XGSP-RBAC: Access Control Mechanism based on RBAC Model in Ubiquitous Collaboration System

Kangseok Kim
(School of Computer Science and Information Technology, Inha University, Incheon, Korea
kskim@inha.ac.kr)

Geoffrey C. Fox
(Community Grids Laboratory, Indiana University, Bloomington, IN, USA
Department of Computer Science, Indiana University, Bloomington, IN, USA
School of Informatics, Indiana University, Bloomington, IN, USA
gcf@indiana.edu)

Abstract: With the advances in a variety of software/hardware technologies and wireless networking, there is coming a need for ubiquitous collaboration which allows people to access information systems independent of their access device and their physical capabilities and to communicate with other people in anytime and anywhere. Current virtual conferencing systems lack support for ubiquitous collaboration. As the number of collaborators with a large number of disparate access devices increases, the difficulties for protecting secured resources from unauthorized users as well as unsecured access devices will increase since the resources can be compromised by inadequately secured human and devices. In this paper we address issues related in building a framework for ubiquitous collaboration. First, to make ubiquitous collaboration more promising, we briefly present a framework built on heterogeneous (wire, wireless) computing environment and a set of session protocols defined in XML to provide a generic solution for controlling sessions and participants' presences in collaboration. Second, to provide a solution for controlling accesses to resources, we present a flexible and fine-grained access control mechanism based on Role Based Access Control model, a generic moderator-mediated interaction mechanism – XGSP-RBAC (XGSP Role Based Access Control). Finally, we show the experimental results obtained from the practical evaluation of XGSP-RBAC mechanism.

Keywords: XML, Role Based Access control, Ubiquitous collaboration, Mobile device, Virtual conferencing
Categories: H.0, H.5.0, H.5.1, H.5.2, H.5.3

1 Introduction

Collaboration is about interaction among people and between people and resources. With the advances in a variety of software/hardware technologies and wireless networking, there is coming a need for ubiquitous collaboration and access which allows people to access information systems independent of their access device and their physical capabilities and to communicate with other people in anytime and anywhere. Also, with the maturity of evolving computing paradigms and collaborative applications, a workspace for working together is being expanded from locally collocated physical place to geographically dispersed virtual place. Mobile computing paradigm [B'Far, 05] made ubiquitous access possible with the integration

of wireless communication technology in anytime and in anywhere. With grid computing paradigm [Berman, 03, Foster, 01] which is about sharing resources, resources are distributed into workspaces and shared among geographically dispersed collaborators. With pervasive computing paradigm [Saha, 03, Weiser, 91], it is becoming possible to make workspaces virtually suitable for collaborating users in the goal of all the time and everywhere instead of accommodating collaborating users to collocated workspace. We believe from Moore's law [Moore, 65] and our development experience that the computing performance of mobile devices as well as desktop computers will continue to improve and networks' bandwidth will continue to increase. Thus the infrastructure improvements of software, hardware, and networking will make ubiquitous collaboration and access more prevalent and make the vision of Mark Weiser for 21st Century Computing [Weiser, 91] more promising as well in the future.

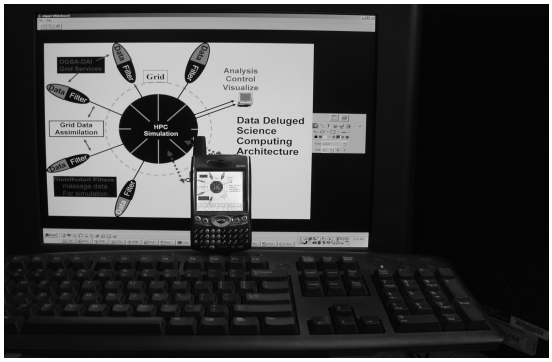

Figure 1: A Screenshot of Collaboration between Desktop and Cell Phone

The following scenario illustrates the needs of ubiquitous collaboration and access, and motivates the research issues described in this paper. Researchers in Community Grids Lab (CGL) [CGL, 01] at Indiana University often travel to attend offline real conference in a shared location. Students in CGL sometimes need to discuss with researchers. Researchers have to find a virtual conferencing system compatible with a conferencing system in CGL to discuss with students while traveling. Further, roaming researchers may have to find a place in which a compatible system is located. As this occurs, an integrated collaboration system, which combines heterogeneous virtual conferencing systems into a virtual conferencing system, will facilitate collaboration between the researchers and students. Virtual conferencing systems over Internet are rapidly increasing. Also, with increasing mobile devices, to integrate diverse mobile devices into a globally virtual conferencing system is becoming increasingly important. Current virtual conferencing systems lack support for ubiquitous collaboration and access. Figure 1 shows an example screenshot of collaboration between desktop device and cell phone.

Students in CGL are going to have a session for their colleague's research presentation. Some students join the presentation session in a shared conference room of CGL and others join at remote locations by using CGL's conferencing

collaboration tool – Global-MMCS system (Global Multimedia Collaboration System) [Fox, 03, Global-MMCS, 03]. The presenter starts her presentation with the conferencing collaboration tool. During her presentation, she may use an application like shared whiteboard to discuss design issues of the research which she is doing on grid computing. In shared workspace with the application, people in offline shared real room see the same whiteboard canvas, while people in online virtual room see their own canvases. Each student in the online virtual room has their own canvas and a set of interfaces to the shared whiteboard application but they see the same results (or views) as others do. Her advisor, researchers, and colleagues in CGL want to make comments on her research by directly manipulating the shared application showing the same views among participants in her research presentation session. Thus, the presenter needs to control their accesses to the shared application by enforcing who is allowed to access the application, and the conditions under that the privileges for the use of the application occur to restrict unauthorized access for the protected application.

This paper is organized as follows. Section 2 presents research issues. We discuss related works in Section 3. Section 4 briefly presents the architecture of collaboration framework and the implementation of it, and describes XML based General Session Protocol (XGSP). Section 5 presents a generic moderator-mediated interaction (request-response) mechanism – XGSP-RBAC (XGSP Role Based Access Control) for controlling accesses to applications and its supporting architecture integrated into our collaboration framework. Section 6 shows the experimental results obtained from the practical evaluation of XGSP-RBAC mechanism. Finally we conclude with a brief discussion of future work and summarize our findings.

