

# A United Access Control Model for Systems in Collaborative Commerce

Han Ruo-Fei

Information and Electric College, Naval University of Engineering, Wuhan, China  
Email: hrf\_402@sina.com

Wang Hou-Xiang, Xiao Qian, Jing Xiao-Pei, and Li Hui

Information and Electric College, Naval University of Engineering, Wuhan, China  
Email: wanghx@public.wh.hb.cn

**Abstract**—The latest researches on access control model are dramatically different from conventional ones. Nowadays, most attention is paid to accessing across organizational boundaries. So, how to identify the applicant and determine authorization with limited information; how to express and exchange control rules expediently; how to protect confidential information and enhance collaboration simultaneously, are the most concerned problems. However, for large commercial organizations, a fine management of internal functions is of the same importance as external service management. It is very troublesome to control authorizations merely with attributes and composition of policies introduced from attribute-based access control (ABAC). So, we introduce a united access control model for systems in collaborative commerce, combining the advantages of conventional role-based access control (RBAC), task-based authentication control (TBAC) and that of recent ABAC and automated trust negotiation (ATN). Innovational ideas in the model are analyzed, and the implement architecture is discussed. The paper concludes with a summary of the united model's benefits and future work.

**Index Terms**—access control, collaborative commerce, service-oriented architecture, collaboration, negotiation

## I. INTRODUCTION

New-style network techniques always come together with new-style information system and new forms of commercial collaboration to optimize collaboration within and among organizations. Collaborative commerce, which evolves from collaboration in the workflow to concurrent engineering, supply chain, and beyond, is the latest form of commercial cooperation [1]. The level of collaboration has moved beyond mere buying and selling to encompass planning, design, development, communication, the sourcing of information, research, and the provision of services among organizations. The idea is partly derived from and based on techniques of service-oriented architecture (SOA). It considers information systems as being made up with autonomous,

dynamic, loosely coupled and service-based components [2]. Services of different information systems can also be easily and flexibly assembled to provide high level service. So the requirement of collaborative commerce can be well fulfilled with services, which can be easily tailored and composed.

There have been many mature techniques and standards to implement SOA. Take the Web Service for example, it use UDDI for service directory, WSDL for interface definition, and SOAP for invocation, and all of which use XML as communications format. [3] These standards remove many dependencies on application servers, operating systems, and machine architecture, making composition of independently developed components far easier. However, there are also many problems emerged in SOA, especially security problems. Every organization is responsible to ensure the security of its services provided to others, and to keep its own confidential information from leaking out. For every organization in collaborative commerce, there should be a balance on cooperation and autonomy. The conflict between internal and external access control policies should be paid attention too.

In order to ensure the security of SOA, many standards have been proposed to provide a means for describing and exchanging security policies among different organizations, such as extensible access control markup language (XACML) [4]. They can be used for service requester and provider to achieve a security agreement. But the security policies for requests from inside and outside the organization should not be the same, and the external request should be translated into internal request for implementation. So, only XACML is not sufficient to model access control in commercial organizations.

Former access control models, such as role-based access control (RBAC) [5] and task-based authentication control (TBAC) [6], are proposed for distributed systems, the subjects involved in the system must be identifiable with these models, which can hardly satisfy the web requirement. However the structured models do well in managing privileges of subjects within organization. Attribute-based access control (ABAC) models subjects,

resources and environment with attributes [7], so the identity of subject needs not to be validated, and the web requirements can be satisfied. However, the attribute management cross domains would be much difficult. Trust and reputation-based access control proposed trust management to validate and determine authorization among unacquainted domains [8]. But the updating and achieving of trust and reputation information is troublesome. Automated trust negotiation (ATN) is another access control approach, which considers trust information in authorization control [9], but the cost of trust negotiation before authorization is very high.

In this paper, we will analyze the existing access control models and propose a united access control model, adopting some useable ideas into it. The proposed model is aimed to control privileges of internal functions and integrate them into service, establish rules for external invoking. The rest of the paper is organized as follows:

Section 2 briefly reviews the existing access control models. Section 3 proposes the united access control model with detailed introduction of ideas involved in internal and external control mechanism. Section 4 simply analyzes the characteristics of the innovational ideas in the model. Section 5 discusses the implement architecture. Section 6 concludes the paper.

## II. EXISTING ACCESS CONTROL MODELS

From different point of view, the access control models can be classified in different ways. From control manner, they can be classified into discretionary access control (DAC) and mandatory access control (MAC); from control activity they can be classified into passive control and active control, as RBAC and TBAC. In this section, we will discuss the existing access control models from the point of their applicable situation, divided into local distributed system and large scale network system. Conventional models RBAC and TBAC together with the newer models ABAC and ATN are discussed in it.

### A. Models for local distributed system

For local distributed system, the amount of legal users involved in it is finite, and their description information is sufficient. So the access control used in it is mainly identity based. The major point emphasized in access control system is a well developed model to restrict the access approach, so that the misuse to resources by legal users can be prevented.

There are two famous principles proposed to verify ability of access control model in this environment [5]:

- Least privilege. The privileges authorized to users must be just adequate for them to implement the function.
- Separation of duty (SOD). Operational permissions to sensitive resources must be dispensed to different users separately.

#### 1) Role based access control

The development of RBAC is extremely supported by National Institute of Standard and Technology (NIST). Up to now, there are a series of mature RBAC models,

which are still most widely applied in various environments and investigated by researchers.

The basic RBAC model groups users of different posts and functions with role, which is the most concerned characteristic in permission dividing. Then the trivial user-permission assignment can be divided into user-role assignment and role-permission assignment, which extremely facilitates the management to permission. Advanced RBAC models add role hierarchy (RH), inheritance with RH, and constraints to basic model, so that the expressive ability of role relationship and restriction is improved [10].

Though flexible and comprehensible, RBAC can not well support the principle of least privilege because of its passive characteristic. Roles are assigned to users previously before they are activated into function. So once the user login system, the role will be activated and he will hold the corresponding permissions during the whole lifecycle of session, no matter he need it or not.

