An Object-Oriented Organizational Model to Support
Dynamic Role-based Access Control in Electronic Commerce Applications

Edward C. Cheng
University of London, Birkbeck College, Computer Science Department. OCT Research Laboratory. edwardc@octlab.com

Abstract

Role-based access control (RBAC) provides flexibility to security management over the traditional approach of using user and group identifiers. In RBAC, access privileges are given to roles rather than to individual users. Users acquire the corresponding permissions when playing different roles. Roles can be defined simply as a label, but such an approach lacks the support to allow users to automatically change roles under different contexts; this static method also adds administrative overheads in role assignment. In electronic commerce and other cooperative computing environments, access to shared resources has to be controlled in the context of the entire business process; it is therefore necessary to model dynamic roles as a function of resource attributes and contextual information.

In this paper, an object-oriented organizational model, OMM, is presented as an underlying model to support dynamic role definition and role resolution in RBAC. The paper describes the OMM reference model and shows how it can be applied flexibly to capture the different classes of resources within a corporation, and to maintain the complex and dynamic roles and relationships between the resource objects. Administrative tools use the role model in OMM to define security policies for role definition and role assignment. At runtime, the resource manager queries the OMM system to resolve roles in order to authorize any access attempts. Similarly, cooperative computing software uses OMM to support task assignment and access control to business processes. Contrary to traditional approaches, OMM separates the organization model from the application model; thus it allows independent and flexible role modeling to reflect realistically a dynamic authorization subsystem in a rapidly changing business world.

1. Introduction

Resource Manager (RM) implementations have historically focused on technologies around access methods, concurrency control, and logging and recovery [1,2,3]. The security model and access control usually assume a simple and static model which are based on user and group identifiers. Many cooperative computing software such as workflow, group scheduling, and electronic commerce applications, simply adopt the user and security model of a relational database management system (RDBMS) as their access control model. However, the user model in RDBMS is designed primarily to support access control in processing isolated transactional operations rather than integrated process activities [4]. It is thus not adequate to model the flexible resource relationship that is required to support cooperative works. Others have proposed specific role models and methodologies for concurrent engineering such as M*-OBJECT[5], ORM [6], SAM* [7], and ObjectFlow [8,9]. They all start from the process view and tightly couple the organization model with the role model, and some even with the process model.

Some implementations attempts to isolate an organization component from the workflow engine. For instance, WorkParty, a workflow system by SIEMENS, has an organization component called ORM which is a standalone client-server database application to support organization modeling [10]. ORM has an application programming interface (API) and a graphical user interface to allow users to define and populate the organizational database. Although ORM separates the organization model from the process model, it did not separate the organization model from the role model; the two are still integrated. Also, the organization definition and the role definition of ORM are still static like other prior arts, and it suffers from lack of a dynamic relationship model.

Still other former efforts address the organizational resource management issue through directory service. Directory services (DS) and other naming services aim to support distributed object lookup with a naming convention [11]. Each object on the system is assigned a static and universally unique identifier (UUID). This approach yields an efficient solution for simple point-to-point interaction in collaborative software; it resolves static addresses for electronic mail, video conferencing, group scheduling and the likes. Nevertheless, DS lacks an organization model and support for dynamic relationships between resources. Consequently, it fails to support advanced applications such as in E-commerce where deferred binding till run-time is
necessary to find out the entity that should be responsible to handle or authorize the transaction.

Overall, the existing organizational resource management approaches suffer from the following weaknesses:

- lack of a conceptual organization reference model. We need a generic solution so that we can apply the model to different concurrent engineering and cooperative computing environments.
- tightly integrated with the process and application architecture. Consequently, it is only adequate to support the BPM systems which observe the specific models.
- support only some predefined resource types. Network DS focuses on machine nodes, users and applications; messaging DS on user addresses; and BPR organization sub-components on users, groups and roles. To support the collaboration between the different applications and users in E-commerce, the organization model must be extensible and flexible in order to define multidimensional resource types which include employees, departments, products, machines, projects, accounts, and others.
- assume only static and hardwired relationships between resources. In reality, relationships between resources are rapidly changing. Relationship exist not only among resources of the same type, but also among different types of objects. For instance, there is a many-to-many relationship between the company projects and its employees. Similarly, a three-way relationship can be defined between customers, bank accounts, and loan agents.
- lack of openness to integrate with other organizational management systems. There are existing directories and organizational resource information systems that run in a cooperative environment. A comprehensive architecture must take consideration to exchange information with the existing solutions.

