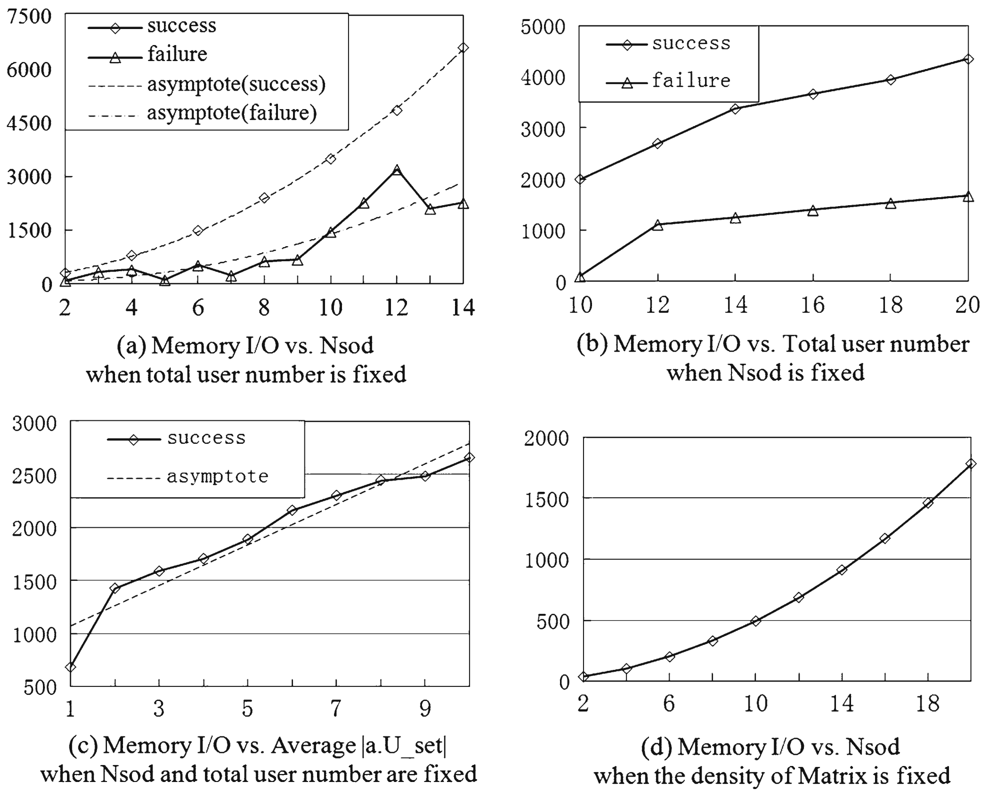


Fig. 6 Experiment results of the SOD-planning algorithm **(a–d)**

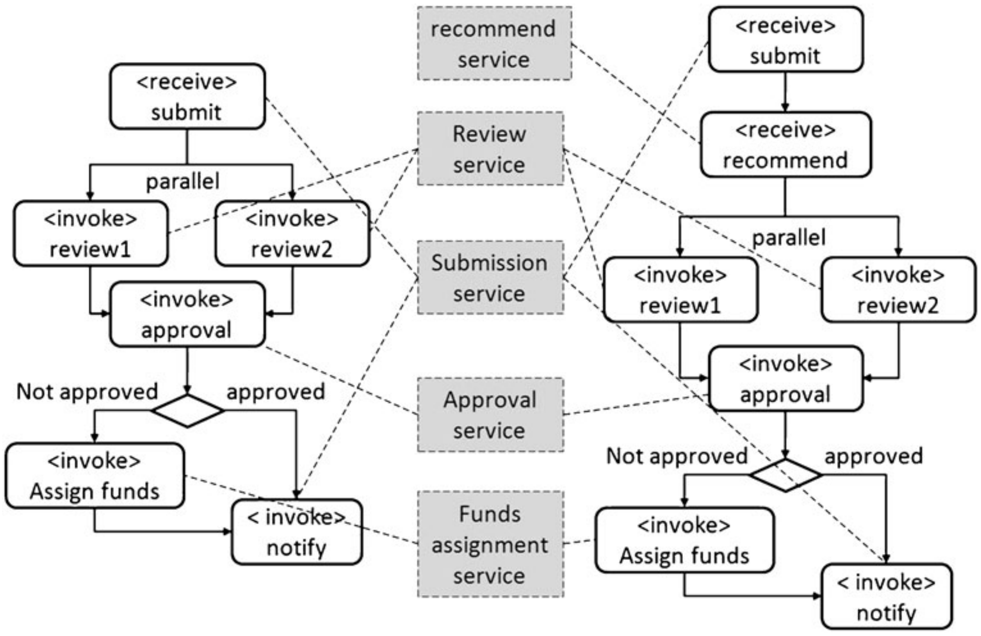


Fig. 7 Two examples of web-based business processes

approve is called. If the award is approved, the operation $< invoke > assign$ funds is performed and notification is sent back to the individual who submitted the application. There are some required security and business constraints, such the separation of duty constraint is required for two reviews. For simplicity, we denote this constraint as $ME(review_1, review_2)$, which means activities $< invoke > review_1$ and $< invoke > review_2$ should be performed by different users. Similarly, the other required constraints are the $ME(review_1, submit)$, $ME(review_2, submit)$, $ME(review_1, approval)$ and $ME(review_2, approval)$. The business constraints include the parallel execution constraint between two reviewers, denoted as $PR(review_1, review_2)$, the sequential execution constraint among activities $< receive > submit$, $< invoke > reviews$ and $< invoke > approve$, denoted as $SR(submit, reviews)$ and $SR(reviews, approve)$.

Example 3 The student affairs office would like to award the scholarship to some students with special contributions to the institute or distinct achievements in research. The process of scholarship award is shown as the right side of Fig. 7. Different with above application, the students applying for the award should have a recommendation by an associate process or full professor. Thus the activity $< receive > recommend$ is required and should be performed before invoking the activity $< invoke > reviews$. Besides the required security and business constraints mentioned in above example, the separation of duty constraint is required for the activities $< receive > submit$, $< receive > recommend$ and two reviews, which means these activities should be taken by different users. So, $ME(submit, recommend)$, the $ME(recommend, review_1)$ and $ME(recommend, review_2)$ are specified. Also, several business constraints in additional to those in above example include the sequential execution constraint among activities $< receive > submit$, $< receive > recommend$ and $< invoke > reviews$, denoted as $SR(submit, recommend)$ and $SR(recommend, review)$.

