Proceedings of the 2006 IEEE
International Conference on Mechatronics and Automation
June 25 - 28, 2006, Luoyang, China

Collaboration Access Control Model for MAS Based on Role and Agent Cooperative Scenarios

Zhiyong Zhang, Jiexin Pu and Hongyi Zhang

Electronics Information Engineering College
Henan University of Science and Technology
Luoyang, Henan 471003, China
{zhangzy & pjx}@mail.haust.edu.cn

***Abstract* - Nowadays intelligent multi-agent collaboration research focuses on role-based collaboration framework and approaches. In a large-scale distributed network environment, with shared resource and operations of MAS radically increasing, how to resolve their security question must be taken into account, and role-based collaboration access control could satisfy the requirement as a future direction in the field. But, so far there is little further research on the aspect. This paper proposes and formalizes a general collaboration access control model called CACM for MAS, based on role and Agent Cooperative Scenario that is a new represented concept in the paper. Besides, the related infrastructure, key protocols and core algorithms including role migration, role conflict avoidance and role coordination mechanisms, are also specified in detail. At last, through a practical application based the model, we validate the model is valid and usable, effectively improving MAS security and efficiency.**

Index Terms - Multi-Agent System, Collaboration Access Control, Agent Cooperative Scenarios, Role, CACM for MAS

I. Introduction

Nowadays the theory and technology of Multi-Agent System (abbr. MAS) are hot topic in Distributed Artificial Intelligent research. As an important function of MAS, multi-agent collaboration research has focused on role-based methods that are very useful in building collaborative systems and resolving confliction in recent years [1, 2, 3]. The development of agent-based system must take into account interaction and collaboration, carefully modelling and engineering them, and roles represent a good concept that can help designers and developers deal with these questions [2, 4]. There are fruitful researches as follows. Further, for the sake of advancing the shared resource security in distributed network environment, collaboration access control plays an important role and is crucial to MAS. Comparing with other traditional access control policies, for example, DAC and MAC, Role-Based Access Control (abbr. RBAC) is a future research direction of collaboration access control [5]. But, there is little further research on the aspect in MAS, and lacks of general reference model and related mechanisms.

First, in recent years some role-based collaboration approaches have been already proposed, and Giacomo Cabri evaluates these approaches, presenting their main characteristics and comparing them each other. Moreover, he also proposed an interaction infrastructure, called Rolesystem, and this system allows agents to assume roles and interact accordingly in large-scale distributed web environments [6]. Moreover, as a role-based developing framework, BRAIN (Behavioral Roles for Agent INteractions) Giacomo Cabri proposed aims to cover the agent-based application development process at different phases [7].

Second, in some applications of MAS, researchers and developers pay more attentions to multi-agent collaboration architecture and how to realize dynamic schedule, avoid collision, harmonize each other and cooperative work, and assure higher efficiency, security and stability of MAS. However agent's capability is single or relatively stable in traditional MAS, so it is not fit for dynamic, complex, and indefinite environment of MAS [8]. Accordingly, an incompact federal framework was presented in Reference [9], achieving distributed problem's computing; Andrea Omicmi introduced a role-based multi-agent cooperation model, architecture and related functions, but it lacked of model formalism [10]; Lai also created a role-based multi-agent workflow model for WFMS, and represented it using BNF extensions [11]. These above mentioned models and application do not deal with collaboration access control, so they lack of multi-agent collaboration security mechanism in distributed network environment.

Besides, some other related researches have progressed, too. For example, Haibin Zhu's research aims to role-based collaboration in CSCW environment. Some issues and methods of role-based collaboration through practical applications were represented, such as role assignment and migration [12], role coordination, role collision [3, 13, 14, 15] and so on. Above all, with respect to conflict resolution based on roles, he introduced an Object Model for Collaborative System and further discussed role management and a role-based conflict avoidance and resolution method in the system MCAS [15]. In 2004, Bo Lang provided a method to build a flexible security mechanism that separated the access control policy from the access control decision function by using the concept of meta-policy [16]. The flexible security mechanism can support multiple security policies dynamically.