#### 2) Task based access control

TBAC is proposed to meet the requirement of workflow management system (WFMS). The idea is first introduced to improve RBAC model, it divide the permissions assigned to role into operation and object. Then, operation is separated solely and managed as task in TBAC.

Actually, task is just a logical concept in TBAC. The more structural elements include authorization step (AS) and authorization unit (AU) [11]. AS is the smallest access control unit, which records the executor and the enabled permission. AU is composed by one or more ASs, it is produced dynamically according to workflow sequence predefined and sent to users. Dependency is defined among different AUs. Task can be AS or AU.

The most obvious advantage of TBAC is active control and dynamic authorization, so that the permission is authorized to user only when it is immediately needed. Together with the definition of dependency, both the two principles can be well fulfilled. However, the high dependence on WFMS and ignorance of organization management badly confined its application. It usually has to combine with RBAC to work, that is called task and role based access control (T-RBAC) [12].

### B. Models for large scale network system

When it comes to large scale network system, the amount of users who access it increases exponentially. By the way, their description information is very insufficient, so the conventional identity based access control models can hardly satisfy the requirement. Another challenge is the discordance of different access control models and conflicts of self-concerned policies among different organization, which makes the agreement in collaboration hard to achieve.

#### 1) Attribute based access control

Since that the full identity information is hard to achieve, it is sensible to make authentication merely with several concerned attributes of users. That is what ABAC

model does. It uses attribute certificates that are issued by authorities to prove the ability or reliability of users.

ABAC simply uses attribute to model the characteristics of involved entities, and then formulates complex policies to describe attribute restrictions semantically [7]. Subject, resource and environment are all described with attribute, so they can easily managed by system with policies.

The most outstanding advantage of ABAC is its great expressive ability. With attribute, it can describe entities from several different points of view; even role can be defined as an attribute of subject. With policy, it also can express the service requirement flexibly. However, the attributes are too subtle and discrete; no relationship among attributes or element value within single attribute is defined. It will increase the expenditure on attribute management. By the way, all restrictions expressed with policy will increase the complexity of analytical arithmetic. At last, the implementation of ABAC depends much on authoritative authorities. Attributes defined internal organization can hardly be recognized by others.

2) Automated trust negotiation

ATN is a policy negotiation mechanism established at the base of recent policy based access control, as ABAC, to resolve conflict of information sharing and sensitive information protection.

Components involved in ATN include policy description language, as XACML, and a negotiation policy, which defines what policy or attribute certificate, and when it can be exposed in the negotiation process. In order to improve the efficiency and successful rate of negotiation, reduce negotiation cost, many researches have been done in this area [13]. We will also define a negotiation mechanism in the united access control model.

III. UNITED ACCESS CONTROL MODEL FOR SYSTEM IN COLLABORATIVE COMMERCE

As mentioned before, the access control requirement in collaborative commerce include privilege management inside the commercial organization and access restriction for external request. Since the external request can only be implemented though internal functions, it is important to model organizational management sensibly, so that

internal functions can be integrated flexibly. In this section, we will propose a united access control model to control internal authorization and external accessing ports. The proposed model absorbs many useable ideas from existing access control models. A concise illustration of the united model is shown in Fig. 1. The related concepts are mainly defined as below.

- Subject. Internal users or systems which can access resources in the organization.
- Object. Internal resources that can be accessed by internal and external subjects.
- Attribute. Characteristics of subjects, objects, and environment which is issued by the organization or other authorities.
- Role. Entities that are set to describe different post internal organization and authorized external user group with different services.
- Task. Entities that are set to describe functions internal organization.
- Service. Remote functions that are enveloped with task for internal invoking or implemented with internal tasks and released for external invoking.
- Policies. Attribute constraints on role to restrict subject, object, and environment in the accessing process.
- Constraints. Mutual exclusive relationship among roles or tasks, including static and dynamic exclusions.
- Contract. Entities that are set to describe the collaborative relationship among internal role, external role and service as authorization certificates.

The link lines with different ends shown in Fig. 1 mean differently. Link line with two small ends means the corresponding two entities have a one-to-one relationship; link line with two big ends means the corresponding two entities have a many-to-many relationship; link line with a small end and a big end respectively means the corresponding two entities have a one-to-many relationship.

Detailed designs for the united model are discussed as follows.

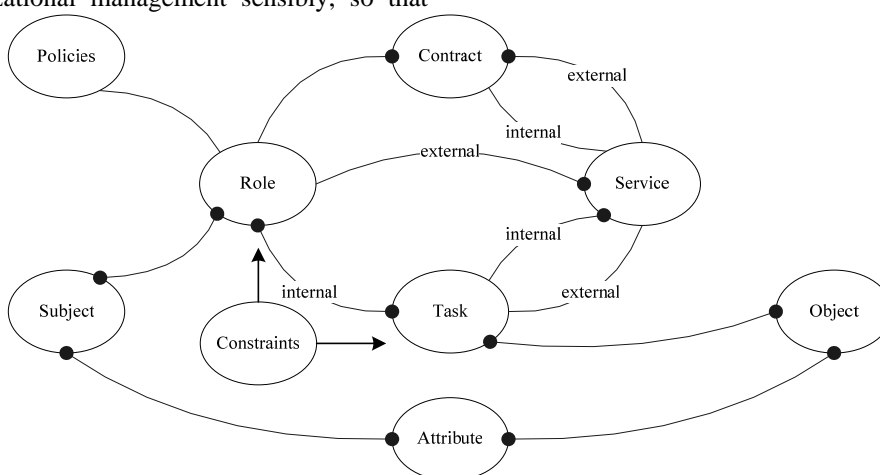


Figure 1. The united access control model

A. Mechanism to control internal access request

The internal access control component is constructed at the base of T-RBAC model, which is proposed for large scale workflow system. Since the collaborative commerce is an advanced form evolving from workflow, it is sensible to reexamine the model and improve its applicability.