From above examples, we could see that there are many common activities between these processes, such as the submission of an application, review, approval, and assignment of funds etc., as well as many common security and business constraints, such as the sequential constraint among activities of submit, reviews and approve etc. Moreover, such processes may dynamically adjusted according to the temporary change of institution policy. And the participants of the processes may be flexible in roles, such as the service provider of each activity, orchestrator of a business process or qualified participants for activities etc. Actually, other business processes in the institution have similar characteristics with them. To support efficient management, it is desired to generate a business process with the reuse of as more as possible existing activities and constraints so as to deploy flexible business processes efficiently and economically without compromising security and business rules. These requirements are our proposed approach in this paper committed.

So the public depositories of standard activities, security and business constraints are established for the

Table 2 Standard activity set

ID	Name	Roles	Released by
C101	$< receive > submit$	Faculty, enrolled students	IT office
C102	$< invoke > review_1$	Faculty	IT office
C103	$< invoke > review_2$	Faculty	IT office
C201	$< invoke > approval$	Head, associated head	Mgt office
C301	$< invoke > assign\ funds$	Staff	FD
C401	$< invoke > recommand$	Faculty	Student office
C402	$< invoke > notify$	Staff	Student office

community i.e. the academic instruction here, which are shown in the Tables 2, 3 and 4. And the role set and role hierarchies are also built, shown as Fig. 8. Each department or office is allowed to release the standard activities and associate business and security constraints with them. When some office uses these activities for the assembly of a business process, they need to specify the qualification requirements for roles performing these activities so as to evaluate the performers (see Tables 5 and 6). For example, in the first application (i.e. example 2), the research management office could select the *assistant professor* as the role for the activity $< invoke > submit$ since it is senior than the role *faculty*. The participants are then dynamically selected to attend the process according to their qualifications.