Summarily speaking, these above mentioned related works do not involves general collaboration access control model, but they are considerably valuable for our theoretic research and application, including related key technology, such as infrastructure, key protocols and core algorithms realization.

II. Our Contributions

According to dynamic, complex, variable environment of MAS and exiting shortcomings nowadays, we do a serial of the following works:

• We proposes and formalizes a new concept "Agent Cooperative Scenarios" (abbr. ACS) that denotes a dynamic agent collaboration environment context, as well as a new reference model called as *CACM for MAS.* The new model flexibly manages agents' capabilities through role and ACS, including role statuses transition, role migration, capability /role conflict avoidance and resolution, agent role coordination.

• The paper specifies realization key technologies related to the model, such as centralized infrastructure, request-reply protocols and key algorithms.

• A practical multi-agent collaboration system based on *CACM for MAS* is illustrated in detail, so it testifies the availability and practical value of the model.

The rest paper is organized as follows. Section III briefly introduces the related theories. Section IV formalizes the model and its characters. Section Ⅴ presents architecture and key protocol functions. A practical application of Multi-agent cooperation based on the model is specified at length in Section Ⅵ. The final section gives conclusions and future research.

III. Agent And RBAC Fundamental Theories

A. *Agent Properties and Multi-Agent Collaboration*

Intelligent agent has some main properties of autonomy, social ability, reactivity and benevolence. These embody that agent could apperceive exoteric environment and react, further affect on real world. One agent's running are not disturbed by others, and it could control its inside status and action; multi-agents have not collision goal each other, cooperative work for an aim through information communication.

The capability of collaborative computing in MAS exceeds single agent, it is a main reason of designing and realizing MAS. Compared with single agent, MAS has some features as follows: every agent only possesses partial system information and computing capability, so its viewpoint is limited. Here agent can not control a whole system, data are dispersive or distributed, and computing procedure is asynchronous, concurrent or parallel. But the structure of MAS could be pure centralized, whole distributed or hybrid form [8]. Because MAS is complicated and dynamic, its collaboration access control is crucial to multi-agent harmony, negotiation, cooperative work in security in an agent society.

B. *Role Based Policy and Representative Models*

Role-Based Policy (abbr. RBP) is a neutral and flexible management thinking that accords with real organization of human beings. It was widely applied in access control, CSCW system, and WFMS, especially in RBAC research field. RBAC mainly includes session, user, role, permission, and constraint concepts, and its superiority embodies that role separates user with permission as a middle entity, making management simple and efficient. Furthermore, "The Least Privilege Principle" and "Data Abstract Principle" actualizing on RBAC are convenient to realize enterprise-level management policy and mechanism. In conclusion, role makes it flexible and dynamic to manage large-scale system with a great deal of entities and operations. The representative models relative to RBP include RBAC96, ANSI-RBAC, etc.

IV. CACM For MAS And Formal Definitions

A. *Proposed Concepts and Ideas*

As a capability management reference model for multi-agent cooperation, the core concept of *CACM for MAS* is role in an agent society. An agent acquiring role(s) could own corresponding multi-capabilities, so achieves definite activity of agent society's task. Furthermore, it introduces a new concept of dynamic ACS on the basis of role-based policy. The model simplifies multi-agent society capability management. Agent not only could active new role in an ACS, but also could discretionarily change granted roles' statuses or become a new role according to new ACS, improving cooperative capability and efficiency. Based on ACS Realizing collaboration constraint and conflict resolution strengthens the security and availability of MAS. Role coordination also improves system concurrency and parallel. *CACM for MAS* is composed of some basic components, such as agent, role, capability, task, activity, society and constraint, as Figure1.There is not role hierarchy, but role migration in the model, which is different from RBAC model.