The elements involved in internal component include S, O, SA, OA, InR, and T, standing for subject, object, subject attribute, object attribute, internal role, and task, respectively. The corresponding relationships among the elements include SAA, OAA, SRA, OTA, RTA, RH, and TI. They are formally defined as follows:

- Subject attribute assignment (SAA) is a many-to-many relationship; the attribute set assigned to subject  $s$  is noted as  $SA(s)$ :

$$SA(s) = \{(AttributeName_i : AttributeValue_i) | i \in N\}$$

- Object attribute assignment (OAA) is a many-to-many relationship; the attribute set assigned to object  $o$  is noted as  $OA(o)$ :

$$OA(o) = \{(AttributeName_i : AttributeValue_i) | i \in N\}$$

- Subject and internal role assignment (SRA) is a many-to-many relationship; the role set assigned to subject  $s$  is noted as  $SR(s)$ :  $SR(s) \subseteq InR^n$
- Object task assignment (OTA) is a many-to-many relationship; the task set assigned to object  $o$  is noted as  $OT(o)$ :  $OT(o) \subseteq T^n$
- Internal role task assignment (RTA) is a many-to-many relationship; the task set assigned to role  $r$  is noted as  $RT(r)$ :  $RT(r) \subseteq T^n$
- Role hierarchy (RH) is a hierarchical relationship among roles; if role  $r_1$  is higher than role  $r_2$  in role hierarchy, it is noted as  $r_1 \geq r_2$ .
- Task inclusion (TI) is an inclusion relationship among tasks; if task  $A$  is a subtask of task  $B$ , it is defined as  $B$  includes  $A$ , noted as  $B \supset A$ .

Except the relationships defined above, there are also many essential ideas in the model are discussed below.

1) Collaboration-oriented inheritance with role hierarchy

Inheritance is first brought in by RBAC together with role hierarchy. But the traditional inheritance is ability oriented; senior role would inherit all the permissions, or abilities, from junior role. However, there are some practical problems in application with this kind of inheritance.

a) *It violates the principle of least privilege.* Although there does exist inclusion of permissions among roles in practical use, the most relationship among roles in collaborative organization is separation of responsibility in working process and collaboration. For example in producing department, producing manager is a senior role and producing operator is a junior role. Both of the two roles can read producing order, but the primary duty of

manager is to design and dispense the produce plan; operator is due to execute the process according to the assigned plan. With complete inheritance, permissions, which are unnecessary for them, will be inherited up to senior roles. It will burden the role and increase the risk of operation misuse.

b) *It conflicts with the principle of SOD.* According to SOD, there inevitably be some exclusive permissions and exclusive roles in access control system. However, with complete inheritance, senior roles inherit from exclusive junior roles will be exclusive roles too. So, there can not be a senior role inherit from several exclusive junior roles. It does not accord with the fact and increases the complexity of maintaining the role and permission assignment.

With an in-depth analysis to the relationship expressed in organization structure, we find that the relationship among roles should not be the inclusion of permission sets described in ability oriented inheritance, but a commander-operator relationship. Different roles cooperate to accomplish the accessing process. The responsibility of senior roles is to dispense and supervise the work that junior roles do. From some point, if role  $A$  can do all the tasks that role  $B$  can, role  $B$  is unnecessary to exist.

In order to distinguish different duties of different roles to task, we define three kinds of operational permission of task from the point of task lifecycle [14], as shown in Fig. 2.

Definition 1 (operational permission of task)

- Executing permission. The permission for users to practically execute the task.
- Supervising permission. The permission for users to initiate, approve, dispense, administrate task execution and dispose the result.
- Invoking permission. The permission for users to initiate task request and acquire its result, it is a part of supervising permission.

Correspondingly, there are three kinds of task sets assigned to each role, executing role set (ET), supervising role set (ST), and invoking role set (IT), so that:

$$RT(r) = ET(r) \cup ST(r) \cup IT(r) \tag{1}$$

Definition 2 (assignment rule of operational permission)

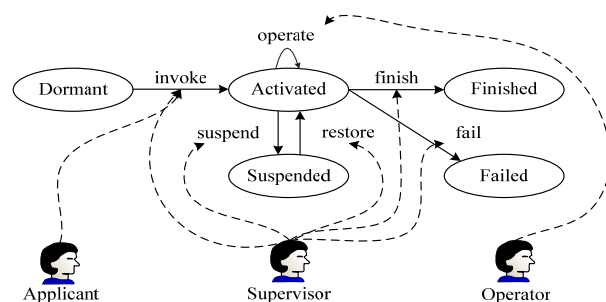


Figure 2. Separation of duty in task lifecycle

- If a task is defined, there must be at least one role which the executing permission is assigned to, so does the supervising permission.
- If the supervising permission of a task is assigned to a role, there must be at least one role which the executing permission is assigned to; the latter can be the same role, or a junior one with role hierarchy to the former.
- If the executing permission of a task is assigned to a role, there must be at least one role which the supervising permission is assigned to; the latter can be the same role, or a senior one with role hierarchy to the former.
- If there are two mutually exclusive tasks, their executing permissions can not be assigned to a same role, but there is no constraint on supervising permission and invoking permission.

*Definition 3 (collaboration-oriented inheritance with role hierarchy)*

- Sequence of inheritance. The transition of operational permission of task is divided into four steps as “executing permission → executing permission and supervising permission → supervising permission → invoking permission” along the bottom-up sequence with role hierarchy.
- Range of inheritance. Number  $n$  of hierarchies involved in every step of inheritance,  $n$  is integer belonging to  $[0, \infty]$ . If  $n=0$ , the step is overleaped, the next step come to work directly; if  $n=\infty$ , the step continues to work until the top of role domain.
- Constraints of inheritance.

a) Only the starting role of the first step and the range of every step need to be set in task assignment. And the assignment must follow Definition 3 (assignment rule of operational permission of task);

b) The summation of ranges must not exceed the number of hierarchies from starting role to the top role. If there is some range of step  $n=\infty$ , the summation of ranges from the first step to the step before must less than the number of hierarchies from starting role to the top role and the range of following steps must be 0;

c) If there are several roles with different hierarchy who use the same operational permission at the same time, operations of senior role will overlap that of the junior ones.

The transition of operational permission with collaboration-oriented inheritance is shown in Fig. 3.

Collaboration is much highlighted in collaboration-oriented inheritance. With the definition of operational permission, most tasks defined in the united model can not be accomplished by a single user alone. The responsibility of different users in the same task is much more definite than former T-RBAC model.