2 Problem Statement

Conference collaboration systems typically provide a group of users with a set of well-defined interactions to access applications and resources, and communications among them. In such collaboration systems a group of users generally work sharing collaborative applications and resources in their workgroups (sessions). The cooperation on the resources shared among a group of users may hence produce new results on the shared resources. Fundamentally collaboration is about interaction among people and between people and resources. The cooperation on the resources shared among a group of users may hence produce new results on the shared resources. On the contrary, security is about restricting unauthorized access to resources and thus it is essential that security of the collaboration environments as well as of collaborative applications running on them is ensured while providing the openness only to users that are authorized to access them. Therefore, difficulties to deal with the conflicting goals of allowing and restricting accesses for resources among a group of users may happen in collaboration environment. The examples of the difficulties include protecting secured computing environments and resources from unauthorized users as well as unsecured remote devices since the environments and resources can be compromised by inadequately secured entities – human, devices, software, data, and so on.

The activities in collaboration system include the interactions for the use of resources as well as for cooperation among a group of users working at remote

locations. The interaction for resources involves not only the use of applications but also the use of hardware devices, software, and data. Some resources in the interaction activities may require authorized access, meaning the resources can be accessed by only authorized users. For the resources an access control policy and a mechanism to enforce the policy should be implemented defining which resources are available, who is allowed to access the resources, and the conditions under that the privileges for the use of the resources occur.

In traditional system such as file system, access rights in access control schemes are usually static permissions that are permanent during the interactive activity in the system [Dommel, 97]. Access control schemes need flexible access rights adapting to the state change of collaborative resources that may be occurred from cooperation in collaboration system. Collaboration system thus needs a scheme to enable collaborating users or collaborative applications to control accesses during their activities at run time.

In collaboration environment collaborating users are generally assigned a role, and collaborative applications have different types of roles which are assigned to a group of users. Access control scheme in collaboration system hence needs fine-grained access control for providing accesses for individual users in group, and for a finer granularity of accesses on individual resources shared in group. In other words, an access control scheme for collaboration environment should allow independent specification of each access right of each user on each protected resource [Shen, 92]. For example, it should allow fine-grained drawing actions and support protection for each of them in whiteboard application.

In this paper we show a moderator-mediated interaction (request-response) mechanism, which uses role entity between collaborating users and collaboration resources for ease of administration, fine-grained access control, and flexible adaptation of collaboration environment's changes.

3 Related Work

In this section we examine existing access control schemes for collaboration system.

3.1 Access Control Matrix

Access control matrix is a scheme that describes current allowed accesses using a matrix. It characterizes the access rights of each subject associated with respect to each object in a system [Bishop, 04]. There are variants of the access control matrix such as access control lists (ACLs) and capabilities [Bishop, 04] enable systems to use more convenient and more optimized mechanisms. An example framework using ACLs is a SUITE [Shen, 92] which is a multi-user editing framework. Shen and Dewan [Shen, 92] extended the conventional access matrix scheme in several ways: the use of collaboration rights, the support of negative rights which is explicit denial of a right, the use of inheritance rules and conflict resolution rules. Another example is a Globus Security Infrastructure (GSI) [GSI] which provides a coarse-grained access control approach and uses a mapping list. The mapping list is used to map user's local account name to DN (Distinguished Name) on the user's certificate. When a user wants to use a service, the mapping list is consulted and the access for

the service is granted or denied depending on whether she or he appears on the list with the correct credentials. An example framework using Capability is a XPOLA (eXtensible Principle of Least Authority) [Fang, 05] which provides fine-grained authorization solution for Grid services to follow the principle of least privilege. Another example is a Community Authorization Service (CAS) [Pearlman, 02, 03] which will be described in section 3.4. The CAS implements the capability scheme using an authorization server called CAS server.

3.2 RBAC (Role Based Access Control)

RBAC model [Ferraiolo, 92, 95, Bishop, 04, Sandhu, 96] is a scheme that describes access rights using the notion of roles predefined in organizations. It characterizes the relationship between users and access right for resources with respect to roles based on job functions in organizations. The relationship includes permission assignment and user assignment; access rights for resources are assigned to roles (permission assignment) and users who are authorized to assume the associated roles are assigned to the roles (user assignment). As RBAC scheme is applied to collaboration system which includes sharing resources and cooperation on them among groups of users, the scheme lacks fine-grained access control for providing accesses for individual users in groups and for a finer granularity of accesses on individual resources. Also collaboration system needs a scheme to enable users or collaborative applications to control access during their activity at run time. To make collaboration system flexible for giving users or their applications authorization to decide access for resources, OASIS [Yao, 01] role-based access control model addresses the issues of role activation and deactivation based on first-order logic which specifies parameters of conditions to determine the activation-deactivations. An example framework using RBAC scheme is PERMIS (Privilege and Role Management Infrastructure Standards) [Chadwick, 02, 03] which will be described in section 3.3.

3.3 PERMIS (Privilege and Role Management Infrastructure Standard)

The Privilege and Role Management Infrastructure Standards (PERMIS) [Chadwick, 02, 03] is a RBAC authorization infrastructure to utilize a scalable X.509 Attribute Certificate (AC) [X.509, 01] based Privilege Management Infrastructure (PMI). The PMI uses AC which holds a binding between a user and her privilege attributes. The ACs are issued to users and a resource gatekeeper reads the privilege attributes in the users' ACs to see if they are allowed to access resources. PERMIS system uses RBAC mechanism based on the X.509 AC for authorization infrastructure.

3.4 CAS (Community Authorization Service)

Community Authorization Service (CAS) [Pearlman, 02, 03] implements the capability scheme using an authorization server called CAS server. Resource providers establish a trust relationship with the administrator of a community served by CAS and then delegate a fine-grained access control policies to the administrator. A user issues a request to the CAS server in her community. The CAS server issues a proxy credential with capabilities (access right lists granted to access resources) to the user. Then the user uses the proxy CAS credential to access the resources. The example resource that can be accessed through CAS is GridFTP [Allcock, 02]. The paper

[Pereira, 06] implements RBAC scheme using the CAS server. Since centralized characteristic of the CAS server, CAS service may have scalability problem in very large VOs (virtual organizations) [Foster, 01] which form a group of users and a collection of resources shared among them, and also single point of failure problem of the CAS server.

