

The Role Analysis and Transitions in a Collaborative System

Haibin ZHU

Abstract- Roles provide a very natural and powerful way for an enterprise administrator or security officer to describe the privileges of various job functions. Roles are also a powerful and policy neutral concept for facilitating distributed systems management and enforcing information sharing. Role-based methods are very useful in building collaborative systems. This paper explains why role concepts in designing a collaborative tool are important, focuses on a discussion of the basic concepts required in a role-based design, the role assignment, management and transition in the system. It analyzes the roles and role transitions in a collaborative system. It also emphasizes the values of roles in the system analysis, design and implementation. In the last section, it concludes that a role-based method can help greatly to design and implement collaborative systems and still needs more comprehensive research.

Index Terms—Role, role analysis, role transitions, collaborative systems

1. INTRODUCTION

Computer-Supported Collaborative Work (CSCW) is an increasingly prevalent means of connecting individuals and teams. Much research is being undertaken to improve virtual environments in order to enhance the ability of collaborators to interact effectively and cooperatively.

CSCW systems combine communication, computer, and decision support technologies to support problem formulation and solution in meetings. Team members, however, are dependent on mediated interactions for coordination, and as a result, are likely to face important deficits in the information they have about the day-to-day activities of their teammates [15].

Collaboration in a computer-supported environment differs significantly from collaboration in a face-to-face environment. Even though scientists are working hard to provide a virtual environment to make collaboration easy, and the collaborators work hard to be cooperative, there are still many problems for collaborators to identify and overcome when using a collaborative system. For example, group interactions are significantly constrained by the structure of the current infrastructure [5].

Beside the communication platforms, research on collaborative systems is mainly concerned about two other categories: information sharing and coordination [20]. Currently, there are problems that need to be resolved in these two areas, such as flexibility of a workflow system, protection, security and efficiency in information sharing, and conflict presentation and resolution in both coordination and sharing requirements.

We may expand information sharing to sharing which includes view sharing [24]. If we can support sharing with reasonable time and cost factors, we can provide acceptable collaboration platforms. It has been accepted

that a role-based method is one good solution to sharing based on the research of access control [18, 19, 20, 21, 23].

Another key activity of collaboration is coordination. In coordination, workflow management systems model the sequence of subtasks in a work process and the roles performed by each individual. Even though there are no roles declared in traditional CSCW applications or systems, there are implicit roles granted in the design and application of them. Hence, we can state that without roles, there would be no collaboration. In the CSCW community (where human agents cooperate) there is no doubt about the importance of roles [2]. However, only a few publications have emphasized roles and role-based methods [6, and 12] in CSCW applications, even though RBAC has been very successful in the last few years.

Edwards introduced access control policies and roles to avoid chaos in collaborative applications [6]. This experience was a positive step in the development of a new role-based model for collaboration. Guzdial, Rick, and Kerimbaev experienced in supporting roles management in the application of a collaborative tool CoWeb, and concluded that role supporting is necessary for long-term and wide-spread use of collaborative tools [12].

Voting is the final step in a group decision procedure, however, no practical voting tools were provided. We believe that a voting tool is a typical collaborative system that has almost all the requirements an ordinary CSCW application has. It requires group management, workflow control, and role management. Therefore, we use a voting tool as an experimental collaborative tool to apply a role-based design.

The role concept includes the characteristic features as follows [10]:

- Various roles of an entity may share common structure and behavior;
- Entities can acquire and abandon roles dynamically;
- Roles can be acquired and abandoned independently of each other;
- Entities exhibit role-specific behavior;
- Roles restrict access to a particular context; and
- Entities may occur repeatedly in the same type of role.

All the above features facilitate the design and implementation of a collaborative tool.

This paper discusses a role-based design and an implementation of a voting tool, and this tool's implementation reflects many aspects of designing a collaborative tool with roles.