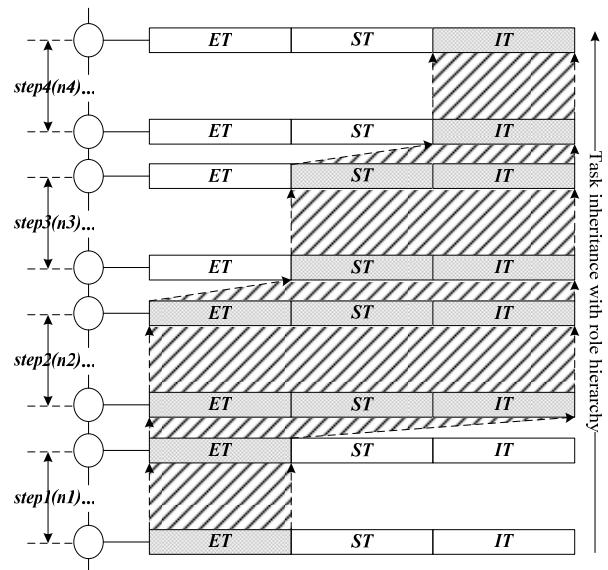


Figure 3. Collaboration-oriented inheritance

2) Task inclusion and combination

Task is introduced from TBAC; it is dynamic authorization unit to operation. However, the subtle operations are only compositional parts of high level functions. In the united model, we propose atomic task and combination task to describe basic operation and compositional function, both of which are assigned to role as task.

In conventional TBAC, task sequence is formulized into workflow, which can not be assigned to role. So the responsibilities distributed in workflow will be unclear. Secondly, the requirement of flexibility in function composition is ever increasing to realize loose coupled service. With the combination task and operational permission, we can easily define responsibilities of collaborative users participating in collaboration task. In the following section, we will propose task combining language to express the task sequence, so users in collaborative commerce can define their own combination task flexibly.

*Definition 4 (task combining language)*

- Symbol of task. ‘ $T$ ’ together with task ID defined in access control system.
- Symbol of task interruption. ‘ $I$ ’ together with interruption ID defined in task combining sentence. It is used to mark the point that working process switches to or switches out.
- Symbol of task sequence.

a) *Serial task flow.* Tasks, included in combination task,  $A$  and  $B$  are executed as a serial flow, noted as  $A > B$ .

b) *Parallel task flow.* Tasks, included in combination task,  $A$  and  $B$  are executed simultaneously, noted as  $\{A, B\}$ . If there are more than two tasks in parallel flow, it can be noted like  $\{A, B, C\}$ .

c) *Diffluent task flow.* Tasks, included in combination task,  $A$  and  $B$  are executed

selectively according to executing result of task foregoing, noted as  $[A, B]$ . If the foregoing task is accomplished successfully, then A starts; if the foregoing task failed, then B starts.

d) *Circular task flow*. Tasks, included in combination task, are executed as a circular flow. It is realized with task interruption  $I$  and diffluent task flow, noted as  $I > A > [B, I]$ . The first appearance of task interruption in task combining sentence is defined as point to switch to; the following appearance of task interruptions in diffluent task flow is defined as point to switch out.

- Task combination. A comparatively integrated task sequence, expressed with a piece of task combining sentence, marked with ‘(’ in the beginning and ‘)’ in the end.
- Task combining sentence. Sentence used to describe a piece of working process with task combining language.

The task combining language is defined formally with context-free grammar as follows.

- (1) **Terminals:**  $T, I, >, \{, \}, [, ], ', (, ), 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.$
- (2) **Non-terminals:**  $\langle TCS \rangle, \langle TC \rangle, \langle PT \rangle, \langle ID \rangle.$
- (3) **Start symbol:**  $\langle TCS \rangle.$
- (4) **Context-free productions:**  
 $\langle TCS \rangle ::= T \langle ID \rangle \mid \langle TC \rangle \mid \langle TCS \rangle \rangle \langle TCS \rangle$   
 $\langle TC \rangle ::= (\langle TCS \rangle) \mid \{ \langle PT \rangle \} \mid \langle TCS \rangle \rangle [ \langle TCS \rangle ,$   
 $\langle TCS \rangle ] \mid I \langle ID \rangle \rangle \langle TCS \rangle \rangle [ \langle TCS \rangle , I \langle ID \rangle ]$   
 $\langle PT \rangle ::= \langle TCS \rangle , \langle TCS \rangle \mid \langle TCS \rangle \rangle , \langle PT \rangle$   
 $\langle ID \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \mid$   
 $\langle ID \rangle \langle ID \rangle$

In the following, we will give an example of working process shown in Fig. 4.

In the example, when the combination task starts, tasks  $T1, T3$  and  $T4$  will be activated simultaneously.  $T2$  will be activated right after  $T1$  is accomplished. After  $T2, T3$  and  $T4$  are all accomplished,  $T5$  and  $T9$  will be activated.  $T6$  and  $T7$  will be activated selectively according to result of  $T5$ , and then  $T8$  starts.  $T10$  starts after  $T9$  finished. If  $T10$  is accomplished successfully,  $T11$  starts; if not, it would be restarted. After  $T8$  and  $T11$  finished,  $T12$  will be activated. When  $T12$  finishes, the whole combination task is accomplished.

The corresponding task combining sentence is that

$$\{(T1 > T2), T3, T4\} > \{(T5 > [T6, T7] > T8), (T9 > I1 > T10 > [T11, I1])\} > T12$$

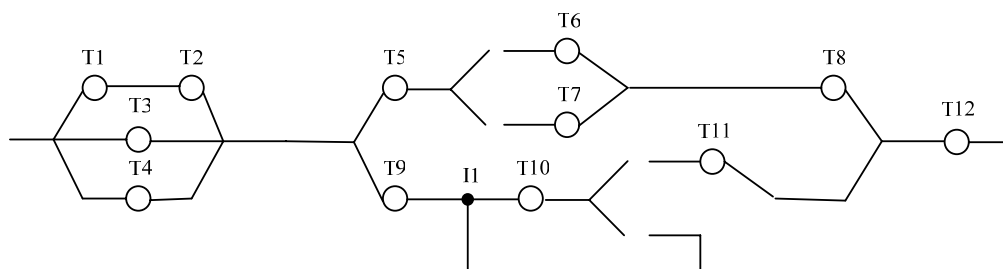


Figure 4. Example of working process

While, the composed task combining sentences must be verified and follow the constraints defined in task dependency.