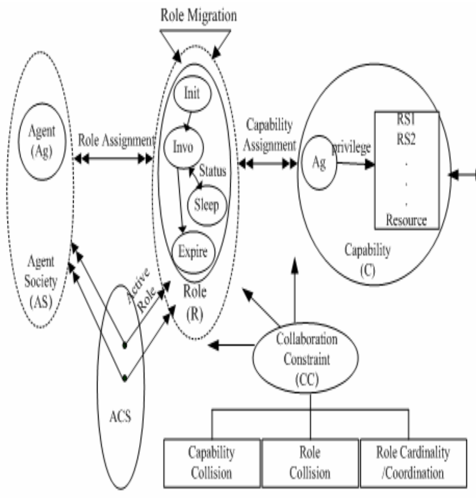

Fig.1 CACM for MAS

B. *Main Components Definition*

The following is a list of CACM for MAS components: Ag, R, AR, S, C, RS, P, CC, AS, T, A are respectively defined as sets of agent, role, active role, states, capability, resource, privilege, constraint, collaboration constraint, agent society, task, activity.

Definition 4.2.1 (Agent): Agent is an entity that has knowledge, faith, intention and expectation. It could apperceive the environment change independently, and make action, further effect environment. So it is a subject with autonomic and social properties. Agent is presented as a seven tuple that consists of AID, sensor, processor, communicator, aim, effector, collaboration scene.

$\forall$ ag (ag $\in$ Ag) ag = { aid, sensor, processor, communication, aim ,effector, acs}

Definition 4.2.2(Capability): Capability is a privilege, permission or condition accessing resource for a task. It could be not only actual read, write a file, but also abstract ability like delegation, revoke, etc. Every capability must and only accomplish an activity. Here defines a function "doactivity()" to denote a one–to-one relation between capability and activity.

$C \subseteq RS \times P$

doactivity (c):$C \rightarrow A$

Definition 4.2.3(Role): Role is an abstract entity who has definite responsibilities and capabilities in an organization or society, agent who acts as a role becomes an instance of the role. Besides role is looked upon as a set of agents, it is also a finite set of capabilities. The relation between role and agent is a many-to-many relation.

$\forall$ r (r$\in$ R) r = {ag_1, ag_2,… ag_n | $ag_i \in$ Ag}

$\forall$ r(r$\in$ R,)r={c_1,c_2…c_m| $c_i \in$ C }

Definition 4.2.4(Society): Society is an organization where some agents or roles cooperate for a goal or task.

$\forall$ s (s$\in$ S) s = {ag_1, ag_2,… ag_n | $ag_i \in$ Ag}

$\forall$ s (s$\in$ S) s = {r_1, r_2,… r_n | $r_i \in$ R}

Definition 4.2.5(Agent Cooperative Scenario): ACS is a dynamic agent cooperative environment that is collaboration context. It includes cooperation time, space, partner, shared resource and attributes that denote shared resource's attributes, and the model does not involve role and capability's attributes.

$\forall$ acs (acs$\in$ ACS) acs = {time, space, parter, resource, attributes }

Definition 4.2.6(Role Assignment): According to cooperative task, agent is assigned to proper roles, and has some related capabilities. RA is a relation of many to many between agent and role.

$RA \subseteq Ag \times R$

R(ag)={r| roles of assigning a agent ag, ag$\in$ Ag}

Definition 4.2.7(Active Role): It is a role that is activated by Agent in a given ACS, AR(acs) denotes active role(s) in ACS. Agent work through a mapping function "agentwork()" between Ag and AR.

$\forall$ acs(acs$\in$ ACS) (ar(acs)$\in$R(ag))

agentwork(ag,ar):$Ag \rightarrow AR$

Definition 4.2.8(Task & Activity): Agent roles' collaboration goal is called as task, the every step of task is called activity. Activity has dynamic and atom characteristic, and is basic unit of task.

$\forall$ t(t$\in$ T)t={a_1,a_2…a_n | $a_i \in$ A}

Property 4.2.1(Agent Society-Task): Every agent society only has a task. Predication "t(s)" denotes the task of society s.

$\forall$ s (s$\in$ S) $\exists$ t_1 ,t_2(t_1 ,$t_2 \in$ T) t_1(s) $\wedge t_2$(s) $\Rightarrow$ $t_1 = t_2$

Definition 4.2.9(Capability Assignment): CA denote assigning appropriate capabilities to role based on responsibility or task of role, and it is many-to-many between role and capability, written as a two tuple CA(r, c).