5.2 System architecture and prototype

To implement the proposed CBPM model, we present the system architecture and illustrate it in Fig. 9. It constants three parts: Activity management, Business Process Engine and WSDL Interface.

1. In the first part *Activity management*, users can specify the activity services and associate security and business constraints with these activities. These constraints are followed during the business process assembly, adjustment and execution. Here we assume that only the charted users (like the registered users) can maintain and update the activity and constraint depositaries so as to make appropriate use of some sensitive information. We would like to point out that the activity and constraint depositaries may increase or update iteratively.
2. The part of *Business Process Engine* is the core of the architecture and is responsible for orchestrating the business processes according to different application requirements to support an active collaboration. On one side the chartered users can dynamically assemble a business process with activities, declare the role qualification requirements, make constraints legitimacy checking and perform the execution planning. On the other side, it ensures the correct execution of a business process by recording the execution history and control the access to sensitive information. The associated roles to each activity only can be active in each instance of a business process, which assures the *right* users performing *right* tasks at *right* time.
3. The third part is *WSDL Interface*, which provides an operation interface for users to make a claim of activities and constraints, or complete the execution of a human activity etc. Each operation requires users' identifications. Here we set the attributes evaluation model to evaluate the Web users' attributes against the qualification requirements of roles performing an activity.

Based on this architecture, we implement a prototype and apply it in a property rights exchange application. Property rights exchange in China covers a wide range of contents, including rights exchange of state owned assets, collectively-owned assets, limited company properties, intangible assets, and other businesses. Regarding to these business processes, flexibility and extensibility are inherent requirements in business process management because of unpredicted

Table 3 Business constraint set

ID	Type	Activities	Released by
BC001	SR	(C101, C102)	IT office
BC002	SR	(C101, C103)	IT office
BC003	PR	(C102, C103)	IT office
BC201	SR	(C102, C201)	Mgt office
BC202	SR	(C103, C201)	Mgt office
BC301	SR	(C201, C301)	FD
BC401	SR	(C401, C102)	Student office
BC402	SR	(C401, C103)	Student office
BC403	SR	(C402, C201)	Student office

Table 4 Security constraint set

ID	Type	k	Activity	Released by
SC001	ME	1	(C101, C102)	IT office
SC002	ME	1	(C101, C103)	IT office
SC003	ME	1	(C102, C103)	IT office
SC004	ME	1	(C101, C201)	Mgt office
SC401	ME	1	(C101, C401)	Student office
SC301	AC	1,1	C301	Mgt office