The combination tasks are classified into predefined ones and user-defined ones. Only atomic task and predefined combination task follow the task template management, and the corresponding task combining sentence will be recorded in task template. The user-defined combination task must be verified before authorization and erased after execution without storing. While, the verified user-defined combination task can be transformed to predefined ones, and task template will be generated and managed by system.

Then we define the operational permission of combination task and confine the ability of user to launch user-defined combination task.

*Definition 5 (separation of combination task duty)*

- The necessary condition for assigning supervising permission of combination task to a role is that:

a) *If there are roles with all the supervising permissions of subtasks, the supervising permission of the combination task can only be assigned to those roles;*

b) *If there is not a role with all the supervising permissions of subtasks, the supervising permission of the combination task can only be assigned to roles with all the invoking permissions of subtasks.*

- The necessary condition for role to initiate a user-defined combination task is that the role must have all the invoking permissions of subtasks. Then the system will search for a role with all the supervising permissions, and send the supervising permission of the user-defined task to him. If there is not that role, then the system will send the supervising permission to the initiating role.

### 3) Constraints

The element is introduced from RBAC and TBAC. In the united model, constraints are restrictions on mutual elements of the same entity, such as ER and TD. They are formally defined as follows:

- Exclusive roles (ER) are recorded with a list of role pairs. It is used to express SOD in post assignment.

According to classification of SOD, ER can be classified into dynamic ER and static ER. So it is described with a three-element set as:

$$[role\ name_1, role\ name_2, exclusive\ type\ (D\ or\ S)]$$

For example, the dynamic exclusive role set of role  $r_1$  is noted as  $ER(r_1, D): ER(r_1, D) \subseteq R^n$ .

- Task dependency (TD) is recorded with a list of task pairs. It is a much subtle expression of SOD.

According to TBAC, TD can be classified into order dependency, defeat dependency, exclusive dependency. So the TD is also described with a three-element set as:

$[task\ name_1, task\ name_2, dependency\ type\ (O, F, or\ E)]$

The order dependent task set of task  $t_1$  is noted as  $TD(t_1, O): TD(t_1, O) \subseteq T^n$ .

The constraints are defined to restrict role and task in their assignment, activation and execution. Other constraints as time-based control can also be absorbed into it according application environment.

**B. Mechanism to control external access request**

The external access control component is constructed at the base of ABAC model, which is widely investigated in recent research. Many techniques and standards have been established to implement the model. In the following section, we will pay more attention to the approaches translating external request into internal functions.

The elements involved in external component include InR, ExR, InSe, ExSe, T, P, and C, standing for internal role, external role, internal service, external service, task, policy, and contract, respectively. The corresponding relationships among the elements include InSeTA, ExSeTA, ExRSeA, InSeCA, ExSeCA, RCA, RPA, and RH. They, all but the last, are formally defined as follows:

- Internal service and task assignment (InSeTA) is a many-to-one relationship; the internal service set assigned to task  $t$  is noted as  $InSeT(t): InSeT(t) \subseteq InSe^n$
- External service and task assignment (ExSeTA) is a one-to-one relationship; the task assigned to external service  $se$  is noted as  $ExSeT(se)$ .
- External role and external service assignment (ExRSeA) is a one-to-many relationship; the external service set assigned to role  $r$  is noted as  $ExRSe(r): ExRSe(r) \subseteq ExSe^n$
- Internal service and contract assignment (InSeCA) is a one-to-one relationship; the contract assigned to internal service  $se$  is noted as  $InSeC(se)$
- External service and contract assignment (ExSeCA) is a one-to-many relationship; the contract set assigned to external service  $se$  is noted as  $ExSeC(se): ExSeC(se) \subseteq C^n$
- Role and contract assignment (RCA) is a one-to-many relationship; the contract set authorized to role  $r$  is noted as  $RC(r): RC(r) \subseteq C^n$
- Role and task assignment (RPA) is a one-to-one relationship; the policy assigned to role  $r$  is noted as  $RP(r)$

Other essential ideas evolved in the model are discuss below.

1) *Trust negotiation with role hierarchy*

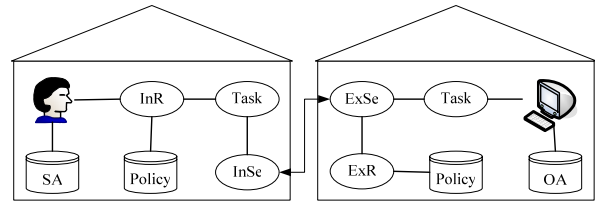


Figure 5. Cross-organization access mechanism

In the commercial environment, most accessing requests are aiming to establish a commercial collaboration between two organizations, which is different from P2P applications. So the accessing subject is just a deputy of the accessing organization, and the collaboration is post related. Correspondingly, the target organization will provide different services to different organizations according to their collaborating operation or their closeness degree. So we introduce role into the united model to mediate the accessing process, as shown in Fig. 5.

As described in the united model, internal service, which is an element constructed to model the remote function, is enveloped with atomic task. So the remote functions can be closely integrated into internal working process. The difference is that supervising role of the task is not decided when the service is invoked by the first time. In this occasion, the applicant with executing permission of the task internal the organization sending identity certificates, internal role policy, and request to the service provider, and then negotiate with it for a policy agreement.

The negotiation process internal organization begins with the comparison between policy of the applicant role and that of the service provider. If the applicant role can not satisfy the policy, the model will search up along the role hierarchy to find a senior role whose policy can satisfy the requirement. Correspondingly, the service provider has to reveal some certificates assertion to satisfy policy of internal role. If the final agreement is achieved, a subject assigned with the internal role will be selected to take charge of the collaboration. And then, the supervising permission of the enveloping task will be assigned to that role, and a contract between the two roles will be send to it.