This paper discusses an organizational and role model, namely OMM (Organization Modeling and Management). The OMM methodology supports the conceptual design and the design implementation phases of the enterprise modeling cycle [12]. It includes a conceptual and reference model for enterprise modeling. OMM does not assume a particular process or application architecture. With this generic approach, OMM is able to map its object types to other organizational data schemes to present an integrated multidimensional view of organizations to E-commerce applications. The paper presents the role resolution concept in groupware and uses a expense voucher example to illustrate how the OMM system is used to support flexible task assignment and authorization in a cooperative computing system.

2. The OMM Methodology

With the workflow approach, process routing control is abstracted from the application logic; it thus results in a flexible design and implementation of flow logic without interfering the implementation of the associated applications. The flow logic concerns mainly the routing decisions throughout the life of a process instance. The

![Figure 1. Expense Voucher Process](image)

Petri-net representation in figure 1 illustrates a flow description of a simple expense voucher [13,14].

In this example, an expense voucher is processed electronically over the network, and a number of human resources are involved. This business process can be initiated by any worker in the company. The submit_voucher step will be executed by the same person. The manager_approval step must be executed by the manager of the flow-initiator. Similarly, the VP_approval task, which is only created when the expense balance is greater than a certain amount, must be run by the vice president of the division to which the flow-initiator belongs. The last four steps in this example, decision, accept, reject, and notify, can be done without any human intervention. A sophisticated workflow system supports the definition of this process by allowing an administrator to define this flow-map through some graphical or scripting interface. The workflow data which impact the routing decision of the flow are also defined as part of the flow definition. Agent applications associated to individual steps are connected to the workflow system through some workflow programming interface. Finally, roles are defined to control task assignment and task authorization. Since the role model of

![Figure 2. OMM Separates the Role and Organization Models](image)
the current BPM systems does not support dynamic relationships between corporate entities, it is not adequate to support even a simple flow such as the one above.

Contrary to existing systems, OMM methodology separates the organization model from the role model of the BPM system [10, 15, 16]. Figure 2 shows how the different models interface with one another. With the OMM approach, organizations are modeled separately from the business processes and applications. Role definition and resolution are done through the organizational modeling and management interface.

3. The OMM Organization and Role Model

The OMM employs a generic reference model which can be applied flexibly to define different resource types, the roles they play, and their inter-relationships. Resource types are user-defined; they may include people, machines, robots, applications, processes, products, customers, and others. Modeling of an enterprise involves defining these classes of resources and the dynamic relationships between these resource objects. A workflow system is interested in assigning tasks to a subset of these resources such as employees, robots, or customers. Furthermore, it may associate a step with a certain application, or assign a set of machines that the step should be running on.

There are three fundamental entities in the OMM model, namely the organizations, members, and virtual links.

An enterprise is composed of a number of OMM organizations. Each organization represents a class of corporate resources such as employees, departments, products or projects. Each member object within an organization maps to an actual entity of the corporation. Members of the same class share a common set of attributes and methods that is extensible by the user. A member can relate or link to other members through virtual links. Contrary to static connections, a virtual link only has a relationship definition which is evaluated and resolved at runtime. Figure 3 uses an E-R diagram to show the OMM model. We shall discuss each of the OMM objects in greater detail.

3.1 Organizations

An enterprise is composed of a number of OMM organization objects. OMM organizations are created to map to the different dimensions and components of a company. Each organization has a unique identifier across the global enterprise. Using the OMM organizations, a company can be partitioned both vertically and horizontally; vertically into different dimensions or resource types, such as people, machines, projects and so on. Horizontal partitioning can be applied to break components of the same type into smaller units, such as breaking people into engineers, sales and marketing, and temporary workers.