4 XGSP Collaboration Framework Architecture and XML based General Session Protocol (XGSP)

Collaboration framework is a basic structure to hold consistent view or information of users' presence and sessions together, and to support diverse collaborative applications to collaborators joining in a conference at remote locations. It also has a capability that allows a user to join a conference using networked heterogeneous (wire, wireless) computing devices anytime and anywhere and to use collaborative applications in the conference. It is important to users joining a conference that it seems to be in offline real conference room even when using heterogeneous computing devices at remote locations. It is typical today and will be more typical in the future that all users can access information independent of their access devices and physical capabilities anytime and anywhere. To maintain consistent information of presences and sessions in a conference, we use a request (query) and response (dissemination) mechanism that requires a user to inquire queries (request event messages) to a chairperson node (conference chairperson) and a conference manager in order to engage in presence and various collaboration activities, and the chairperson node and conference manager to disseminate the queried information to all the participants through our messaging and service middleware – NaradaBrokering [NaradaBrokering, 01, Pallickara, 03, 05]. A set of protocols are defined in section 4.7 for maintaining consistent collaboration state information among participants in conference collaboration. As shown in Figure 2, the collaboration framework is structured as three layers and six major components: control manager, session / membership control manager, access / floor control manager, policy manager, request and reply event message handlers, and communication channel. We describe the components in turn.

4.1 Control Manager

A control manager is an interface component located between sessions and managers in collaboration framework for providing conference management services such as presence, session, and access and floor control managements for participants in collaboration. Presence of participants, creation/destroy of sessions, and activation/deactivation of actions to access resources are serviced through this manager into each of control management services. The control manager also has factories for all kinds of applications, and hence can create new application instances and invoke, start, and destroy them.

4.2 Session and Membership Control Manager

This manager manages information about who is currently in the conference and has access to what applications, and which sessions are available in the conference. The

session and membership control manager has a set of control logics that are used to manage presences of and connectivity among collaborating users in collaborating workgroups, and organize the workgroups. The control logics communicate through a set of predefined protocols (session control protocols) for streaming control messages to exchange presence information of collaborating users and state information of various collaborative sessions. The session control protocols account for policy, presence, session creation, initiation, teardown, and so on. To describe presences, connectivity, and states of sessions, XML is used as a protocol definition language of the session and membership control. The XML based General Session Protocol (XGSP) is described in section 4.7 briefly and in more detail in [Wu, 04].

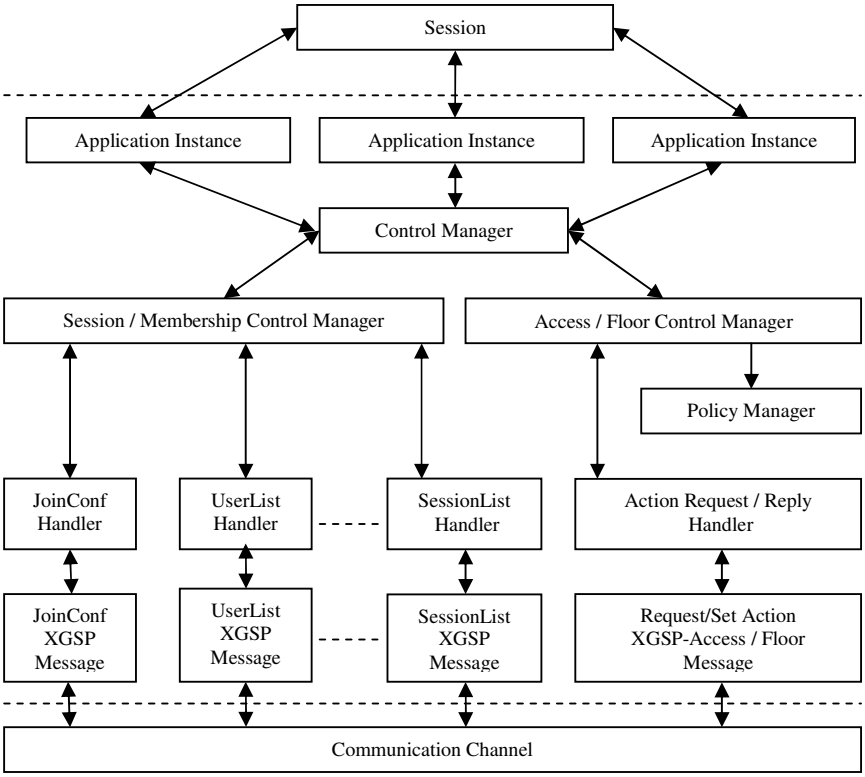

Figure 2: Collaboration framework architecture consists of three layers (collaborative applications, managers, and communication service) and six major components.

4.3 Access and Floor Control Manager

The access and floor control manager component in the collaboration framework is responsible for handling accesses to collaborative applications through the request and reply event message handlers which are one of components in the framework. A user requests an access to use resources like collaborative applications to a

chairperson or moderator through a request event message handler. The chairperson or moderator responses a decision (grant, deny, or queued) to the requesting user who wants to access resources through a reply event message handler. The chairperson or moderator also broadcasts the decision to make the change of access state to each resource globally visible to all the participants in a session. A GUI (Graphic User Interface) on the framework, which is used to display access state information for resources, is used to request accesses to resources. Within the access and floor control manager, policies are read from a file, a request is validated through a policy manager and one of classified access types is returned into the manager through an access type decision service. With the returned access type, a chairperson or moderator makes a decision and the decision is dispatched to the requesting user. Also the decision is broadcasted into each node to update the access state information for the resource. The XGSP Access control mechanism (XGSP-RBAC) is described in section 5 in more detail.

4.4 Policy Manager

Access control policy is written in XML and put into the conference manager which resides on web server running on tomcat for globally consistent use. When a new user joins a conference, the conference manager pushes the policy into the node (or host) of the new user as a stream message, and the policy is stored in local policy store (a file) of the node during joining (connecting) in the conference. The policies describe which roles (users in them) in collaboration are allowed to perform which actions on which target applications. As a request event message for accessing applications arrives, the policy manager pulls the policy from the policy store. The policy manager is responsible for validating the request event messages based on the access control policy pulled from a local policy store.

4.5 Request and Reply Event Message Handlers

An event message handler is a subroutine that handles request and reply event messages. The control manager manages the associations between incoming and outgoing event messages with each of event message handlers. According to the associations, generated outgoing (request) event messages are first processed by the associated request event message handlers in each node (or host). Incoming (reply or response) event messages are also serviced by the associated reply/response event message handlers. The messages are sent to a broker (messaging and service middleware) via the communication channel shown in Figure 2. The broker disseminates the messages to other participants connected to the collaborating workgroup.