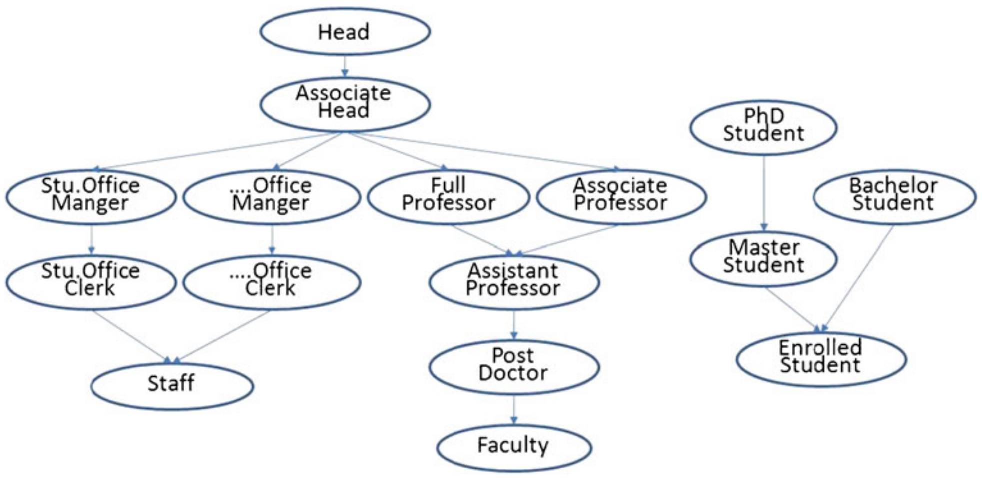


Fig. 8 Role hierarchies in the illustrative example

changes of the state policies and business development. Also, security is extremely important because a large number of users are often involved in the business processes and granted access to a huge number of secure objects to perform their different duties. In addition, the validity of legal business process is critical since each property rights exchange relates a large mount of national assets and people's profits. Any fraud or security compromise would result in huge damage. We found this to be an challenging domain for using our model and have implemented the Property Right Exchange Systems (PRES). In this PRES project, the overall objective was to develop an open, secure, flexible and extensible environment for administration and application. Figure 10 is the screen snapshot of the PRES system. Details could be got in reference (Sun and Pan 2005).

6 Related literature

The problem of secure authorizations for collaboration have attracted much attention from both academia and industry in the past decades. The Task-Based Authorization Control model (TBAC) (Thomas and Sandhu 1997) lays the foundation for active security models that are required for agent based distributed computing and workflow management. TBAC can enable permission granting, tracking and revoking to be automated and coordinated with the processing of tasks. The activity-based access control model and the task-role-based control model (T-RBAC) focus on the requirements of access control in enterprise environment and propose an improved access control model through integration of role based access control and activity based access control model (Tolone et al. 2005; Oh and Park 2003). However, from a conceptual standpoint, our consideration and method are significantly different and comprehensive. They mostly support the active authorization for a predefined workflow while we solve the problem of secure authorizations in a dynamically customized business process, which is more complex, as we state above. Also they manage the authorization by controlling the life cycle of authorization from a task-oriented perspective while we can support the reuse of constraints in different business processes beyond the restriction of access rights to a specific activity.

Our work is also related to the TMAC model, which is proposed based on the clinical workflow scenario (Thomas 1997). Two important aspects of collaborative context, user and object, were defined. The user context provides a way for identifying specific users to play a role in a team at any moment, and the object context identifies specific objects required for the collaboration purpose. As an extension, the context-based TMAC model (Georgiadis et al. 2001) integrates RBAC to include more contextual information, like time, place and so forth. Although they discuss the multidimensional

Table 5 Assembled activities in application 1 by research office

Activity	Selected roles	Qualification requirements
C101	Assistant prof., PhD	$(Assistant\ professor \wedge Age \leq 40) \vee PhD$
C102	Associate prof., full professor	$Associate\ professor \vee Full\ professor$
C103	Associate prof., full professor	$Associate\ professor \vee Full\ professor$
C201	Head, associate head	$Head \vee Associate\ head$
C301	Staff	$Faculty \wedge Department = FD$
C402	Staff	$Faculty \wedge Department = Research\ office$

Table 6 Assembled activities in application 2 by student office

Activity	Selected roles	Qualification requirements
C101	Enrolled student	Enrolled student
C102	Faculty	$(Assi.,\ asso.,\ full)professor \wedge Department = CS$
C103	Faculty	$(Assi.,\ asso.,\ full)professor \wedge Department = CS$
C401	Faculty	$(Assi.,\ asso.,\ full)professor \wedge Department = CS$
C201	Head, associate head	$Head \vee Associate\ head$
C301	Staff	$Faculty \wedge Department = FD$
C402	Staff	$Faculty \wedge Department = Student\ office$

collaborative contexts, such as organization entities and workflow tasks, neither the security constraints for flexible business processes nor the reuse of activities is supported in these models.