The other sections are organized as follows. Once we briefly introduce the basic concepts used in the role-based design, we discuss the role specifications and the

transitions roles in the voting tool. Then, we discuss some implementation issues relevant to the role-based design in the voting tool. Lastly, we summarize the properties of roles in the voting tool and draw the conclusion that a role-based method can be helpful in building collaborative systems and propose some new issues that will require more research concerning role-based design of collaborative systems.

2. THE BASIC CONCEPTS IN THE ROLE-BASED DESIGN

Role definition is a difficult job in managerial and behavioral sciences and many consultant companies offer services such as position or role descriptions to help enterprises to find a correct person to do a special job [3, 13, 14].

In RBAC, a role is defined a semantic construct forming the basis of access control policy [21]. However, it is argued that the role concept should be defined with more consideration at the point of view of objects [9]. We also need to define and clarify the role concept in collaboration with our specific requirements.

In behavioral science, a role is defined as a prescribed pattern of behavior expected of a person in a given situation by virtue of the person's position in that situation [13]. In other words, a role is defined by the responsibilities that a human can or must take. Nevertheless, it is difficult to describe a role clearly and strictly because natural languages are by their very nature ambiguous. For this reason, different persons in identical positions may make different contributions.

In a collaborative system, we may specify responsibilities clearly to a person with some special technical supports. The possibility is due to the exactness of computer languages and the limited computer resources a person can control at a time.

Our basic idea about a role is that if a human user, logged in a collaborative system that can designate clearly what objects he/she can access with specific rights and which users he/she can manage or communicate with, he/she can accomplish his/her jobs meaningfully and efficiently. This can be done by specifying front-end interfaces [4].

The basic concepts have been established by accepting ideas from object-orientation [10, 17], agent-orientation [1, 2, 16], behavioral science [3, 13, 14] and RBAC [7, 8, 21].

In a role-based system, there should be users, roles, objects, operations and permissions [8] wherein,

- A user is defined as a human being.
- A role is a job function within the context of an organization with some associated semantics regarding the authority and responsibility conferred on the user assigned to the role.
- Permission is an approval to perform an operation on one or more related objects.

- An operation is an executable image of a program that upon invocation executes some functions for the user.

In our design, we introduced roles and groups with some object-oriented concepts because roles are generally defined and enacted within groups [13].

- An object is used to express everything in a collaborative system [17].
- A user is a participant in collaboration.
- An operation is a command that may be issued by a human user. Note that we use operations instead of messages, because they directly affect the objects in a system.
- An interface is a set of accessibilities (including create, read, modify or delete) and a list of operations to objects in the system or to the system itself.
- A role is a special object that symbolizes a logged user in the system and it is related with an interface.
- A group is a special object defined logically as a set of roles.
- We accept classes as the templates of objects [17].

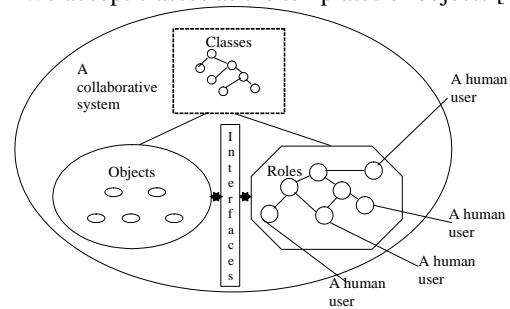


Fig. 1. The collaborative mode with roles

With the introduction of roles, a collaborative system will support the collaboration mode depicted in Fig. 1, where each user should log in the system with a role; with a specific role, he/she uses a specific interface to access objects in the system, to perform operations to the system and transfer to other roles. With the interactions of roles, collaboration takes place and the outcomes are the objects in the system.

Evidently, this mode supports both synchronous and asynchronous collaboration [24]. In this mode, the collaborators can become aware of others by accessing the objects including roles through the interfaces [11, 22].

3. THE ROLES AND OBJECTS IN THE VOTING TOOL

In the voting tool, roles are special objects to represent human users' states in the system. In fact, a role is defined in the system by an interface used by a human user. This interface specifies what objects he/she can access and what operations he/she can perform to the system. We can overcome the ambiguities of a role with an exact interface (Fig. 2).