$CA \subseteq R \times C$

CA(r,c)={c_i| capability assigned to a role r, r$\in$ R, c$\in$ C}

Property 4.2.2(Role/Capability Revoke): When the attributes of resource change, not according with agent's ACS and role/capability, or system end, role/capability will be revoked.

C. Role Temporal Properties

Definition 4.3.1(States Set): The states set of role S={init, invoke, sleep, expire}, init is beginning-state, invoke is active-state, sleep is sleepy-state, expire is exiting-state. Here Role's expire-status does not equal role revocation. If expires, role could be set RTL afresh or revoked.

Definition 4.3.2(Role Time Limit):DCRS has the property of time limitation, DTL={x | x=[τ_{bi}, τ_{ei}](i=1,2…n)}, where τ_{bi} is begin-time , and τ_{ei} is end-time.

Definition 4.3.3(State Transitions):ST is system time, $\forall$ i(i$\in$ N) ST$\notin$ [τ_{bi}, τ_{ei}] $\wedge$ ST< τ_{b1} $\rightarrow$ S=init ; $\exists$ i (i$\in$ N)ST$\in$ [τ_{bi}, τ_{ei}] $\rightarrow$ S=invoke; $\forall$ i(i$\in$ N)ST$\notin$ [τ_{bi}, τ_{ei}] $\wedge$ (ST> τ_{b1}) $\wedge$ (ST< τ_{en}) $\rightarrow$ S=sleep ; $\forall$ i(i$\in$ N)ST$\notin$ [τ_{bi}, τ_{ei}] $\wedge$ST>τ_{en} $\rightarrow$ S=expire.

Property 4.3.1(Activity Temporal Order): The relation of activities is synchronization or concurrent. Between every two activities exits steady time order that meet partial order relation, denoted by "□".

$\forall$ t,a_i,a_j(t$\in$ T,a_i,$a_j \in$ A)($a_i \in$ t $\wedge$ $a_j \in$ t $\rightarrow$ a_i□a_j)

$\forall$ t,a_i,a_j,a_k(t$\in$ T,a_i,a_j,a_k $\in$ A)(a_i□a_j $\wedge$ a_j□a_k $\rightarrow$ a_i□a_k)

Property 4.3.2(Role Run-Order): Role is executed in serious or concurrently, and is partial order, denoted by " $>_{RR}$".

$\forall$ t,r_i,r_j,r_k(t$\in$ T,r_i,r_j,$r_k \in$ t)($r_i >_{RR} r_j$ $\wedge$ $r_j >_{RR} r_k$ $\rightarrow$ $r_i >_{RR} r_k$)

Property 4.3.3(Activity-Role Temporal Order Consistency): Activity and Role/Capability is consistent in time order owing to their one to one mapping relation. Predication "inclu(r_i,a_j)" denotes activity a_i owns delegated capability dc_j.

$\forall$ t,a_i,a_j,r_i,r_j(a_i,$a_j \in$ t)(inclu(r_i,a_i) $\wedge$ inclu(r_j,a_j) $\wedge$ a_i□a_j $\rightarrow$ $r_i >_{RR} r_j$)

$\forall$ t,a_i,a_j,r_i,r_j(a_i,$a_j \in$ t)(inclu(r_i,a_i) $\wedge$ inclu(r_j,a_j) $\wedge$ $r_i >_{RR} r_j$ $\rightarrow$ a_i□a_j)

D. Role/Capability Constraint Rules

CACM for MAS strengthens role/capability assignment management, preventing agent from deliberately or involuntarily acquiring illegal privileges, and improving system security and controllability. The important basic rules include role/capability non-collision constraint and role cardinality constraint, respectively acting on role assignment, migration and role coordination. They are defined as follows.

Definition 4.4.1(Collision Capability): Two capabilities c_i and c_j are collision capabilities, if they are not assigned to a agent ag_k at the same time, we denote it by $collc(c_i, c_j, ag_k)$.