Some literatures investigate the specification and enforcement of authorization constraints for business process management systems. Bertino et al present a language to express authorization constraints and design algorithms to check the consistency of constraints as well as perform execution planning (Bertino et al. 1999). Wang et al. propose the role-and relation based access control (R^2BAC) model for workflow system (Wang and Li 2007). In Chaari et al. (2004), the authors propose a workflow access model capable of specifying authorization only during the execution of task. The main difference is that our approach support the full specification of the legitimacy of both security and business for flexible business processes in an open environment. Specially, we also consider the reusability of activities and constraints. Although there are quite a few literatures discussing the flexibility problem of business process, such as the framework that makes use of specially built activities that provide the functionality to define a change into an open workflow instance (Mangan and Sadiq 2002), none of them takes authorization constraints into consideration, which unavoidability lets the business processes risk various threats.

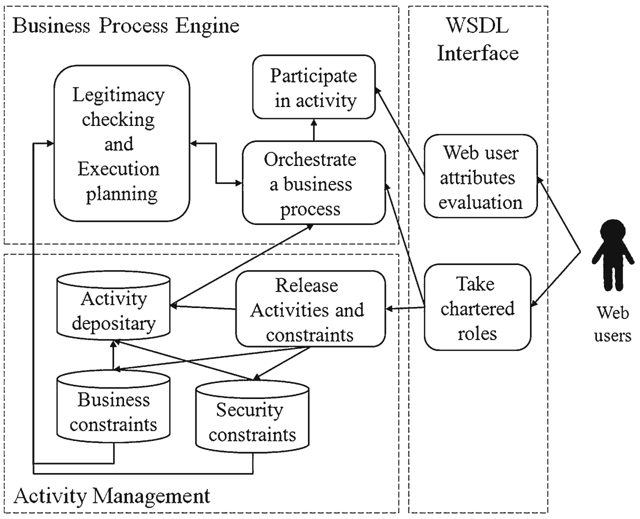
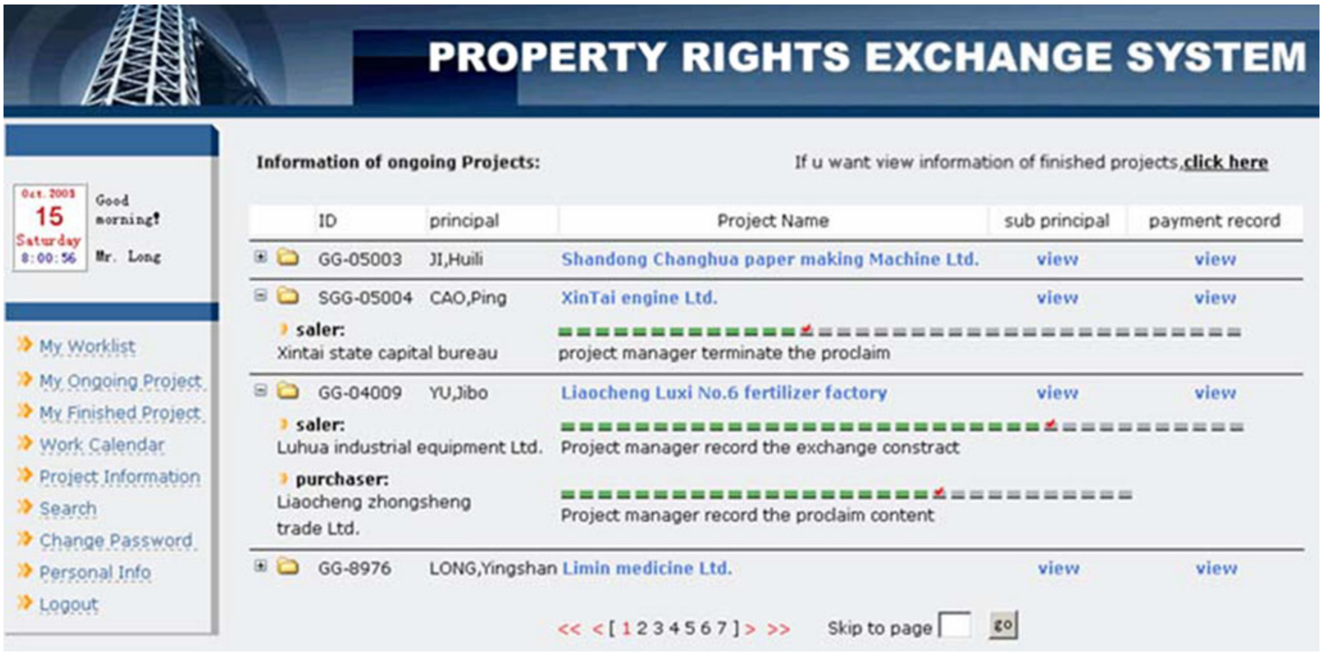