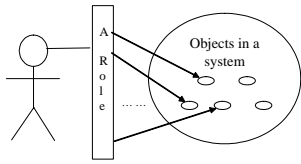


Fig. 2. A role is an interface for a human user to access the objects in a system.

In this voting tool, we use static roles to enhance the coordination in a voting procedure. By static roles we mean that the roles are defined and incorporated in the system design and implementation. The workflows are definite according to the specific working threads for predefined roles.

Before defining the roles, we need to define some classes of objects as follows in order to support voting activities.

- User: A user object expresses a user who has signed up to the system. These objects are only managed by an administrator of the system.
- Group: Conceptually it is a set of users with different roles. A group may be private or public that affects its accessibility by a role.
- Topic: It is a meaningful object that group members would like to vote on. A topic might also be private or public that affects its accessibility by a role.
- Vote: It is a special permit for a group member to vote.

In order to support a voting procedure in a flexible way, the following roles were defined. However, the roles may be defined more clearly with the interface implementation.

- User: Every one who has signed up and logged in to use the voting tool. He/She can access (read and modify) his own profile, and can select a group to join.
- Administrator: A special user designated with special priorities. He/she has the responsibilities for managing all the other users' registrations. He/She can manage all the objects in a system and is

granted the role of a group leader when entering a group and that of a topic moderator when entering a topic.

- Group leader: A special user who has created a group. He/She can manage the group and all the topics in the group. He/She is granted the role of a topic moderator when he/she enters a topic in his/her group. That is to say, the role group leader is a super role of the moderator [10].
- Member: A special user having joined a group. He/She can access (read) all the topics in the group and select one to enter. The member role is also a super role of the voter.
- Moderator: A special user who has created a topic. He/She can manage (read, modify, delete) the topic and the votes for it.
- Voter: A special member who has a vote for a topic. He/She can vote on this topic.
- Guest: A member who has no vote. He/she can only read the topic information.

In the system, roles are confined with different operations to different objects such as groups, topics and votes. Table 1 shows the primary specification of roles in the voting system. The group, topic and vote are specific for the roles relevant based on the current state of a user. For example, a group leader can modify and delete the group he/she created and a moderator can modify and delete the topic he/she created. That a voter can delete a vote means that after voting he/she actually deletes the vote. The administrator can access all the operations to roles and objects in the system such as create, modify, delete, and assign. By this table, we can also find that the roles with the same names such as a group leader have not the same accessibility for objects.

“User” is not listed in the table as an object because a user with any role can only access the user object relevant to him or her, i.e., user objects are inaccessible objects for normal roles.

Table 1: The specification of roles in the voting system

Role	Object	Group						Topic						Vote		
	Operation	C	Join	M	D	Enter	Exit	C	M	D	Enter	Vote	Exit	C	D	Apply
User		X	X													
Group Leader				X	X	X	X	X	X	X	X		X	X	X	
Member						X	X	X			X	X	X		X	
Moderator									X	X	X			X	X	
Voter											X	X	X		X	
Guest											X		X			X

Keys: C - Create; M - Modify; D - Delete; and X - Access

4. THE TRANSITIONS OF ROLES

Role transitions are complex tasks in a collaborative environment. Role management is a complex task that must be paid more attention. The more roles there are, the

more complicated the Petri nets are for describing transitions among roles, and the more complicated algorithms and regulations are to solve conflict situations. Therefore, there should not be too many roles in one situation at one time in a system, i.e., roles must be managed with separate times and contexts. That is to say, a

user can only play one role at a time even though he holds many roles at the same time.