For instance, during the organization conceptual design phase, an OMM organization may be defined to represent the people of the company, another to represent the different projects, even another to represent the robotics machines, and so on. This creates a view of vertical partitioning of the corporation. Each partition keeps the organization information of a particular dimension. In addition, we can further divide an organization within the same dimension horizontally. For instance, people belonging to the engineering department may be included in one organization, while people in the marketing department are placed in another. In other words, vertical partitioning helps to define the different types of resources within the enterprise, while horizontal partitioning allows users to logically divide resources of the same dimension into smaller sub-components. Figure 4 depicts the horizontal and vertical partitioning of the OMM organizations.
Since different departments or divisions now own their individual organizational definition, a much greater level of autonomy in defining and managing organizational information is granted to them. They can update, delete, or append to their own organizational definition without impacting others. For major restructuring, users may alter the organization schema which corresponds to their units only. In addition, the granularity of partitioning is controlled entirely by the user. Users have the flexibility to decide how fine they want to divide the organization. When the business conditions change, they may choose to merge together or to further divide their organizations.

At the organization design-implementation phase, we will consider the database schema of the OMM organizations. The different partitions of OMM organizations correspond naturally to database tables. It is typical to use some tables within a database environment to capture the information of an organizational partition. The OMM methodology does not dictate the underlying data model, although our current implementation uses an RDBMS. When a relational database implementation is chosen, users define the attributes of the members as columns in a relational table. In an object-oriented database environment, the member attribute definition maps directly to a class definition. This constitutes a class of members for each OMM organization.

3.2 Members and the Information Model

OMM uses an object-oriented model to capture its member information. An enterprise has a main member class which is the super-class containing a list of system-defined attributes and methods. All user-defined member classes are subclasses of the main member class and inherit the properties of the super-class. Figure 5 shows the class hierarchy.

![Figure 5. Class Hierarchy of OMM Member](image)

The identifier attribute is unique for each member across the entire enterprise. Each member object has a name that is given by the user and is unique only within an OMM organization. Each member object in OMM goes through a life-cycle which is represented by the state transition diagram shown in figure 6.

When a resource is created, it enters the active state. Thereafter the state changes are triggered by the user through a member method, setState(). A member object may cycle between the active and inactive states, simulating the reality of some resources being suspended, off-line, or on-leave. When a resource is removed, its information may still be retained in the repository and be queried until it enters the forgotten state which corresponds to the situation where the resource information is archived away.

![Figure 6. State Transitions of an OMM Member](image)

In OMM, member ownership can be transferred from one organization to another. When a member is moved to another organization, some of the user-defined attributes from the original organization may be mapped to the new one, and all other irrelevant information is dropped. However, the system-defined attributes are always retained. This maintains the unique identity of the member even though it may be moved around the enterprise from place to place.

From the class definition point of view, the OMM model is similar to the Object Class in the directory model of X.500 [11]. OMM members are different from the members of X.500 in that they support class inheritance, method extension, and object life-cycle. The latter captures the dynamic behavior of a resource within an actual corporation. It also allows the workflow engine to properly perform worklist management; based on the state of a resource at runtime, the workflow manager may choose to avoid pushing a task to a worker unless it is in the active state, thus reduces the possibility of assigning work to personnel that are unavailable. Furthermore, the OMM model is unique in that members may relate dynamically to one other through virtual links.

3.3 Virtual Links and the Relationship Model

As collaborative effort exists between company resources, it is necessary to model relationships between them [17,18]. OMM uses virtual links to define dynamic
relationships between member objects. Virtual links are rules constructed based on the member attributes and contextual variables. The OMM engine evaluates the rules to identify roles and relationships that resources have in the company. In OMM, a relationship is established from one resource to another, and as such it can be represented as a directed edge. If a bi-directional relationship (such as supervisor-subordinate relationship) is desired, it can be modeled as two relationships; one as a reverse relationship of the other. In this respect, resource objects are like nodes while virtual links are the directed edges in a graph. A virtual link is defined by the following BNF syntax [19]:

\[
\text{<Virtual Link> ::= <Owner>, <Relationship Type>, <Expression>, <Organization Scope>}
\]

\[
\text{<Owner> ::= null | <Member ID>}
\]

\[
\text{<Relationship Type> ::= <Relationship Name> [REVERSE] [TRANSITIVE]}
\]

\[
\text{<Relationship Name> ::= <Character String Constant>}
\]

\[
\text{<Expression> ::= <Expression> <Rel Op> <Expression> | <Attribute Name> <Op> <Value>}
\]

\[
\text{<Attribute Name> ::= <Character String Constant>}
\]

\[
\text{<Op> ::= == | != | >= | > | < | <=}
\]

\[
\text{<Value> ::= <Constant> | <Attribute Name>}
\]

\[
\text{<Rel Op> ::= AND | OR}
\]

\[
\text{<Organization Scope> ::= <Organization Name>}
\]