Fig. 9 The system architecture for implementation of the CBPM model

Our work is also related with the authorization management in business process based on Web services. With the widespread adoption of Web services composition to implement complex business process and of WS-BPEL as the standard language to specify business processes based on Web services, the problem of how to associate authorized users with the activities of a WS-BPEL process is gaining attention. Xiangpeng et al. propose an RBAC access control model for WS-BPEL business process (Zhang et al. 2006), in which the separation of duty constraints can be specified by using linear temporal logic (LTL) and are verified for the completion of a business process. Koshutanski and Massacci (2005a) propose an authorization model for business processes based on Web services, where the authorization logics is decoupled from the application logic of the business process. The RBAC-WS-BPEL is a role based authorization model for WS-BPEL business process (Paci et al. 2008a). It applies to WS-BPEL business processes deployed in a single organization composed of different organizational units. RBAC-WS-BPEL inherits all the components of traditional RBAC models and supports the specification of authorization constraints such as separation of duty and binding of duty that restrict the set of users that can perform a given activity. However, these models do not support the flexility of business processes such as the dynamic customization and adjustment of activities, as well as the lack of the reusability of activities that guarantee the legitimacy of both security and business for each instance.

7 Conclusions

Flexible collaboration is a notable attribute of Web 2.0, which is often in the form of multiple users participating different activities that together complete a whole business process. In this paper, we have studied the critical legitimacy problem of both security and business for flexible business process in such an active environment.



Information of ongoing Projects: If u want view information of finished projects,click here

ID	principal	Project Name	sub principal	payment record
GG-05003	JI,Huili	Shandong Changhua paper making Machine Ltd.	view	view
SGG-05004	CAO,Ping	XinTai engine Ltd.	view	view
saler: Xintai state capital bureau		project manager terminate the proclaim		
GG-04009	YU,Jibo	Liaocheng Luxi No.6 fertilizer factory	view	view
saler: Luhua industrial equipment Ltd.		Project manager record the exchange constract		
purchaser: Liaocheng zhongsheng trade Ltd.		Project manager record the proclaim content		
GG-8976	LONG,Yingshan	Limin medicine Ltd.	view	view

<< <[1234567]> >> Skip to page GO

Fig. 10 Screen snapshot of the PRES system

A novel model, namely the Constraint-based Business Processes Management Model (CBPM) is proposed. The objective is to support the legally flexible business processes while reduce the complexity of management. By introducing the assembly operations and standard activities with constraints, this model enables the secure access to a dynamically generated legal business process. At the same time, the reusability of activities improves the efficiency of management. We present a few algorithms to verify the consistency of constraints and the soundness of a generated business process. The execution planning is also conducted for a customized workflow to guarantee its correct execution on the precondition of satisfying all security constraints.