Different from using internal role hierarchy to express collaboration-oriented inheritance, the external role hierarchy is designed to describe the level of collaboration cross organizations. There must be a lowest role to provide basic services with the least restriction to applicant. Other roles inherit from it with senior services and its corresponding more restrictions to applicant. In order to improve the flexibility of policy mergence, we classify the inheritance into four kinds, when the senior role inherits from multi junior roles, formally defined below:

*Definition 6 (negotiation-oriented inheritance with role hierarchy)*

Suppose that ExR  $r_0$  inherits from  $r_i (i=1 \dots N)$ ,

- Selective inheritance. If the applicant has to follow the policy assigned to  $r_0$ , and that to any one of  $r_i$ s, in order to act  $r_0$ , noted as:

$$r_0 \leftarrow r_1 \vee r_2 \vee \dots \vee r_N \text{ or } r_0 \leftarrow \bigvee_{i=1}^N r_i \quad (2)$$

- Complete inheritance. If the applicant has to follow the policy assigned to  $r_0$ , and that to all of  $r_i$ s, in order to act  $r_0$ , noted as:

$$r_0 \leftarrow r_1 \wedge r_2 \wedge \dots \wedge r_N \text{ or } r_0 \leftarrow \bigwedge_{i=1}^N r_i \quad (3)$$

- M valve inheritance. If the applicant has to follow the policy assigned to  $r_0$ , and that to any  $M$  of  $r_i$ s, in order to act  $r_0$ , noted as:

$$r_0 \leftarrow \bigwedge_{i=1}^M (r_1 \vee r_2 \vee \dots \vee r_N) \text{ or } r_0 \leftarrow \bigwedge_{i=1}^M \bigvee_{i=1}^N r_i \quad (4)$$

- Composite inheritance. If the applicant has to follow the policy assigned to  $r_0$ , and a mixer of the three kinds of inheritance above, in order to act  $r_0$ .

Together with policies, the corresponding attributes of organization that can be exposed to applicant during the negotiation process are assigned to the external roles respectively too. When there is an applicant who did not contract with the organization, the access control system will send polices assigned to roles back to it, from the lowest role up along the hierarchy, receive reply from the applicant, and provide requested attribute assertion if necessary and permitted.

If the negotiation turns out to be successful, the target organization will find an ExR sequence from the lowest role to the one with the requested service. In the same process, the applicant will find a supervising role whose policies agree with the required policies. Then the two sides exchange their attributes certificates required, verify them, and launch the service. At last, the target organization will produce a contract with digital signature; both record it internally and send it to the collaborative organization.

## 2) Policies and Contract

Policy is a familiar concept introduced from ABAC, which is highly emphasized in recent cross-domain access control. Many policy languages have been developed in ATN to describe and implement access control policies of different domains. In the united model, policies are assigned to roles, since that the accesses in collaborative commerce are post related. They can be restored with XACML or any other policy language according to implemental requirement. We recommend XACML, which is XML-based and comprehensible for both human and computer.

Policies assigned to internal role are basic attribute restrictions on internal subjects who want to act it, and on external services which need its cooperation. Policies assigned to external role are attribute restrictions on external subjects and environment. The negotiation process assists service provider to find a responsible internal role whose policy can agree with restrictions of external role on subjects.

According to the special requirement of collaboration among organizations, we bring in contract to describe the

collaborative relationship. Collaborative roles of both sides and the required service are recorded into it. It is produced with digital signatures of the two sides, period of validity, and a rechecking frequency to ensure validity and consistency of collaboration.

## IV. MODEL ANALYSIS

There are many innovational ideas introduced with the construction of the united model, such as the collaboration-oriented inheritance, task combining language, and the negotiation-oriented inheritance. Their performance is simply discussed in this section.

### A. Collaboration-oriented inheritance vs. Conventional inheritance

As shown in Fig. 6, the collaboration-oriented inheritance and conventional inheritance deal with collaboration among roles in different ways. With conventional inheritance, both an administer task and an executive task must be defined in system; and TD between the two tasks must be defined to constrain their duty and executing sequence. With collaboration-oriented inheritance, we only have to predefine one task and assign different operational permissions to different roles, and then the collaboration can be expressed.

So the advantage of collaboration-oriented inheritance lies in that:

- Reduce the scale of task set, since that many tasks are inherently related.
- Simplify some sort of task dependencies, which are more natural to be described with operational permission.
- Strictly support the principle of need-to-do and SOD in that collaboration is inevitably needed to accomplish tasks.

### B. Combination task vs. Workflow

From Fig. 7, we can see the different management to compound functions in conventional T-RBAC and the united model. In conventional T-RBAC, workflow, which is not assigned to role, is a separated entity composed with tasks and a fixed task sequence. However, in united model, combination task is defined as a compound of subtasks, atomic or combinational, and assigned to role the same as atomic task. The executing sequence of subtasks is described with task combining sentence related to the combination task.

The advantage of combination task is that:

- Clarify different responsibilities of collaborative roles in the compound functions, which is unclassified in workflow.
- Define an ultimately responsible role in compound functions to supervise the process. As in Fig. 7,  $T0$  and  $T5$  are administer task which is assigned as supervising permissions of  $T1 \sim T4$  to the ultimately responsible role.

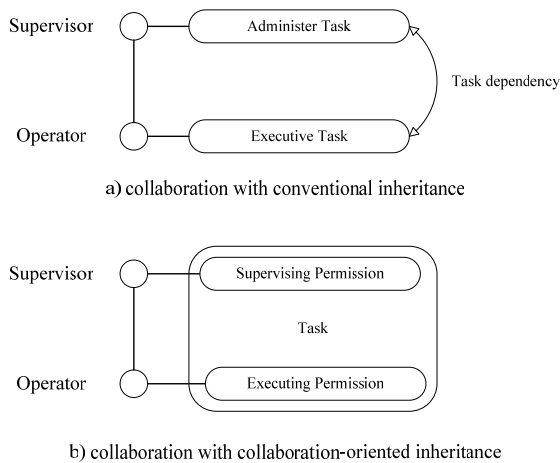


Figure 6. Comparison of conventional inheritance and collaboration-oriented inheritance