4.6 Communication Channel

The communication channel is responsible for controlling interactions among participants and communications among collaborative applications. The channel uses topic-based publish-subscribe mechanism that defines a general API for group communication. The API for the topic-based publish-subscribe mechanism is used as an interface for group communication of sessions in a conference and between collaborative applications and a broker. In the topic-based publish-subscribe mechanism, the topic information contained within messages is used to route the

messages from publisher to subscriber. The topic information has two kinds of naming schema: a name separated simply by slash("/") strings like /XGSP/Conference-ID/Application-Session-ID can be used and another naming schema can be described using a set of tag=value pairs, a set of properties associated with the message, verbose text, or XML. The messages containing topic information are sent to a broker through the communication channel. And the messages are disseminated through router nodes, referred to as brokers to subscribers which registered a subscription to the topic.

4.7 XML based General Session Protocol (XGSP)

Collaboration can be defined as interaction for cooperation on shared resources among people working at remote locations. The interaction in collaborative computing requires a simple and universal access means and mechanism for people to easily access information or to conveniently communicate with other people. Interactions and cooperation for collaboration can be generally provided through the unit of conference and sessions. A conference is composed of a set of sessions, where a session means online workgroup of collaborating users working with sharing various collaborative resources. A conference needs control logic to maintain state information among sessions and presence information among participants in a conference. The control logic is used to manage presences of and connectivity among collaborating users in the online workgroup (session), and organize the online workgroups (sessions or conference). The control logic needs a protocol for streaming control messages to exchange presence information of collaborating users and states of various collaborative sessions. To describe control logics of presences, connectivity, and sessions' states, we use XML as a protocol definition language of session control. The XML based General Session Protocol (XGSP) [Wu, 04] is a protocol for streaming control messages written in XML to provide various collaboration sessions in a conference for users according to the presences and connectivity. The session control protocol account for policy, presence, session creation, initiation, teardown, and so on. The details of conference, session, and presence management protocol are described in [Wu, 04].

5 XGSP-RBAC (XGSP Role Based Access Control)

The basic idea behind RBAC [Ferraiolo, 92, 95, Bishop, 04, Sandhu, 96] is the notion of role used as an intermediate entity between users and protected resources. The intermediate entity – a role is assigned to a group of user with which collaboration is associated and is assigned a set of access rights to perform operations on resources in the collaboration. XGSP-RBAC uses the concept of the role as an intermediate control entity between collaborating users and collaboration resources. The XGSP-RBAC provides effectiveness with respect to ease of administration, flexible adaptation to the state change of collaboration resources, and fine-grained access control. It uses XML for policy specification as well.

- ✓ Collaboration roles in XGSP-RBAC are a representation to categorize users joining a conference for collaboration. The roles are based on the users'

privileges and devices' capabilities allowed to manipulate the protected resources in the collaboration.

- ✓ In XGSP-RBAC collaboration, the use of role simplifies the administrative management of access rights for resources since a user can easily be reassigned from one role to other roles without modifying the access control policy. Also, the use gives an administrator flexibility adapting to the change of collaboration environment by allowing a user to take multiple roles simultaneously, assigning new roles to the user, or revoking roles from the user. The XGSP-RBAC scheme provides flexibility adapting to the state change of collaborative resources that may be occurred from cooperation among collaborators at run time in collaboration system.
- ✓ A fine-grained access control for the instance of individual resource is used in collaboration. For example, the actions (access rights) to perform operations on the whiteboard which is a shared application in our collaboration are fine-grained into line, rectangular, oval, pen (a series of contiguous lines) drawings, and so on. Also, a fine-grained access control on individual user in a role can be used. For example, a moderator in collaboration can give access rights for resources to a specific user in a role (a user in a workgroup or in a session) since XGSP-RBAC uses moderator-mediated interaction mechanism. But a moderator needs to give a user the least of privilege needed in collaboration session (principle of least privilege [Bishop, 04]) in the fine-grained access control on individual resource.
- ✓ To specify access control policies and exchange request-response messages of access control for resources between normal user node (request node (or host)) and moderator node (response node), XML is used for streaming request-response messages of access control for resources and for specification of policies since it is easy to understand and use with pre-existing industry standard parsers.

XGSP-RBAC is a role based access control mechanism mediated by a moderator in collaboration, where policy is written in XML and stored in a local policy store – a file residing in each node (or host). The policy is dispatched to each node from the conference manager shown in Figure 7 at joining time in a conference. The XGSP-RBAC architecture is composed of four major components: activation/deactivation service, access control decision service, local policy store, and authentication and secure delivery service. At request time for accessing collaboration resources, a user sends a request message in XML stream to moderator node (or moderator). XGSP-RBAC mechanism makes its decisions according to the policy read from the policy store of moderator node at decision time. If the request is validated by the access control decision service, then a moderator in collaboration grants or denies the requesting user's access to the collaboration resources. At decision response time, a moderator responds a decision to the requesting user in XML stream as well.

The following subsections provide protected resource access policy, collaboration role and fine-grained action definition, secure end-to-end delivery of messages for authentication and encryption-decryption of messages, and the architecture of XGSP-RBAC integrated into our collaboration framework.

5.1 XGSP-RBAC Policy

XGSP-RBAC policy specifies which roles (users in them) in collaboration are allowed to perform which actions on which target resources. The XGSP-RBAC policy (resource access policy) is described in terms of roles, protected resources (collaborative applications), and fine-grained actions permitted on the protected resources. Also, an access type is placed on the resource access policy based on the characteristics of collaborative applications. The access type in our collaboration means rules categorized to access collaborative applications. The access type includes shared, exclusive, released, and implicit types. The access type shared means the fine-grained action in a collaborative application can be shared among collaborating users. The access type exclusive means the fine-grained action is not able to be shared among collaborating users. It hence means a floor control [Dommel 95, 97] mechanism has to be able to provide the floor for the action on the shared application for only one participant in the synchronous online session at a time. The access type release means the action with the type can be used for releasing the action a user holds. For example, in our whiteboard application, the action slave has the access type released. The access type implicit means the action with the type can be granted without the mediation of moderator according to the resource access policy. In the whiteboard application, a moderator has actions with the access type. The grant mechanism with this type is similar to the capability scheme of access control matrix holding a capability token (a set of access rights). In our collaboration system, a role is a collection of representations capable to operate on collaborative applications with heterogeneous computing devices. We used chairperson, moderator, non-mobile users (desktop users), mobile-users (cell phone users), and chess players (white player, black player, and observers) as a set of example roles in our collaboration system. Actions are a set of operations permitted on the protected resources. The type of actions is dependent on the type of resources and the capabilities supported by heterogeneous computing devices (desktop and cell phone). For example, the role non-mobile-user (desktop user) can have actions including capability moving drawing objects (line, rectangular, oval, pen) in our shared whiteboard application with image annotation while the role mobile-user (cell phone user) is not able to have the capability moving the objects because the whiteboard application on mobile device (cell phone) does not support the capability. Note that we did not define the role hierarchy policy in the XGSP-RBAC policy and implement the mechanism to enforce the policy, and hence we will design and implement it with fault-tolerant role delegation issue as a next phase in future work. The example XGSP-RBAC policy, used in our collaboration system, is shown in Figure 4.