A complete practical example is presented to illustrate how to apply our model in real applications. We have presented the system architecture for implementation of the CBPM model and a prototype system for property rights exchange application at Shandong Province. The initial running of the application has demonstrated that our approach is cost-effective in multiple users access control and development of flexible legal business processes.

A future direction is the extension of this model to support the semantic authorization management for business process. In the development of electronic business, collaborative operations among different enterprises are the common requirement, where the components of access control, such as roles and activities, are locally defined and self-understanding. Thus to support transparent interoperability, the management of security and business process need to provide the system intercommunication between these organizations. Such requirement will generate new problems, such as interferences between two systems, semantic-mapping of security policies etc.

Another direction is the resiliency problem in Web based business process management. In the active Web2.0 environment, users performing business steps may be absent anytime, which will unavoidably affect the execution of a business process. So how to guarantee a business process normally executed in an open environment is challenging. We would also investigate the secure delegation problem in the flexible business process. Considering that delegation is an important business rule, we plan to introduce the fine-grained access right delegation into our model.

References

Adam, N. R., Atluri, V., Bertino, E., & Ferrari, E. (2002). A content-based authorization model for digital libraries. *IEEE Transactions on Knowledge and Data Engineering, 14*(2), 296–315.

Ahn, G.-J., & Sandhu, R. (2000). Role-based authorization constraints specification. *ACM Transaction on Information System Security, 3*(4), 207–226.

Bertino, E., Ferrari, E., & Atluri, V. (1999). The specification and enforcement of authorization constraints in workflow management systems. *ACM Transactions on Information System Security, 2*(1), 65–104.

Chaari, S., Biennier, F., Ben Amar, C., & Favrel, J. (2004). An authorization and access control model for workflow. In *Proceedings of the 1st international workshop on computer supported activity coordination, Porto, Portugal* (pp. 21–30).

Clark, D. D., & Wilson, D. R. (1987). A comparision of commercial and military computer security policies. In *Proceedings of the 1987 IEEE symposium on security and privacy* (pp. 184–194). Silver Spring: IEEE Computer Society Press.

Georgiadis, C. K., Mavridis, I., Pangalos, G., & Thomas, R. K. (2001). Flexible team-based access control using context. In *Proceeding of ACM symposium on accesss control models and technoloy* (pp. 21–27). Chantilly, VA.

Gordon, L. A., & Loeb, M. P. (2006). Economic aspects of information security: An emerging field of research. *Information Systems Frontiers, 8*(5), 335–337.

Koshutanski, H., & Massacci, F. (2005a). Interactive credential negotiation for stateful business processes. In *Proceedings of 3rd international conference on trust management (iTrust 2005), LNCS* (Vol. 3477, pp. 256–272). Rocquencourt: Springer.

Koshutanski, H., & Massacci, F. (2005b). An access control framework for business processes for web services. In *Proceedings of ACM workshop on xml security, Fairfax VA, USA* (pp. 15–24).

Li, N., Tripunitara, M. V., & Wang, Q. (2006). Resiliency policies in access control. In *Proc. ACM conference on computer and communications security* (pp. 113–123).

Mangan, P. J., & Sadiq, S. (2002). A constraints specification approach to building flexible workflows. *Journal of Research and Practice in Information Technology, 35*(1), 21–39.

Oh, S., & Park, S. (2003). Task-role-based access control model. *Journal of Information System, 28*, 533–562.

Paci, F., Bertino, E., & Crampton, J. (2008a). An access-control framework for WS-BPEL. *International Journal of Web Service Research 5*(3), 20–43.

Paci, F., Ferrini, R., Sun, Y. Q., & Bertino, E. (2008b). Authorization and user failure resiliency for WS-BPEL business processes. In *Proceeding of the 6th international conference on service oriented computing, University of Technology, Sydney, Ultimo City* (pp. 116–131).