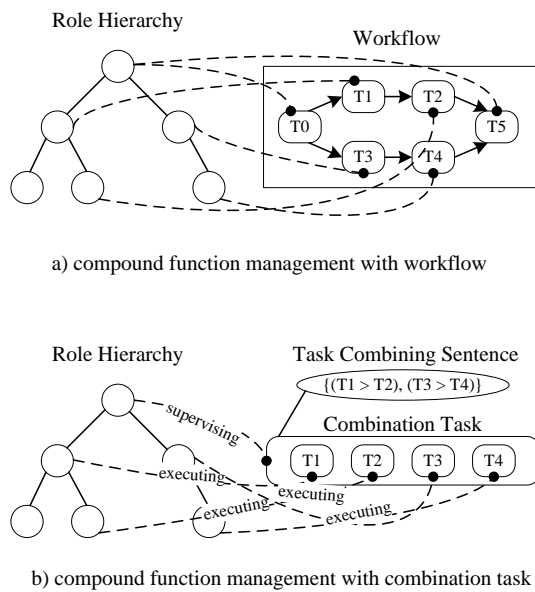


Figure 7. Comparison of workflow and combination task

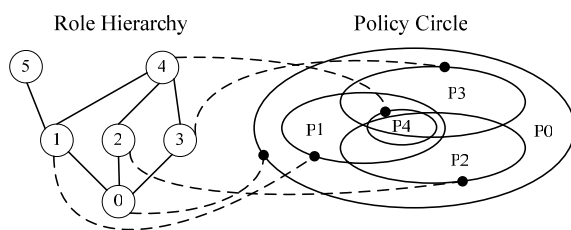


Figure 8. Policy inheritance with role hierarchy

- Remove the restriction of workflow on task sequence with task combining language, so that user can define their own compound functions within the range of their ability.

C. Role-based negotiation mechanism

Negotiation policy is an inevitably needed component in ATN. The service provider must establish a series of pre-policies to decide what policy can be exposed to

applicant with the trust development. However, what degree have the trust developed to is hard to describe. External role and role hierarchy in the model are used to describe the trust degree. Corresponding service level, policy requirement and certificates that can be exposed are assigned to role. The negotiation-oriented inheritance provides a flexible policy composing method, which can freely express the policy required.

As shown in Fig. 8, for example, we use circle to stand for policy assigned to role. Points surrounded in circle are applicants that can be assigned with the corresponding role. So, all the points within circle  $P0$  can be assigned with role  $R0$ . If an applicant wants to act  $R4$ , the corresponding point must fall into the space circled by  $P4$ . It means that the point must go across  $P0, P1, P4$ , and at least one of  $P2$  &  $P3$ . The corresponding inheritance rule is that:  $R4 \leftarrow R1 \wedge (R2 \vee R3)$ .

V. IMPLEMENTATION ARCHITECTURE

In this section, we discuss the functional modules needed to implement the united model in collaborative commerce environment. A generic architecture is described in Fig. 9. To simplify the structure, only the access process from requester to the requested resource is labeled.

As shown in Fig. 9, there are three central functional modules in the access control system, Role Manage Center (RMC), Task Schedule Center (TSC), and Policy Negotiation Center (PNC). They respectively take charge of role activation and maintenance, task verification and deployment, and policy evaluation and decision making.

The internal access process is started from internal users involving the following steps:

- In 1 An internal user sends an invoking request for a task and its initialization information to RMC.
- In 2 RMC verifies that whether invoking permission of the requested task is assigned to the active role. If true, it transfers the request to TSC.
- In 3a TSC verifies the corresponding task combining sentence of the requested task according to TD if necessary, and translates it if the verification is successful. Then, TSC establishes instances of the requested task and sends it to an available user with supervising permission. Particularly, if the requestor has the supervising permission, it will be sent back to him. Next, TSC aids the supervisor to find appropriate executors of the subtasks according to TD and then establishes instances of the subtasks. If the subtask is related to internal function, it will be sent to corresponding users, and the functions will be launched. Else if the subtask is related to internal service, go to step In 3b.
- In 3b TSC communicates with RMC and checks whether there is a contact associated with the internal service or not. If not, go to step In 4, else find the internal role with the contact. If the role is right the supervisor of the whole combination task, he has to provide the contract; if it is not, the supervisor has to get the supervising delegation of the subtask from

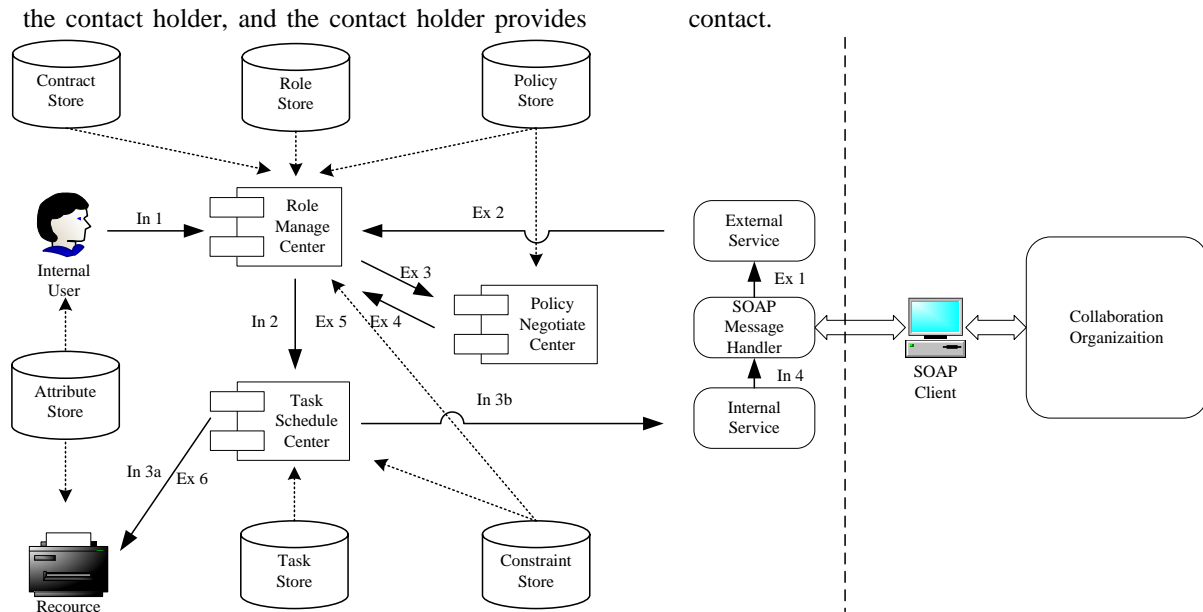


Figure 9. Access control architecture with the untied model

In 4 If there is no contact, the request for service together with the executor identification and role policy will be sent to SOAP Message Handler. Else if there is a contact, then the role policy will be replaced with contact.