The connection between resources is dynamic and virtual because the relationship is defined with a regular expression over the attributes rather than a pair of static resource IDs. There may be a predefined owner of a relationship, or the owner can be associated at runtime. When a user resolves a virtual link, the relationship expression is evaluated over the member attribute values, and there may have any number of resources satisfying the criteria indicating a relationship with the owner in question. An example of a relationship can be:

- **Owner:** null
- **Relationship Name:** manager_of
- **Expression:** (owner.deptNo == X.deptNo) AND (X.jobCode < 101)
- **Organization Scope:** employee

To find all the employees that are under the managerial responsibility of an executive with member ID `john_smith`, we can resolve this virtual link upon the employee organization. The owner is set to `john_smith`, its attribute values are retrieved and used to substitute corresponding fields in the virtual link expression. Each member within the employee organization is evaluated against the expression; the `X.attributeName` is substituted with the corresponding attribute values of the member under evaluation.

Despite dynamic characteristics of relationships in OMM, hard coded relationships between two specific entities can still be modeled with virtual links. To define that Mary Ann is acting_for John Smith, we have:

- **Owner:** 'mary_ann'
- **Relationship Name:** acting_for
- **Expression:** (X.name == 'john_smith')
- **Organization Scope:** employee

Note that a link may or may not be transitive in nature. When a transitive relationship `r1` is defined, and if member `m1` relates to member `m2` in `r1`, and `m2` relates to `m3` in `r1`, it follows that `m1` also relates to `m3` in `r1`. Obviously there is a cost associated with resolving transitive relationships; they should therefore be used with care.

When defining a relationship type, a reverse relationship can be specified. For example, if relationship types `r1` and `r2` are defined as reverse relationships to each other, and if member `m1` relates to `m2` in `r1`, then `m2` relates to `m1` in `r2`.

Figure 7 shows a relationship graph within an organization; note that here the supervisor_of and the subordinate_of are represented by reverse links to each other:

![Figure 7. OMM Relationship Graph](image)

Although the example only covers relationships within an organization, virtual links can actually be defined across multiple organizations. In this case, the organization scope will list all OMM organizations involved. For instance, a relationship graph may be desirable to represent the connections between a project and its machine resources and the employees who are involved in the project. Here, the owner is a particular project while the organization scope will include both machine and employee.

A virtual link may also be defined between a member and an organization. When an OMM organization object is part of a relationship, all member objects within that organization are involved in it. For instance, if Tom Moore is a supervisor of an organization, then all resources within that organization are supervised by him.

Using virtual links a workflow system assigns and authorizes steps flexibly to resources who play different roles in the company. Referring to the expense voucher process in section 2, the roles “manager of the initiator of the flow” and the “vice president of the division in which the initiator belongs” can be expressed easily with the regular expression in virtual link.
4. Role Resolution

Concurrent engineering technology supports business process integration and automation [2,20]. It provides a framework on which multiple tasks and applications are integrated to form a network of steps to accomplish a business process [13]. A business process can be formulated as a set of nodes, representing the tasks or steps, connected by some directed edges which are condition arcs governing the route of the process (refer to figure 1 for a Petri-net representation of a workflow process). To ensure this model has a consistent flow behavior, a process always has a BEGIN and an END step. The BEGIN step only has outgoing arcs and the END step only has incoming arcs. The other steps exist between BEGIN and END have one or more incoming arcs and outgoing arcs [9].

As the process progresses in time, different tasks are created and assigned to various resources in the company. Sometimes a particular resource may be chosen to execute a step (the push model), other times a group of workers are identified as potential candidates to perform a task; the workers will pick up the task on their own choice (the pull model). In both cases, authorization checking must be performed when someone attempts to open and work on a workflow step.