With the descriptions of the roles in the above section, we illustrate the transitions among roles with Fig. 4 - Fig. 10. The human users access an interface with the operations shown in the figures and the operations may create a new interface for a user's role transition. In general, each role may become a user when he/she logs off the system. We can use the set concept to demonstrate the general relationships among the roles (Fig. 3). From this figure, we know that a user might hold a group leader, a member, a moderator, a voter or a guest at the same time. A user cannot hold an administrator and a member at the same time. It also shows us that some roles must be transferred from other roles. For example, any role other than "user" must be transferred from the role "user". We use dotted line for human users to express this set's openness. In the figures, we use ellipses to express roles and rectangles operations.

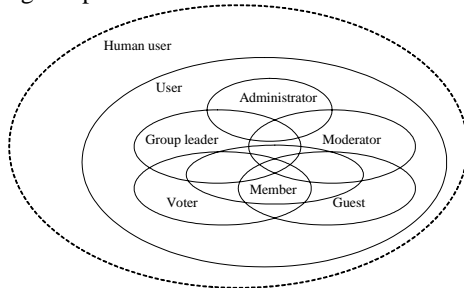


Fig. 3. The relationships among the roles.

2.1 User

A user object is created when a user or simply user signs up the system. Before logging in the system, a user, expressed by the dotted line in Fig. 3, cannot participate in collaboration. After a user has logged in the system, he/she becomes a logged user ("User" in Fig. 4) and can access some objects in the system and perform operations to the system. He/She can also transfer to other roles such as a group leader, or a member, by activating specific operations. Note that, a user becomes an administrator by a special login process.

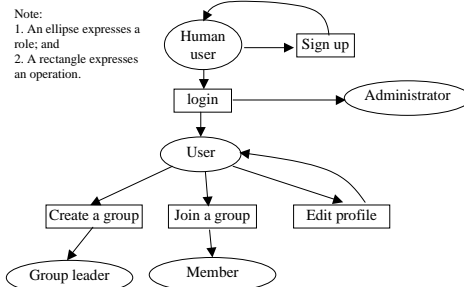


Fig. 4. The transitions for a user.

2.2 Administrator

An administrator is a special logged user. He/She manages the users, groups, and roles in the system. He/She does not transfer to other roles (Fig. 5).

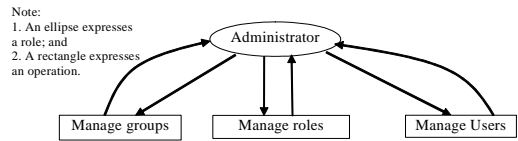


Fig. 5. The transaction of an administrator.

2.3 Group leader

After he/she has created a group, a logged user becomes a group leader. As a leader of a group, he/she can transfer to a moderator by entering or creating (in the sub-interface of topic administration) a topic (Fig. 6).

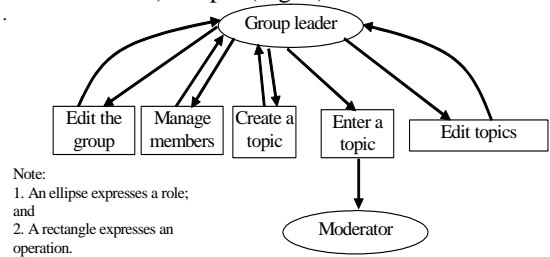


Fig. 6. The transitions of a group leader.

2.4 Member

A member is a logged user who has joined in a group. He/She can access the objects in a group. He/She can transfer to other roles such as a guest, a voter or a moderator by specific operations (Fig. 7).

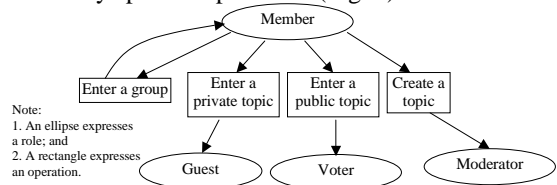


Fig. 7. The transitions of a member.

2.5 Moderator

A group leader becomes a moderator when he/she enters a topic and a member may become a moderator when he/she creates a topic (Fig. 6). A moderator can moderate a topic and manage the votes of this topic (Fig. 8). He/She reverts back to a user when he/she logs out the system (Fig. 8).