Definition 4.4.2(Collision Role): Two roles r_i and r_j are collision roles, if they are not assinged to a agent ag_k or or are actived at the same time, we denote it by $collr(r_i, r_j, ag_k)$.

Constraint Rule 4.4.1(Non-Collision Constraint): Every two capabilities or roles do not exist assignment collision in R and C sets.

$$\forall c_i,c_j\ (c_i\in C,\ c_j\in C,\ ag_k\in Ag)\ collc(c_i, c_j, ag_k)=F$$

$$\forall r_i,r_j\ (r_i\in R,\ r_j\in R,\ ag_k\in Ag)\ collr(r_i, r_j, ag_k)=F$$

Definition 4.4.3(Role Cardinality Constraint): Role cardinality is the maximal number of agents acquring the role. So the number of every role in R set does not exceed its cardinality that is a natural number commonly.

E. Agent Dynamic Properties

Definition 4.5.1(Agent Role Migration):
According to ACS, Agent could change from a role to another of acquired role set. Role migration operation is written as $\backsim$, denoting transition from former role to latter role in ACS. Role migration relation is equipollence having reflexive, symmetric, transitiv properties.

$$\forall ag, r_i, acs\ (r_i\in R(ag),\ acs\in ACS)\ r_i \backsim r_i = T$$

$$\forall ag, r_i, r_j, acs\ (r_i\in R(ag), r_j\in R(ag), acs\in ACS)\ r_i \backsim r_j = T$$

$$\forall ag, r_i, r_j, acs\ (r_i\in R(ag), r_j\in R(ag), acs\in ACS)\ r_i \backsim r_j \rightarrow r_j \backsim r_i$$

$$\forall ag, r_i, r_j, r_k, acs\ (r_i\in R(ag), r_j\in R(ag), r_k\in R(ag), acs\in ACS)\ r_i \backsim r_j \wedge r_j \backsim r_k \rightarrow r_i \backsim r_k$$

Definition 4.5.2(Agent Role Coordination): According to ACS and system efficiency, system must adjust some current the number of agent acting on the definite role in real-time. Predication ARC(ag, r, acs) denotes in ACS the number of ag satisfies role r coordination requirement, that is, it is in a definite natural number range.

$$\forall ag, r, acs\ (ag\in Ag, r\in R, acs\in ACS)\ ARC(ag, r, acs) = T$$

Ⅴ. CACM For MAS Key Technologies

A. Collaboration Access Control Infrastructure

The paper introduces RoleCertificate (abbr. RC), which is a text information on role relative attributes, and the infrastructure of *CACM for MAS* contains two parts of capability/role management and access decision based on the concept, as Figure 2. The former is in charge of assignment and revocation capabilities/role to agent in constraint conditions that are stored in role constraint database beforehand, and the procedure are accomplished before system initialization as well as system end, influencing agent capability access decision; the latter mainly accomplishes agent capability control based on initialization and real-time ACS in the process of system execution. As a complementary access track function, audit subsystem records all activities of agent accessing to shared resource, further safeguarding MAS security.

Every agent acquires the relevant role(s) and access to shared resource only via RC that includes all assigned roles to the agent and assignment timestamp at the beginning of system. With system running, the information of role migration, role coordination, role status is also recorded dynamically in RC as an accessory. RC is formally defined as follow using BNF:

```
RC::=<Basic Information><Accessional Information >
/* static role information*/
<Basic Information>::={roleid, rolename,
                       initial-status, roleattributes,
                       timestamp
                       }
/* dynamic role information*/
<Accessional Information>::=<role migration><role
coordination ><role status transition>
 <role migration>::={migrationtime,
                    pre-migration role,
                    post-migration role,
                    timestamp
                   }
<role coordination >::={agent number of every role}
<role status transition>::={current status,
                           timelength,
                           fore-status
                           }
```