The external access process is started from external request involving the following steps:

- Ex 1 The SOAP Message Handler translates the request message and finds the requested external service.
- Ex 2 If there is a contract provided with the request, RMC checks if there is a corresponding contract recorded internal the organization and verifies its validity; and if the verification is successful, go to step Ex 5. If there is no contract, go to Ex 3.
- Ex 3 PNC achieves the policy of requesting role and starts policy comparison and negotiation.
- Ex 4 When the negotiation ends, PNC returns feedback information to RMC. If the negotiation is successful, go on to step Ex 5, else RMC rejects the access request.
- Ex 5 TSC translates the external service into internal task request, and control its execution the same as internal access control.
- Ex 6 It is the same as internal access step In 3a.

## VI. CONCLUSION

As analyzed in section 4, the united access control model is designed to fulfill the whole security requirement of systems in collaborative commerce, including internal functional integration and external policy negotiation. By absorbing and improving spirits from former access control technologies, the proposed model embodies some advantages as that:

- Clear SOD and strictly least privilege. Responsibility is highlighted in the model, even among roles in the same task. In the dynamic authorization process, all the combination tasks will be translated into the atomic tasks which is

contact.

the subtlest authorization unit corresponding to the undividable operation. So both the SOD and least privilege principle are supported.

- Flexible tailorable service. With task combining language, users can freely compose their controllable tasks into combination task, and released to external role as service. Also they can invoke remote service just like local functions.
- Structural negotiation policy. Modeling the negotiation with external role and role hierarchy is a sensible manner for people to deal with external applicants. It is also helpful to system in assigning access policies.

An obvious problem of the model is the complicated structure. But considering that the model is used for large systems in collaborative commerce, it is tolerable. The united access control model can be used to secure Enterprise Resource Planning (ERP) systems in collaborative commerce. In this paper, we focused our energy on expatiating ideas of the united model, since the implemental technologies are already very mature and widely investigated. However, there are still some problems that are not discussed for the moment; such as delegation, information protection within combination task, and problem of cyclic service invocation among several organizations. They are potential research to be investigated, and the detailed research on its implementation will be discussed in future work.

## ACKNOWLEDGMENT

The authors wish to thank Prof. Ma Liang-li, Mrs. Li Juan, and Mr. Li Yong-Jie. This work was supported in part by a grant from Department of Computer Engineering, Information and Electric College, Naval University of Engineering.

## REFERENCES

- [1] Eldon Y. Li, Timon C. Du, and Jacqueline W. Wong, "Access control in collaborative commerce", *Decision Support Systems*, 2007, vol. 43, pp. 675-685.
- [2] Fethi A. Rabhi, Hairong Yu, Feras T. Dabous, and Sunny Y. Wu, "A service-oriented architecture for financial business processes", *ISeB*, 2007, vol. 5, pp. 185-200.
- [3] Jun Li and Alan H. Karp, "Access Control for the Services Oriented Architecture", *SWS'07*, pp. 9-17, November 2007.
- [4] OASIS, The XML Access Control Markup Language (XACML) OASIS TC Homepage, [http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=xacml](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml)
- [5] Ravi S. Sandhu, Edward J. Coyne, and Hal L. Feinstein, et al, "Role-Based Access Control Models", *IEEE Computer*, pp. 38-47, February 1996.
- [6] Thomas RK, and Sandhu RS, "Task-Based authentication control (TBAC): A family of models for active an enterprise-oriented authentication management", *IFIP'97*, pp. 11-13, 1997.
- [7] Eric Yuan, and Jin Tong, "Attributed Based Access Control (ABAC) for Web Services", *ICWS'05*, 2005.
- [8] R. Bhatti, E. Bertino, and A. Ghafoor, "A Trust-based Context-Aware Access Control Model for Web-Service", *ICWS'04*, 2004.
- [9] W. H. Winsborough, K. E. Seamons, and V. E. Jones, "Automated Trust Negotiation", Proceedings of 2000 DARPA Information Survivability Conference and Exposition. *IEEE press*, pp. 88-102, 2000.
- [10] R.S.Sandhu, D.F.Ferraiolo, and R.Kuhn, "The NIST Model for Role Based Access Control: Towards a Unified Standard", Proceeding of the 5th ACM Workshop on Role-Based Access Control, *ACM press, Berlin*, pp. 47-63, 2000.
- [11] Deng Ji-bo, and Hong Fan, "Task-Based Access Control Model", *Journal of Software*, 2003, vol. 14, pp. 76-82.  
邓集波, 洪帆, "基于任务的访问控制模型", *软件学报*, 2003, 14, pp. 76-82.
- [12] Sejong Oh, and Seog Park, "Task-role-based access control model", *Information Systems*, 2003, vol. 28, pp. 533-562.
- [13] LIAO Zhen-Song, JIN Hai, and LI Chi-Song, et al, "Automated Trust Negotiation and Its Development Trend", *Journal of Software*, 2006, vol. 17, pp. 1933-1948.  
廖振松, 金海, 李赤松, 等, "自动信任协商及其发展趋势", *软件学报*, 2006, 17, pp. 1933-1948.
- [14] Ruo-Fei Han, Hou-Xiang Wang, Qian Xiao, and Xiao-Pei Jing, "Investigation in inheritance with role hierarchy for systems in collaborative environment", 2008 International Workshop on Information Technology and Security, *World Academic Press*, pp. 129-133, December 2008.