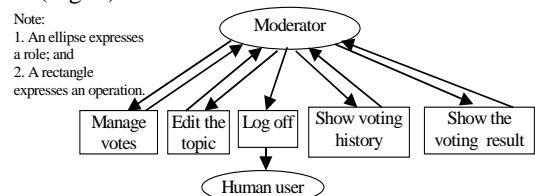


Fig. 8. The transitions of a moderator.

2.6 Voter and guest

A member becomes a voter when entering a public topic

and a guest when entering a private topic (Fig. 7). A guest can become a voter when he/she gets a vote. A voter or a guest reverts back to a user when he/she logs out the system (Fig. 9 and Fig. 10).

Note:

1. An ellipse expresses a role; and
2. A rectangle expresses an operation.

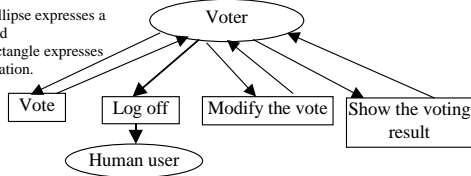


Fig. 9. The transitions of a voter.

Note:

1. An ellipse expresses a role; and
2. A rectangle expresses an operation.

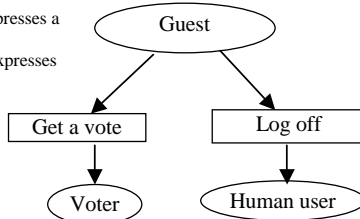


Fig. 10. The transitions of a guest.

5. IMPLEMENTATION

Our aim is to get some experience and practice by implementing a voting tool that is a typical collaborative tool with role-based methods. An additional benefit is that this voting tool can be used for a group interested in making a final decision after having discussed a relevant topic with other discussing tools such as BBS, Webboard, WebCT, etc. It can also help an instructor conduct online testing. In our implementation, all the roles are formed by a browser a user is working on. A user is clear what role he/she is playing. In the implementation, we need to consider different aspects based on the roles in the system.

5.1 Basic Assumptions

The basic assumptions for the voting tool are as follows.

- 1) Every user wants to join a group to vote for a specific topic.
- 2) The roles and the transitions among them mainly concern the activities relevant to the voting.
- 3) Votes are changeable. That means a voter can change his/her vote in some situations.
- 4) The vote cannot be changed after a voter has seen the voting result. This is a function very useful for an instructor to conduct an online testing or an online assignment submitting.
- 5) Every voter can only vote for one topic in one group at one time, in other words, one voter has a separate vote for each topic.
- 6) If one moderator hopes to activate a voting procedure again after some votes have been submitted, he/she may save the voting result to initiate a new round of voting.

5.2 The States of Roles

Roles may have different states during collaboration. Roles with different states may interact with the system by different interfaces. For example, in the voting system, a voter may have four states (Fig. 11): votable: the voter can vote for the topic; voted: the voter can modify the vote he/she has submitted; and done: the voter cannot modify his/her vote.

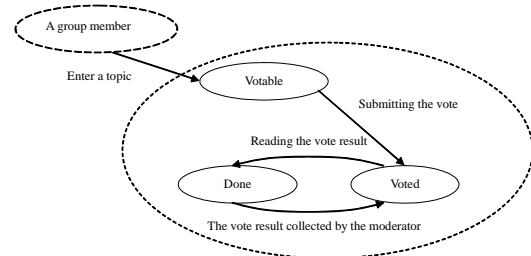


Fig. 11. The states of a voter.

A moderator has three states (Fig. 12): applied: the moderator is waiting that the created topic is approved; approved: the moderator can edit the contents of the topic; and voted: the moderator cannot change the contents because some voters have voted.

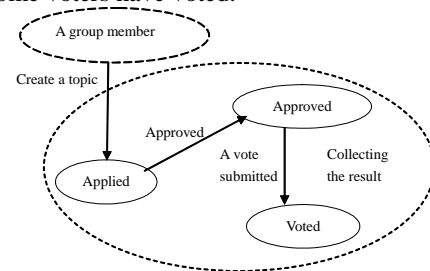


Fig. 12. The states of a moderator.