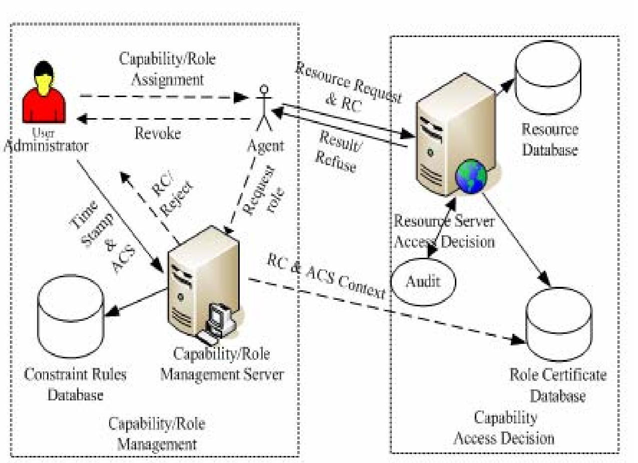
Capability/Role Assignment
Revoke
User Administrator
Agent
Resource Request & RC
Result/ Refuse
Resource Database
RC/ Reject
Request role
Time Stamp & ACS
Resource Server Access Decision
Audit
RC & ACS Context
Constraint Rules Database
Capability/Role Management Server
Role Certificate Database
Capability/Role Management
Capability Access Decision

Fig.2 Centralized collaboration access control infrastructure based on RoleCertificate

B. Key Protocols

The key protocols and related semantics of this model are illustrated in Table1, for example, agent role request function AgentRequest; Server reply function ServerAnswer; Server role assignment and revocation function ServerAssign, ServerRevoke; the two protocols related to agent role migration are request-reply protocols on RoleMigration and MigrationPermit. These protocols are called by Management Server and Agents on different system period, and are also close relative to core algorithm in foregoing Section.

TABLE I
KEY PROTOCOLS

Key Protocols	Related Semantics
AgentRequest (ag, server, RequestFlag, Id, ACS, timestamp)	Agent sends request for role/capability to Management Server in ACS and timestamp.
ServerAnswer (server, ag, result, timestamp)	Server sends result to agent in timestamp after verification of ACS context.
ServerAssign (server, ag, RequestFlag, Id, timestamp)	Server assigns role/capability to agent in timestamp.
ServerRevoke (server, ag, RequestFlag, Id, timestamp)	Server revokes role/capability from agent in timestamp.
RoleMigration (ag, server, oldrole, newrole, ACS)	Agent migrates from an oldrole to newrole and notifies server.
MigrationPermit (server, ag, result)	Server decides whether permit migration or not and informs agent about result.

Considering multi-agent collaboration on heterogeneous platform and software migration, the above mentioned key functions were realized in Java, and defined main entities' attributes and methods. Based on these protocols, integrated process of agent role management and access to resource is as follows:

• **System Initialization** Agent calls agentRequest (ag, server, RequestFlag, Id, ACS, timestamp) and sends request of role or capability, after receiving the request, Management Server estimates whether assignment or not , then, calls ServerAnswer(server,ag, result, timestamp) to send result. At last, if result is permitting assignment, administrator directly calls function ServerAssign(server,ag, RequestFlag, Id, timestamp) to assign roles/capabilities to agent, not through agent's request. Administrator could directly assign role/capability to agent according to system requirement in Management Server, not via agent's request. So, the former two request-reply protocols could be omitted on some occasions. After assignment, system initialization is end, so agent acquires defined role(s) and capabilities. This procedure needs some core algorithms, such as role assignment Algorithm 1mentioned in Section 5.3.

• **System Executing** In the procedure of agent's accessing to resource, it could send role migration request at any time through RoleMigration (ag, server, oldrole, newrole, ACS), if permit, Server send MigrationPermit(server,ag,result). This procedure needs role migration Algorithm 2 in Secition 5.3, involving role conflict resolution and coordination. Besides, in the executing process, Server could also revoke agent's role or capability at any moment when their attributes are not consistent with ACS.

• **System End** Server calls ServerRevoke(server,ag, RequestFlag, Id, timestamp) to revoke assignment and accomplish last works.