The task assignment and task authorization, which have to do with role resolution, are among the biggest challenges of a successful workflow solution. Role resolution refers to identifying the right resources that are playing a certain role [6,18,22]. Two types of questions are asked in role resolution. One is definitive in nature, and the other is relational. The following examples illustrate both types of questions respectively:

1. Is X an engineer? Or who are the engineers? (definitive)
2. Is Y the manager of X? Or who is X’s manager? (relational)

The OMM methodology provides a strong basis to model task assignment and authorization in a workflow management system [2,11]. Although the syntax of assignment and authorization specifications in workflow is implementation dependent, most systems support the abstraction of roles to allow more flexibility than simply using user Ids [2,9,22]. A control statement is usually included in the step definition for that purpose. The following syntax of control statement illustrates the notion of such specifications:

```
{... step definition ...} <Control Statement>;
<Control Statement> ::= <Operation> BY <Role>
<Operation> ::= READ | WRITE | EXECUTE | MANAGE
<Role> ::= <Role Label>
<Role Label> ::= <Character String Constant>
```

The following is an example of authorization statement within the workflow script:

```
{... step definition ...} READ BY Manager;
```

With the current state of the art, users are assigned to take different roles identified by labels like 'Manager'. The use of role labels, although has more flexibility than simply using a user name in the control statement, does not support resource relationships which are required in most realistic business processes such as the expense voucher process discussed in section 1. With OMM, role definition can be expanded to cover relationships:

```
<Role> ::= <Role Labels> | <Relationship Name> 
Resource> ::= <Character String Constant>
<Role Label> ::= <Character String Constant>
$INITIATOR_OF_PROCESS
```

where $INITIATOR_OF_PROCESS is a workflow system-defined data item which can be retrieved through the workflow interface. The workflow script now reads:

```
{... step definition ...} EXECUTE BY manager_of $INITIATOR_OF_PROCESS
```

At runtime, when a member M attempts to open this step for execution, the workflow engine will query the OMM system to verify if M is a manager_of the initiator of the flow process. Here manager_of is a virtual link, M becomes the owner, and the initiator of that process instance is the member in question. The authorization checking therefore reduces to the question

Is M the manager_of of the $INITIATOR_OF_PROCESS?

where manager_of is defined by an expression such as:

```
(owner.deptNo == X.deptNo) AND (X.jobCode < 101)
```

or

```
(M.deptNo == $INITIATOR_OF_PROCESS.deptNo) AND
$INITIATOR_OF_PROCESS.jobCode < 101)
```

M is the user accessing the process step, and $INITIATOR_OF_PROCESS is known by the workflow engine, the above expression can be evaluated to return a boolean value of whether M is authorized to execute this step. For instance, if M is Charles and the $INITIATOR_OF_PROCESS is Susan, then:

```
Charles is Susan’s manager if:
(Chars.deptNo == Susan.deptNo) AND (Susan.jobCode < 101)
```

5. The OMM System Architecture

At the design-implementation phase, the existing organizational databases, such as the human-resource (HR) database and the corporate directory, are analyzed and mapped to the OMM organization design. Based on this
mapping, the *agent programs* which make up a part of the OMM server architecture can populate the OMM data store by accessing the existing databases. In some cases, due to the continual usage of legacy HR applications over the existing organizational databases, it is necessary to periodically refresh some part of the OMM data store by rerunning the agent programs. The actual mapping of the various database schemes to the object-oriented OMM scheme is outside the scope of this paper.

At runtime, the resource manager accesses the OMM organizational information and perform role resolution by calling the OMM application programming interface (API). The OMM server evaluates the rule representing this role on the current organization database and return the result to the RM. A graphical administrative tool also calling the OMM API is used for users to manage the organizational objects through a graphical user interface (GUI).

Figure 8 shows the OMM runtime system architecture with the RM and existing organizational databases.

![Figure 8. OMM Runtime System Architecture](image)

### 5.1 Domain UUID and Naming Convention

Each OMM server attends clients within a *domain*. OMM servers exchange information with one another through the regular OMM APIs. A domain corresponds to a physical implementation of a data store in OMM. Multiple OMM organizations may reside in a domain but an organization does not span across domains. A domain has a globally unique identifier while organization names are unique only within a domain. However, the relatively unique name of an organization, combined with the unique domain name, must be a universally unique identifier (UUID). For instance, domain *london* and domain *seattle* may both contain an organization named *employee*. The corresponding unique organization names will look like this:

- employee.london
- employee.seattle

Similarly, although a member name is only unique within an organization, by concatenating the member name with the UUID of the organization, we can obtain a UUID for the member as well. For instance, the member names

- john_smith.employee.london
- john_smith.employee.seattle

are globally unique.