5.3 Special considerations about roles

Some special strategies as follows are considered in an implementation of a voting tool relevant to roles. These considerations show that roles are specified not only by the operations relevant to it but also the processing strategies of the system.

- 1) When can a moderator read the voting result?
 - After the votes have reached a specified number: this requires the moderator to set a number to collect the voting results. This number guarantees the objectiveness of the voting results.
 - When the due date is imminent: this is useful when there is a deadline for the moderator to collect the results. This result might be not as objective as that of the strategy with a specified number of votes.
 - Any time he/she wants to read: this can help the moderators to find the evolutions of the voting results.
- 2) When can a moderator edit a topic?

- Before no voter votes for it: this is reasonable because the moderator should not affect the voting results by modifying the topic. It produces an objective result.
 - After the moderator has collected the result: this is reasonable because the moderator has collected the result for the earlier voting version and can re-collect the result for the next one.
- 3) When can a voter see the voting result?
- After he/she has voted: it is a common situation for distance learning applications. A student can read the answer only after he/she has submitted his/her answer.
- 4) When can a voter modify the vote?
- Before he/she reads the voting results; or
 - After the moderator has collected the voting result and started a new round of voting.

The rationalities to consider such a question are based on that a voter should express his/her final decision.

5.4 Forming voting workflows

Workflows are important in facilitating collaborative activities. Managing roles actually provides a way to support workflows. In the implementation, some workflows are built by the role transitions and the cooperation among roles.

For example, a human user's workflow (U) is formed as follows when his or her role transits from a user to a voter.

- U_1 : A user applies to join in a group so as to become a member;
- U_2 : The member selects one topic to enter and becomes a voter; and
- U_3 : The voter uses the vote.

At the same time, other roles will cooperate to form another workflow (V) as follows.

- V_1 : The group leader approves the user's application;
- V_2 : The moderator approves one vote; and
- V_3 : The moderator gathers the votes.

5.5 Dynamic roles

Dynamic roles are necessary for collaboration [6]. In our practice with Microsoft ASP (Active Server Page), we defined half-dynamic roles in the designing and implementation of the voting tool besides the static roles specified in section III. By half-dynamic we mean that the operations the users can perform in the system are limited and definite. An administrator can add, delete, and modify a role, a group and a user based on the limited number of operations (Table 2). Actually, in our voting system, a role is defined all the operations to all the objects tables in the system. An administrator is responsible for role definition and user assignment. There are two ways in user assignment; (1) The direct way is that the roles are assigned directly by the role manager; (2) The indirect way is that the roles are assigned based on the application of a user.

For totally dynamic role mechanisms, we must provide a platform to dynamically add, delete, and modify the objects and operations to the system. We need to clarify the issues such as exact role definition and representation, role-based workflow definition and the application of newly defined roles. We are now working on a role-based model to support thoroughly dynamic roles to support role-based activities in collaboration.

Table 2: The administrator role

Role	Object	Group			Role				User		
	Operation	Create	Modify	Delete	Create	Modify	Assign	Delete	Create	Modify	Delete
Administrator		X	X	X	X	X	X	X	X	X	X

5.6 Roles' identities

In a software system programmed by an object-oriented programming language such as C++ or Java, we often encounter a problem to transfer the object identification. That is to say, we must have an identity to access an object. In the voting tool, when a user logged in, he/she must play a role. The question is: should we attach a role's identity to the user? The answer is evidently "Yes". In our implementation, we take advantages of the session mechanism of ASP. A session tells a user's current state. A session can help define many current states to facilitate the role the user is playing. That is to say, defining a user variable specifies the current user and defining a role variable is actually stating the current role of the user.