C. *Core Algorithms Realization*

Algorithm 1 (Role Assignment & Access Decision)

```
Role assign_access (ag, assignedrole, RC, ACS): AssignFlag
or AccessFlag
Input ag, assignment, RC, ACS
Output AssignFlag, AccessFlag
{/* Role assignment
  read  (RC, ACS)
  for (i=1;n;i++)
    consresult=contraintrule(assignedrole, R[i])
   If  consresult=0 then return(0);
  end;{for}
  assign (ag, assignedrole,timestamp);
  write (ag, assignedrole,timestamp,AC);
 /* Access Decision
 read (ag, currentroles)
 decisionresult=accessdicision (ag, currentroles, resource)
 if decisionresult = 0 then refuse (ag);
 else
    execute access process;
end}
```

Algorithm 2 (Role Migration)

```
RoleMigration(ag, RC,ACS,  newrole): MigrationFlag
Input ag, RC, ACS, newroleid
Output MigrationFlag
{  read (AC,ACS)
   for (i=1; n; i++)
     if AR[i].status=invoke then
       consresult=contraintrule(newrole, AR[i])
     if consresult=0 then return(0);
   end;{for}
   if curragent>=maxagentnumber then
        Return (0);
      else
         newrolestatus=run;
        write (newrole,timestamp,status,RC);
       return (1);
    end
 end }
```

VI. A PRACTICAL APPLICATION BASED ON CACM FOR MAS

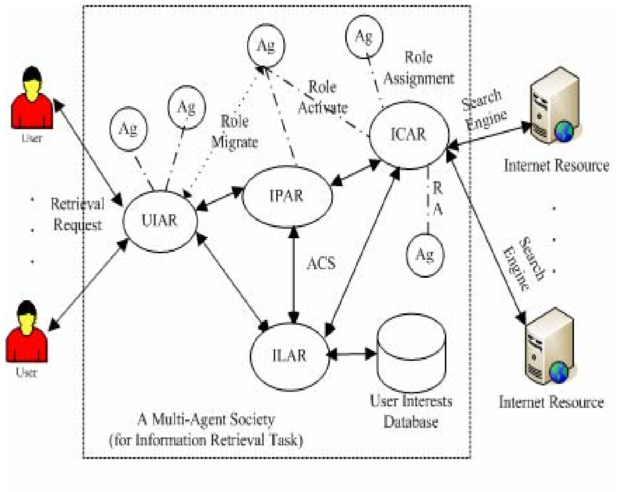

Fig.3 Framework of ABTIRA based on *CACM for MAS*

The designing and realization of Agent-Based Text Information Retrieval System (ABTIRS) adopted this model, so improved on the issue of agent's capability and cooperative work, advancing system efficiency and parallel processing. The system defined and realized User Interface Agent (UIA), Information Collection Agent (ICA), Information Processing Agent (IPA), Interests Learning Agent (ILA)and their corresponding role based on different responsibility and capability, such as UIAR, ICAR, IPAR and ILAR, respectively accomplishing retrieval demand analysis and feedback, text information collection, information filtering, as well as user' interests collection. Multi-Agent Collaboration Management Subsystem (MACMS) implemented agent-role assignment, role-capability assignment and collaboration constraint. Moreover, Agent Capability Access Decision Subsystem (ACADS) controls a serial of role dynamic change based on the above mentioned architecture and key protocol functions in section 5.Because agents are intelligent and independent, they could not only adjust and activate their new roles according to the change of ACS, but also migrate role for sharing resource and cooperation in this system. The framework of ABTIRA is as figure3.

Ⅶ. CONCLUSIONS

CACM for MAC is a general multi-agent collaboration access control reference model supporting role status transition, migration, conflict resolution and coordination based on role and ACS. In the model, agent's capability is not single and static, but multiple and dynamic assignment, and agent role constraints improve agents' cooperative work and security of MAS. One of the future researches is visual modeling the model using object-oriented modeling language, for instance, UML, on the basis of CSCW visual modeling research [17]; the other is further enriching the model semantics when agent and role attributes changes in any time and in any way.