A user has to join a conference by sending her initial presence in join-conference XML stream to a moderator node and a conference manager before the user can establish a session in the conference on the conference manager in order to receive policies for setting session policies up and accessing to resources. The conference manager informs a XML stream binding a policy that is used for requests of protected resources and then she can be an active member of the predefined role assigned in the collaboration. An example of the policy binding stream is shown in Figure 3.

```
<ReplyPolicy>
<ConferenceID>ourtestroom</ConferenceID>
<User><UserID>kskim</UserID><UserName>kangseok-kim</UserName></User>
<Policy><XGSP-RBACPolicy>……</XGSP-RBACPolicy></Policy>
</ReplyPolicy>
```

Figure 3: XML Stream Binding a Policy from Conference Manager showing conference ID, user ID, user name, and resource access policy (XGSP-RBAC Policy).

```
<XGSP-RBACPolicy>
 <ResourceAccesspolicy>
  <RoleName>mobile-user</RoleName>
  <ApplicationRegistries>
   <ApplicationRegistry>
    <ApplicationID>wb</ApplicationID>
    <MainClass>cgl.myprofessor.whiteboard.Whiteboard</MainClass>
    <Actions>
     <Action>
          <ActionName>slave</ActionName>
          <Capabilities>read</Capabilities>
          <AccessType>released</AccessType>
     </Action>
     <Action>
          <ActionName>master</ActionName>
          <Capabilities>read+write</Capabilities>
          <AccessType>exclusive</AccessType>
     </Action>
     <Action>
          <ActionName>line</ActionName>
          <Capabilities>linedrawing</Capabilities>
          <AccessType>shared</AccessType>
      </Action>
           .
           .
      <Action>
          <ActionName>pen</ActionName>
          <Capabilities>pendrawing</Capabilities>
          <AccessType>exclusive</AccessType>
      </Action>
    </Actions>
   </ApplicationRegistry>
  </ApplicationRegistries>
  </ResourceAccesspolicy>
</XGSP-RBACPolicy>
```

Figure 4: An Example of XGSP-RBAC Policy with the Role Name mobile-user and Application Name whiteboard

5.2 Collaboration Role and Fine-grained Action in XGSP-RBAC

Collaboration roles in XGSP-RBAC are a representation to categorize collaborating users joining a conference session for collaboration. The roles are based on the users' privileges and devices' capabilities to manipulate protected shared collaborative applications. In this section we present how collaboration roles used in XGSP-RBAC are represented. For the representation we use functional notion to show the relationship between roles, and action privileges. In role abstraction domain of the function we express the collaboration roles to be assigned to users joining sessions. In action representation domain of the function we express actions permitted to manipulate protected collaborative applications in sessions. The function representation is shown in Figure 5. The definition of the collaboration actions depends on the type of applications. As an example we use shared whiteboard application for the definition of actions in Backus-Naur Form (BNF) below. In BNF we also define collaboration roles and actions as follows.

CollabApp ::= WB
CollabRole ::= Chairperson | Moderator | Non-mobile User | Mobile User
CollabAction ::= Master | Slave | Line | Rect | Oval | Pen | Eraser | Clear | Load | Move

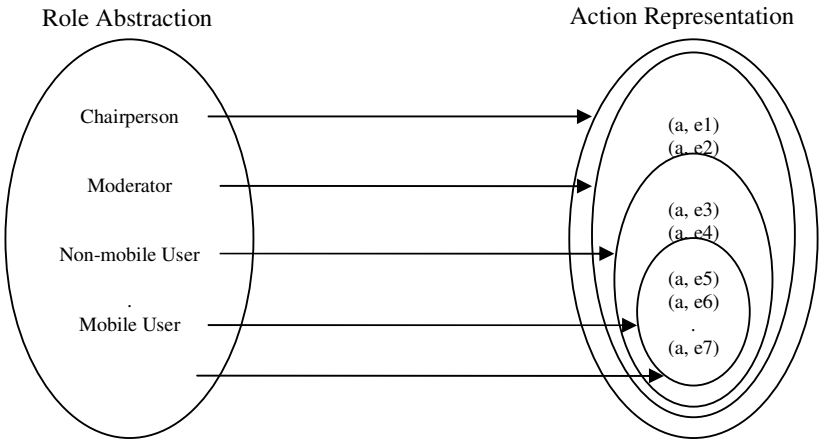

Figure 5: A collaboration action is represented as a pair (a, e) ∈ A x E, where a ∈ A is an application and e ∈ E is the authorized smallest major event defined by a, and A is a set of applications, E is a set of the smallest major events defined by an application in A.

We define fine-grained actions in our collaborative application as the smallest interactive major events (semantic events [Qiu, 05]). For example, in the whiteboard application, drawing a line includes clicking, dragging, and releasing a mouse on the whiteboard canvas. For a user working alone with the whiteboard, user input events (low level events such as mouse click, drag, and release) can be interactive major events between the user and whiteboard application. For users working with others sharing the application, the smallest major event means "drawing a line" (semantic

event) and the user input events will then be an event data (mouse click – the first point of the line and mouse release – the second point of the line). CGL built a shared SVG (Scalable Vector Graphics [SVG]) browser and a collaborative chess game application with SVG [SVGArena 03, Qiu, 03, 05]. In the collaborative chess game application, the smallest major events are to click on an object, to move, and to release the object during moving the object. After the completion of each move (as the mouse is released), the semantic event (moving an object) is dispatched to another player as the smallest interactive major event. Then the user input events will be an event data for moving an object in the chess game affecting the chess board (view-sharing) of another player as well as observers. Therefore, the major events can be different according to the types of applications. The fine-grained action in our collaboration means an interactive smallest major event affecting the shared view (or result) among users in collaboration.