6. CONCLUSIONS

Roles clearly exist in a simple voting system and role transitions are common for users. Clearly specifying roles can help us to implement definite workflows more easily and make the collaborators in the voting procedures understand better what they can and cannot do. Clearly, the system we discussed uses the role-related interfaces to control the rights of users to access the system's resources including operations and objects. This system resembles the ideas used in RBAC.

From the analysis of roles in the voting system, we find the properties of roles in this system:

- A role is transferred when a user activates an operation;
- A user plays only one role at a time;

- There is a hierarchy between roles. A role might be superior to another role. E.g., the group leader is a super role of the moderator role, i.e. the operations of a group leader cover all those of a moderator.
- A role is different from another even their names are the same. We can state that roles have a property of polymorphism.
- A user is reminded of his/her role by the user interface;
- The system's implementation is based on roles; and
- The roles determine the processing and workflows of the system.

This tool demonstrated that we could define a role with an interface by specifying the accessibilities to objects in a system. It can be used by a variety of groups to make a final decision after discussion and to support an instructor in conducting online testing. However, we need to develop new mechanisms to facilitate this requirement when the number of the classes of objects in a system is large.

REFERENCES

- [1] Barber, K. S., Liu, T. H., Ramaswamy, S., "Conflict detection during plan integration for multi-agent systems", *IEEE Trans. On Systems, Man, and Cybernetics- Part B: Cybernetics*, vol. 31, no. 4, August 2001, pp. 616-628.
- [2] Becht, M., Gurzki, T., Klarmann, J., Muscholl, M., "ROPE: Role oriented programming environment for multiagent systems", *Fourth IECIS International Conference on Cooperative Information Systems*, September 1999, pp. 325- 323.
- [3] Biddle, B.J., and Thomas, E.J. (Ed), *Role Theory: Concepts and Research*, John Wiley & Sons, Inc.
- [4] Dewan, P., and Shen, H., "Controlling access in multiuser interfaces", *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 5, no. 1, March 1998, pp. 34 – 62.
- [5] Dourish, P., "Using metalevel techniques in a flexible toolkit for CSCW applications", *ACM Transactions on Computer-Human Interaction*, vol. 5, no. 2, June 1998, pp. 109-155.
- [6] Edwards, W. K., "Policies and roles in collaborative applications", *Proc. of Conference on Computer-Supported Cooperative Work (CSCW'96)*, Cambridge, USA, 1996, pp. 11-20
- [7] Ferraiolo, D. F., Barkley, J. F., and Kuhn, D. R., "A role-based access control model and reference implementation within a corporate intranet", *ACM Transactions on Information and System Security (TISSEC)*, vol. 2, no. 1, February 1999, pp. 34-64.
- [8] Ferraiolo, D. F., Barkley, J. F., and Kuhn, D. R., "Proposed NIST standard for role-based access control", *ACM Transactions on Information and System Security*, vol. 4, no. 3, Aug. 2001, pp. 224-274.
- [9] Genilloud, G., Wegmann, A., "A foundation for the concept of role in object modeling", *Proc. of the 4th International Enterprise Distributed Object Computing Conference (EDOC2000)*, Makuhari, Japan, September, 2000, pp. 76-85.
- [10] Gottlob, G., Schref, M., and Röck, B., "Extending object-oriented systems with roles", *ACM Transactions on Information Systems (TOIS)*, vol. 14, no. 3, July 1996, pp. 268-296.
- [11] Gutwin, C. and Greenberg, S., "The effect of workspace awareness support on the usability of real-time distributed groupware", *ACM Transactions on Computer-Human Interaction*, vol. 6, no. 3, September 1999, pp. 243-281.
- [12] Guzdial, M., Rick, J., and Kerimbaev, B., "Recognizing and supporting roles in CSCW", *Proc. of the ACM 2000 Conference on Computer supported cooperative work*, Philadelphia, Pennsylvania, United States, December 2000, pp. 261-268.