For a user to be able to access the global organizational information, updates to domains and OMM organization definitions ought to be propagated to all OMM servers on a regular basis (such as once every hour). It is not necessary to escalate updates of members, virtual links, or attributes outside of a domain, for the organization UUID will indicate if the underlying information is managed by another server. Based on this UUID, the local server may retrieve data from the remote server.

### 6. Related Work

Some prior significant directory technologies have been developed with the intent to provide global infrastructure. The X.500 Directory Service [11] supports remote directory access, centralized and distributed topologies, and centralized or distributed update methods, as well as peer-entity authentication, digital signatures, and certificates [23]. The X.500 DS has a multiple class inheritance model and is able to describe the organizational hierarchy and capture membership information. As a directory service, X.500 and the associated protocols focus on retrieving objects with a UUID; it does not support dynamic relationships and lacks method extension. OMM can coexist with an X.500 directory, it can also be mapped to the LDAP directory and retrieve information through the LDAP interface so that users can access and display the directory objects through OMM.

The *Oval* project by Malone and his coworkers at MIT provides a handy tool for inventing organizations [24,25,26]. It has an object model for constructing organization information and structure. Through user-specified rules, it can process message objects such as notification or customized information flows according to a user’s need. Oval also supports adding hard links between resource objects but not dynamic links.

Other researchers have proposed visual and programming languages for organizational and office systems such as Officeaid-VPE [27], HI-VISUAL [28], M*-
OBJECT [5], Regatta VPL [29], and ORM [6,10]. Officeaid-VPE and HI-VISUAL were limited to the description of single office tasks. They are therefore not adequate for the integration and collaboration across multiple offices in an enterprise. M*-OBJECT and Regatta VPL have a comprehensive process model and an abstract view of organizations; however, the coupling of the process model with the organization model limits their flexibility in organization design. ORM separates the workflow implementation from the organization component, but the role model is still an integral part of the organization model.

7. Conclusion

In this paper, a dynamic organizational information system, the OMM methodology and organization model, along with its system architecture are presented as a comprehensive tool to model roles to support dynamic authorization in E-commerce. The application of the OMM methodology in role resolution of an electronic expense voucher is discussed. Compared to previous efforts [4,9,11,12,16,19,24,27], OMM is similar in having a strong object model and a separation between the organization model and the process model. However, OMM also abstracts the organization model from the role model, thus giving flexibility in complex organization modeling. It is novel in having a dynamic inter-relationship notion that is expressed by using regular expressions over member attributes and contextual variables. This paper shows that the relationship model is essential in supporting access control of cooperative software such as electronic commerce applications, for authentication, authorization and dynamic job assignment. Using virtual links, OMM can model dynamic roles such that policies regarding various operations over the work objects can be defined and maintained. Finally, the explicit life-cycle of the OMM members reflects the dynamic state changes of resources in reality. This provides a handle for better support of organization management and makes task re-routing and optimistic exception handling in an E-commerce system possible in case a resource is absent from its duty.

A Java-based OMM prototype (code name OMM S-25) has been developed at OCT Research Laboratory [30]. With S-25, users can model the different resource types and create resource objects representing various entities in the enterprise. Relationships between the objects are modeled as virtual links using regular expressions. A web interface is provided for users to browse through the enterprise and discover the detailed information and connections of the resources from different point of view. We have applied this methodology to model the organizational infrastructure of a rapidly changing 25,000 people hi-tech company in Silicon Valley; by defining 4 virtual links, over 2,000 graphical models are automatically generated and maintained, and E-commerce applications can be built on top of such a dynamic environment.

8. Acknowledgments

Most of the research on OMM is initiated at the OCT Research Laboratory. The author likes to express thanks to many designers and developers who have contributed to the concept and the implementation of the OMM system. Thanks to George Loizou for his insightful input, the numerous discussions with him on the OMM model have been very helpful. Thanks to Dieter Gawlick for his input on workflow and the requirement of publish-and-subscribe which has greatly impacted my thoughts on the topic. Thanks to my wife for her continuous support and encouragement throughout the entire project.

References