5.3 Secure and Authorized End-to-End Delivery of Messages

In this section we present a security framework [Pallickara, 06] for secure and authorized end-to-end delivery mechanism of messages between entities (publishers and subscribers) in our messaging system based on publish-subscribe paradigm. The messages delivery for communication between the entities is based on the knowledge of topic. Publisher publishes messages over the topic while subscriber registers a subscription to the topic. The capabilities for creation, advertisement, discovery, and restriction of topics are provided by Topic Discovery Node (TDN) [Pallickara, 05] which is regarded as a specialized node in the system. Topic owner creates and advertises topics, and enforces constraints related to the discovery of the topics through the TDN. The TDN advertises the signed topic which is regarded as a secure topic in the system. Publisher encrypts the content payload of a message with the secret key that is retrieved from Key Management Center (KMC) [Pallickara, 06] and signs the encrypted payload involving computing the message digest of it and encrypting this hashed value with private personal-key. Also the publisher signs signed-payload with a secret token that is generated from KMC. An authorized subscriber verifies the signature to ensure the message's integrity and decrypts the encrypted payload with the previously distributed secret key.

As shown in Figure 6, the security framework is structured as five major components: Certificate Authority (CA), Topic Discovery Node (TDN), Key Management Center (KMC), publisher and subscriber. We describe the components in turn.

5.3.1 Certificate Authority (CA)

A CA is responsible for issuing certificates to entities and managing revocation lists pertaining to compromised entities within our messaging system. The CA notifies brokers and KMCs within the system about any additions to the revocation lists.

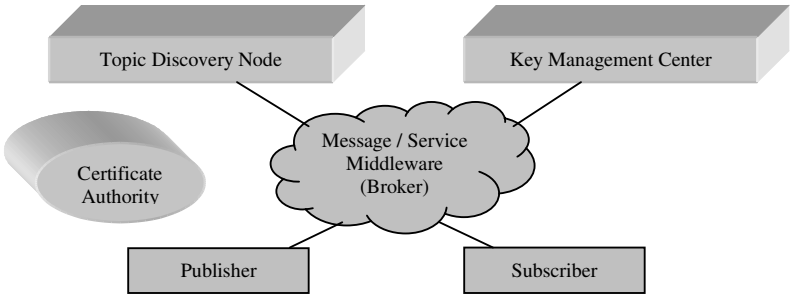

Figure 6: The security framework consists of five major components: Certificate Authority (CA), Topic Discovery Node (TDN), Key Management Center (KMC), Publisher and Subscriber

5.3.2 Topic Discovery Node (TDN)

This node [Pallickara, 05] provides topic discovery and creation scheme for the creation, advertisement, and authorized discovery of topics by entities within our messaging system. Through this node, topic creators can advertise their topics and enforce constraints related to the discovery of the topics.

5.3.3 Key Management Center (KMC)

A KMC [Pallickara, 06] is a specialized node within the system which is responsible for managing information pertaining to secure topics. The KMC generates secret symmetric key for encrypting-decrypting the content payload of messages and security token for establishing entity's rights and duration of them over a secure topic. Also this maintains the list of authorized entities and information related to the entities.

5.3.4 Topic Publisher

Publisher encrypts the content payload of message with the secret key that is received from KMC. The publisher signs the encrypted message and security token together by computing the message digest of the encrypted content payload and then encrypting this computed message digest with its private key. After performing the procedures, the publisher disseminates the message through our messaging system.

5.3.5 Subscriber

Subscriber includes the security token related to the secure topic in its subscription request. Through verifying header and payload signatures of received message and decrypting the message, the subscriber consumes the message.

5.4 XGSP-RBAC Architecture

As shown in Figure 7, the XGSP-RBAC manager integrated into our collaboration framework is structured as four major components: activation/deactivation service, access control decision service, local policy store, authentication and secure delivery service. We describe the components in turn.

5.4.1. Activation / Deactivation Service

When a user requests an action for accessing a protected resource in a session, the request is transformed into a XML stream as shown in Figure 8 and the XML stream is sent to a moderator node through a broker from the communication channel of the request node. Then, the request from the request node is passed to the access control decision service in the access/floor control manager of a moderator node through the action request/reply handler, shown in Figure 2 to ask if the request action is allowed to perform an operation on the requested resource. The following two streams show the action request and grant decision response stream between a request node and a moderator node.

- **Access Request Stream**

A list of actions available for accesses of protected resources in a session is represented with actions which other active users currently hold in the access control GUI of each node. An example GUI is shown in Figure 10. The human-computer interaction with the GUI transforms the access request of a user to perform an operation over a protected resource into a XML stream. The following example XML stream in Figure 8 transformed from the human-computer interaction enables a user (user id: kskim) to request an action (action: pen) over a protected resource (application: whiteboard) in a session (application session ID: NewSession).

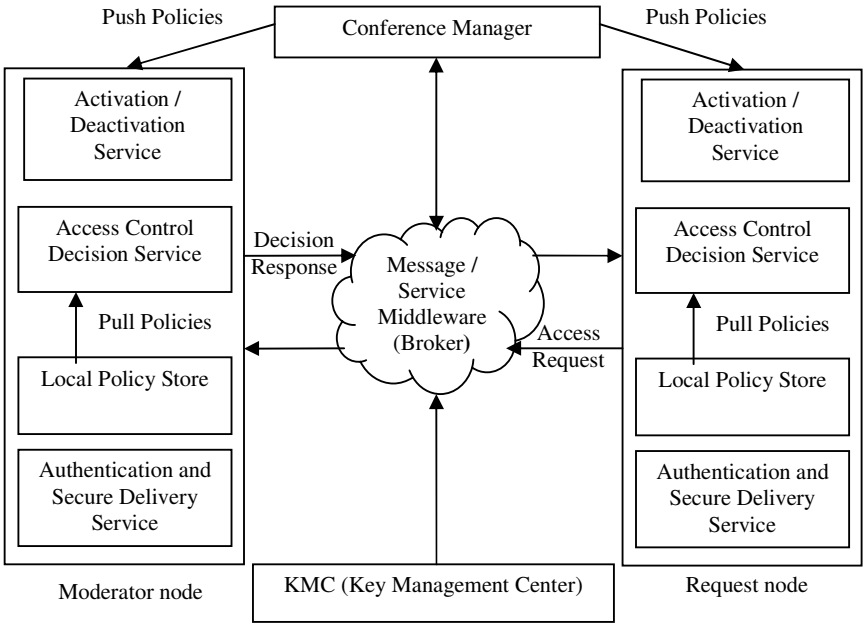

Figure 7: XGSP-RBAC manager integrated into collaboration framework is structured as four major components: activation/deactivation service, access control decision service, local policy store, authentication and secure delivery service.