- [13] Hawkins, D. I., Best, R.J., and Coney, K. A., *Consumer Behavior*, Business Publications, Inc., Plano, Texas.
- [14] Hellriegel, D., Slocum, J.W., Jr., Woodman, R. W., *Organizational Behavior*, West Publishing Co. St. Paul, Minnesota.
- [15] Jeffrey, P. and McGrath, A., "Sharing serendipity in the workplace", *Proc. of the Conference on Collaborative Virtual Environments (CVE 2000)*, San Francisco, USA, 2000, pp. 173-179.
- [16] Jung, H., Tambe, M., and Kulkarni, S., "Argumentation as distributed constraint satisfaction: applications and results", *Proc. of the Fifth International Conference on Autonomous Agents*, Montreal, Quebec, Canada, 2001, pp. 324 – 331.
- [17] Kay, A. C., "The early history of Smalltalk", *The Second ACM SIGPLAN History of Programming Languages Conference, ACM SIGPLAN Notice*, vol. 28, no. 3, March 1993, pp. 69-75.
- [18] Kern, A., Kuhlmann, M., Schaad, A., and Moffett, J., "Role engineering: Observations on the role life-cycle in the context of enterprise security management", *Seventh ACM Symposium on Access Control Models and Technologies*, Monterey, California, USA, 2002, pp. 43-51.
- [19] Park, J. S., Sandhu, R. and Ahn, G. J., "Role-based access control on the web", *ACM Transactions on Information and System Security (TISSEC)*, vol. 4, no. 1, February 2000, pp. 37-71.
- [20] Poltrock, S. and Grudin, J., "CSCW, groupware and workflow: Experiences, state of art, and future trends", *Proc. of the conference on CHI 98 summary: human factors in computing systems: human factors in computing systems*, Los Angeles, California, United States, April 1998, pp. 119-120.
- [21] Sandhu, R., Bhamidipati, V., and Munawer, Q., "The ARBAC97 model for role-based administration of roles", *ACM Transactions on Information and System Security (TISSEC)*, vol. 2 no. 1, February 1999, pp 105 – 135.
- [22] Schmidt, K., "The problem with "awareness": Introductory remarks on "awareness in CSCW"", *Computer Supported Cooperative Work (CSCW)*, *The Journal of Collaborative Computing*, vol. 11, no. 3-4, 2002, pp. 285-298.
- [23] Zhang, Z., Haffner, E., Heuer, A., Engel, T. and Meinel, C., "Role-based access control in online authoring and publishing systems vs. document hierarchy", *Proc. on the Seventeenth Annual International Conference on Computer Documentation*, New Orleans, Louisiana, United States, October 1999, pp. 193-198.
- [24] Zhu, H., Turoff, M. and Li, Z., "An object model for collaborative systems and a toolkit to support collaborative activities", *Proc. of the 2000 American Conference on Information Systems (AMCIS'00)*, Long Beach, USA, August 2000, pp. 590-594.



Haibin Zhu received B.S. degree in Computer Engineering in 1983 from Institute of Engineering and Technology, China, and M.S. and Ph.D. degrees in computer science in 1988 and 1997 from the National University of Defense Technology (NUDT), China.

He is an Assistant Professor of the Department of Computer Science and Mathematics, Nipissing University, Canada. He was a visiting professor and a special lecturer in the College of Computing Sciences, New Jersey Institute of Technology, USA from 1999 to 2002. He was a lecturer, an associate professor and a full professor from 1988 to 2000 at NUDT. He has published about 50 papers, three books and one book chapter on object-oriented programming, distributed systems, collaborative systems and computer architecture.

He is the receipt of a 2nd Class Nation-Level Award of Education Achievement from Ministry of Education of China (1997), a 2nd Class Nation-Level Award of Excellent Textbook from the Ministry of Education of China (2002), three 1st Class Ministry-level Research Achievement Awards from DOD of China (1997, 1994, and 1991), and a 2nd Class Excellent Textbook Award of the Ministry of Electronics Industry of China (1996).

Dr. Zhu is a member of IEEE and a life member of the Chinese Association for Science and Technology, USA.