REFERENCES

[1]Zhu, H., "Role Mechanisms in Collaborative Systems", *International Journal of Production Research*, vol. 44, no. 1, pp. 181-193, Jan. 2006.

[2]Giacomo Cabri, Luca Ferrari and Letizia Leonardi, "Agent Role-based Collaboration and Coordination: a Survey About Existing Approaches", *Proc. of The 2004 IEEE Systems, Man and Cybernetics Conference*, Hague, Netherlands, October 2004.

[3]Zhu, H., "A Role-Based Conflict Resolution Method in a Collaborative System", *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics (SMC'03)*, Washington D.C, USA, Oct. 2003.

[4]Zhang Zhiyong, Pu Jiexin and Feng Changyuan, "Multi-Agent System Management Model Based on Role & Agent Collaboration Scenarios", *Computer Application and Software (Proc. of China National Open and Distributed Parallel Computing Symposium 2005)*, Vol.22, No.10, pp.107~109, Shanghai, October, 2005.

[5]Zhao Bin, "Collaboration Access Control", *In Seminar on Network Security 2001, Telecommunications Software and Multimedia,* 2001.

[6]Giacomo Cabri, Letizia Leonardi and Franco Zambonelli, "Implementing Role-based Interactions for Internet Agents", *Proc. of The 2003 International Symposium on Applications and the Internet (SAINT),* Orlando, Florida, USA, January 2003.

[7]Giacomo Cabri, Luca Ferrari and Franco Zambonelli, "Role-based Approaches for Engineering Interactions in Large-scale Multi-Agent Systems", *Software Engineering for Multi-Agent Systems II*, Pereira de Lucena, Garcia, Romanovsky, Castro, and Alencar editors, Lecture Notes in Computer Science n. 2940, pp. 243-263, April 2004.

[8]Wang Junsong and Cui Shigang, "Multi-Agent Technology and Applications", *Computer Engineering and Applications*, 2003, 18：61-62，66.

[9] Ren Ming and Wang Chengdao, "Research of Multi-agent Cooperating Based on Federation Structure", *Journal of East China University of Science and Technology*, 2004, Vol.30, No.3:311-314.

[10]Andrea Omicmi, Alessandro Ricci and Mirko Viroli, "RBAC for Organisation and Security in an Agent Coordination Infrastructure", *Electronic Notes in Theoretical Computer Science*, 2005 (128): 65-85.

[11]Lai Huixia and Zhang Shi, "Role-Based Multi-Agent Workflow Model", *Computer Applications*, 2004, Vol.24, No.12:316-318.

[12]Zhu, H. and Seguin, P., "The Role Transition Mechanisms in Role-Based Collaborative Systems", *Proceedings of IEEE Canada Conference on Electrical and Computer Engineering (CCECE'05)*, Saskatoon, Canada, May, 2005, pp. 1305-1308.

[13]Haibin Zhu, "Some issues of Role-Based Collaboration", *Proceedings of IEEE CCECE2003- CCGEI2003*, Montreal, May, 2003.

[14]Haibiin Zhu, "A Role-Based Conflict Resolution Method for a Collaborative System", *Proc. of IEEE INTERNATIONAL CONFERENCE ON SYSTEMS MAN AND CYBERNETICS*, 2003, 4135-4140.

[15]Zhu, H., "Conflict Resolution with Roles in a Collaborative System", *International Journal of Intelligent Control and Systems,* Vol. 10, No.1, pp.11-20, 2005.

[16]Bo Lang ,You Lu, Xin Zhang and Weiqin Li , "A flexible access control mechanism supporting large scale distributed collaboration", *proceedings of the 8th International Conference on Computer Supported Cooperative Work in Design*,Vol.1,pp.500- 504 , 26-28, May, 2004.

[17]Zhang Zhiyong and Pu Jiexin, "Delegation Model for CSCW Based on RBAC Policy and Visual Modeling", *Proceedings of The 11th Joint International Computer Conference 2005* , World Scientific Press Company, Nov.10-12，Chongqing, China,126-130.