<RequestAction>
<AppSessionID>NewSession</AppSessionID>
<UserID>kskim</UserID>
<ActionDescription> pen</ActionDescription>
</RequestAction>

Figure 8: Action Request XML Stream

- **Access Grant / Deny Stream**

To check the access privilege of a user over a protected resource, 3-tuple <role name, protected application name, request action name> is consulted in the access control decision service of moderator node. If the role of the requester is allowed to perform the request action according to the resource access policy in the XGSP-RBAC policy, then the request action to access the protected application is granted. Otherwise, the request action is denied. The XML stream in Figure 9 enables a user (user ID: kskim) to execute the request action (action: pen) over a protected resource (application: whiteboard) in a session (application session ID: NewSession). Then, the granted action with the name of the user is represented in the access control GUI of each node as an active action of the user in the session. An example GUI is shown in Figure 10.

<SetAppAction>
<AppSessionID>NewSession</AppSessionID>
<UserID>kskim</UserID>
<ActionDescription> pen</ActionDescription>
</SetAppAction>

Figure 9: Grant Decision Response XML Stream

Figure 10: An Example GUI on Cell Phone

5.4.2 Access Control Decision Service

Policy manager in collaboration framework shown in Figure 2 reads the XGSP-RBAC policy from a local policy store, e.g. a file. The requested action is validated against the policies in the XGSP-RBAC policy read from the policy store. The validation is to check if the action is allowed for the role assigned to the user and for the resources considering all the conditions specified within the resource access policy. If the request is invalid, it is denied. If the request is valid, access type decision service returns an access type value to the access control decision service. The access control decision service makes a decision based on the returned access type value. The decision from the service is passed to moderator. Then the moderator makes a decision on the request. The decision is transformed into a XML stream as shown in Figure 9 and the XML stream is sent to the request node through a broker from the communication channel of moderator node.

5.4.3 Local Policy Store

When a user joins a conference, the conference manager shown in Figure 7 sends a XGSP-RBAC policy to the user by the XML stream as shown in Figure 4. The policy is stored in a file residing in the user's node. This ensures that the policy is up-to-date and consistent among collaborating users. Note that our mobile device, Treo600 [Treo 600] cell phone, does not support writing the policy into itself. The phone then throws a security exception. Thus we held the policy as a string during an online session.

5.4.4 Authentication and Secure Delivery Service

As described in section 5.3, this service encrypts the content payload of decision response message with the secret key that is received from KMC. This service signs the encrypted message and security token together by computing the message digest of the encrypted content payload and then encrypting this computed message digest with its private key. After performing the procedures, a moderator node disseminates the encrypted decision through a broker. The request node consumes the decision response from moderator node through verifying header and payload signatures of received decision response message and decrypting the message.

Note that we did not implement the encryption mechanism of messages for roaming users with cell phone. In future work we will design and implement the authentication service for users joining a conference during roaming with cell phone devices, and the encryption service of messages sent to and from the cell phone devices.

6 Performance and Analysis

In this section, we discuss an experiment with our collaborative application built in heterogeneous (wire and wireless) computing environment to show the viability of XGSP-RBAC mechanism. The main purpose of the experiment is to identify key factors that influence the performance of XGSP-RBAC mechanism comparing overheads incurred from wired-networked environment with those incurred from

wireless-networked environment. In the experiment, we measured mean network transit time (request-response time), mean waiting time in a queue and mean access control decision service time in a moderator node involved in performing communication (an access request for resources and a decision response) between the request nodes and response node (moderator node) for mean interarrival time among access requests in heterogeneous networked environments over a variety of locations.

In the experiment, we utilized two desktop devices, one cell phone and one broker. The collaboration framework on cell phone and desktops is located in Community Grids Lab at Indiana University. The broker ran on a 2.4 GHz Linux with 2 GB RAM located in Community Grids Lab at Indiana University, a 1.2 GHz Linux with 8 GB RAM located in NCSA (National Center for Supercomputing Applications) at UIUC (University of Illinois at Urbana-Champaign), and a 1.2 GHz Linux with 8 GB RAM located in SDSC (San Diego Supercomputer Center) at UCSD (University of California at San Diego). The experiment results were measured from executing collaboration framework and the shared whiteboard application built on the framework running on Palm OS 5.2.1H Powered Treo600 [Treo 600] cell phone platform with 144 MHz ARM Processor and 32MB RAM connected to cellular network, and running on Windows XP platform with 3.40 GHz Intel Pentium and 2 GB RAM and Windows XP platform with 3.40 GHz Intel Pentium and 1 GB RAM connected to Ethernet network respectively. The application codes on the cell phones are written in J2ME (Java 2 Micro Edition) [J2ME] and the application codes on the desktops are written in Java 1.5. A conference managing server (conference manager) is operated as an apache web server. The XML activities on non-mobile (desktop) devices are parsed by and handled with JDOM [JDOM] that is a Java implementation of Document Object Model (DOM). The XML activities on mobile devices (cell phones) are parsed by and handled with kXML [kXML] that is a J2ME implementation of DOM. The following subsections show baseline performance result, experimental scenario, overhead timing considerations, and analysis about the performance measurements.

6.1 Baseline Performance Result

In this section we show the baseline performance results of network (wire, wireless) used for communication between our messaging/service middleware (broker) and collaboration framework built on cell phone and desktop devices. Note that the results are not to show better performance enhancement but to quantify the network performance of wireless cell phone and wired desktop devices for a variety of datasets. The quantified results will be used as a reference of the experimental results of the performance measurements used in the following sections. In our experiment, we measured the round trip time involved in performing communication between collaboration framework and a broker in heterogeneous networked environments over a variety of locations. The experiment result was measured from executing collaboration framework running on Palm OS 5.2.1H Powered Treo600 cell phone platform connected to cellular network, and running on Windows XP platform with 3.40 GHz Intel Pentium and 2 GB RAM connected to Ethernet network. The collaboration framework on cell phone and desktop is located in Community Grids Lab at Indiana University. The broker ran on a 2.4 GHz Linux with 2 GB RAM located in Community Grids Lab at Indiana University, a 1.2 GHz Linux with 8 GB

RAM located in NCSA (National Center for Supercomputing Applications) at UIUC (University of Illinois at Urbana-Champaign) and a 1.2 GHz Linux with 8 GB RAM located in SDSC (San Diego Supercomputer Center) at UCSD (University of California at San Diego). Figure 11 and 12 show the round trip time to transfer bytes data between collaboration framework and a broker through wired and wireless network respectively including the corresponding execution time of the broker. As the size of data increases, the time for transferring the data increases as well, as shown in the figures. Note that where the results in Figure 11 are in the range of only milliseconds, the results in Figure 12 are in the range of seconds. This measurement results will be used as a baseline for the performance measurements in the following sections.