Sandhu, R., Coyne, E., Feinstein, H., & Youman, C. (1996). Rose-based access control model. *IEEE Computer, 29*(2), 38–47.

Sun, Y. Q., & Pan, P. (2005). PRES-A practical flexible RBAC workflow system. In *Proceedings of the 7th international conference on electronic commerce, Xi'an, China* (pp. 653–658).

Thomas, R. (1997). Team-based access control. In *Proceeding of 2nd ACM workshop on role-based access control, Fairfax VA* (pp. 13–19).

Thomas, R. K., & Sandhu, R. (1997). Task-based authorization controls (TBAC): A family of models for active and enterprise-oriented autorization management. In *Proceedings of the IFIP 11th international conference on database securty XI* (pp. 166–181).

Tolone, W., Ahn, G., Pai, T., & Hong, S. P. (2005). Access control in collaberative systems. *ACM Computing Survey 37*(1), 29–41.

van der Aalst, W. M. P. (1996). Three good reasons for using petri net-based workflow management system. In *Proceedings of the international workflow conference on information and process integration in enterprises (IPIC'96)* (pp. 179–201).

van der Aalst, W. M. P. & Berens, P. J. S. (2001). Beyond workflow management: Product-driven case handing. In *Proceeding of ACM conference on supporting group work, Boulder, Colorado* (pp. 42–51).

van der Aalst, W. M. P. & Jablonski, S. (Eds.) (2000). Flexible workflow technology driving the networked economy. *International Journal of Computer Systems, Science, and Engineering, 15*(5, special issue).

van der Aalst, W., & van Hee, K. (2004). *Workflow management models, methods, and systems*. Cambridge: MIT.

Wang, Q., & Li, N. (2007). Satisfiability and resiliency in workflow systems. In *Proc. European symp. on research in computer security (ESORICS)* (pp. 90–105).

Yu, X., et al. (2007). A model-driven development framework for enterprise web services. *Information Systems Frontiers, 9*(4), 391–409.

Yuan, Z. (2005). *The theory and apllication of petri-net*. Beijing: Electronic Industry Publishing Company. ISBN 7-121-00970-6.

Zhang, X. P., Cerone, A., & Krishnan, P. (2006). Verifying BPEL workflows under authorisation constraints. In *Proceedings of fourth international conference on business process management (BPM 2006)*. Vienna, Austria.

Zisman, M. D. (1977). *Representation, specification and automation of office procedures*. PhD theses. Philadelphia: University of Pennsylvania Wharton School of Business.

Yuqing Sun received her BSc, Master and PhD degrees in Computer Science from Shandong University, China. She is currently an associate professor in the School of Computer Science and Technology at Shandong University. She has been a visiting scholar at Hongkong University in Hongkong and at Purdue University in West Lafayette. Her research interests include access control model and technology, security policy, Web services, and workflow management. She has published more than thirty papers in refereed journals and in international conferences and symposia proceedings. She also reviews for international journals and serves on the program committees of many international conferences.

Joshua Zhexue Huang is the Assistant Director of E-Business Technology Institute at the University of Hong Kong. He received his Ph.D. degree from the Royal Institute of Technology in Sweden. Before joining ETI in 2000, he was a senior consultant at the Management Information Principles, Australia, consulting on data mining and business intelligence systems. From 1994 to 1998 he was a research scientist at CSIRO Australia. His research interests include data mining, machine learning, clustering algorithms, and Grid computing.

Xiangxu Meng received the PhD degree in computer science from Institute of Computing Technology, Chinese Academy of Science in 1998. He is a professor at the Shandong University, China. He has been active in the areas of CSCW, CAD, Human-Computer Interaction, Virtual Reality, and Grid Computing. He has published more than 100 research papers in refereed international journals and conference proceedings. Now he is a commissioner of information department of science and technology committee of Ministry of Education, associate director of CAD/CG Committee of Chinese Computer Association.