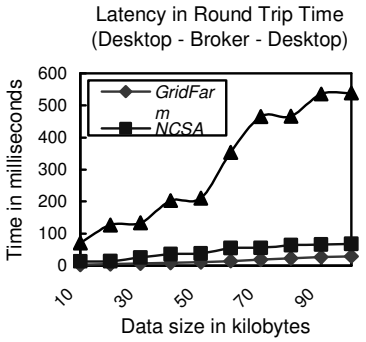

Figure 11: Latency in Round Trip Time between Desktop and Broker

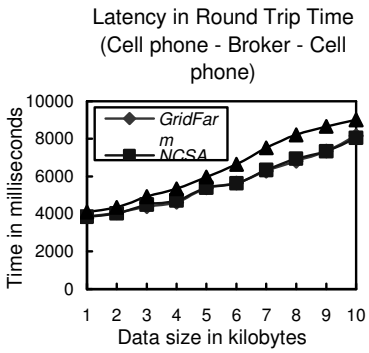

Figure 12: Latency in Round Trip Time between Cell phone and Broker

6.2 Experimental Scenario

Our experiment is carried based on the XGSP-RBAC mechanism which is described in section 5. The access request for resources from a request node and the decision response from a moderator node in the experiment involve the XML streams in Figure 8 and Figure 9 respectively. The experiment is also carried with the simulation program which is behaved by Coloured Petri-nets (CP-nets) [Jensen, 97]. The simulation program uses the exponential function provided by the CP-nets to generate access requests with pre-known mean interarrival time. The access request arrival times form a Poisson process since the interarrival times of the requests are independent random variables with exponential distribution with pre-known mean interarrival rate. In our experiment, we suppose the requests randomly arrive with the pre-known arbitrary mean interarrival rate. The experimental scenario overview is depicted in Figure 13. Note that we did not use the decision behavior of a moderator (human) since the behavior of a human does not reflect the consistent reaction in time that may affect the latency of requests waiting in a queue. The decision result from the access control decision service will thus be directly sent to request nodes without the decision interruption of a moderator. We discuss the overhead costs in the next subsection and how these affect XGSP-RBAC mechanism as involved with cell

phone devices since cell phone devices are sensitive to the network delay as shown in Figure 11 and Figure 12.

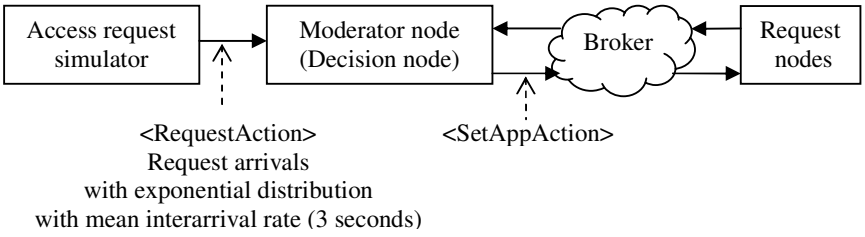

Figure 13: Experimental Scenario Overview

6.3 Overhead Timing Considerations

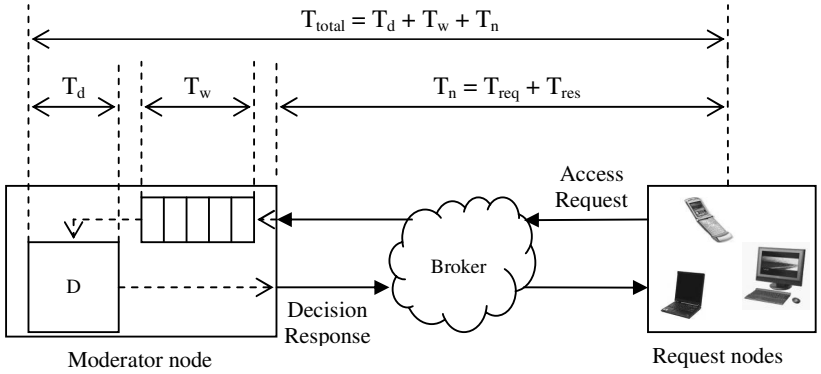

Figure 14: Total latency = Decision time (T_d) + Waiting time (T_w) + Network transit time ($T_n = T_{req} + T_{res}$), where D means an access control decision service

Figure 14 shows a breakdown of the latency for serving a request. The cost in time for XGSP-RBAC mechanism has three primary overheads.

- Transit cost ($T_n = T_{req} + T_{res}$) – The time to transmit an access request (T_{req}) to and receive a decision response (T_{res}) from moderator node.
- Access control decision service cost (T_d) – The processing time to make a decision on an access request for resources at moderator node. This cost includes reading a XGSP-RBAC policy from a file and validating access requests from the policy.
- Waiting cost (T_w) – The time between arriving at a queue and leaving the queue (being served by the access control decision service) at moderator node. The queue is implemented as FIFO (First-In, First-Out) order. The arrival of new request is modeled as Poisson processes with arrival rate λ where the interarrival times between interarrival requests are independent random variables with

exponential distributions with mean interarrival rate $1/\lambda$. The arrival rate λ means the average number of arrivals in unit time. To get independent random variables with exponential distributions with some mean interarrival rate in terms of the arrival time variable of new request, we simulated the exponential distribution of arrival times with an automated simulation tool [CPN]. The simulation tool randomly generates independent new access requests with an arbitrary mean interarrival rate which is already known before the simulation of the new requests' arrival.

Examining overhead costs and total cost, we measured the mean overhead cost for 100 access requests in heterogeneous networked environment over a variety of locations. The results are shown in Figure 15 with the mean completion time of a request.

6.4 Experimental Result and Analysis

In this section we present an experimental result that we have measured to analyze the overheads incurred from controlling fine-grained accesses in XGSP-RBAC mechanism. The simulator generates new access requests on behalf of users on request nodes. The access request generation process follows an exponential distribution. The generated request events, according to the order delivered from the simulator, are stored in a request queue. The experiment is run through the mean request interarrival time (3000 milliseconds) which is an average interarrival time between two successive requests issued by the simulator.

Figure 15 depicts mean completion time of a request vs. mean request interarrival time for three different network combinations involved in our collaboration over three different locations: collaboration using only desktop devices (wired network), collaboration using only cell phone devices (wireless network), collaboration using desktops and cell phones together (wired and wireless network). The comparison shows when cell phone devices using wireless network are involved in our collaboration, the mean completion time of a request is increased since the wireless network has high latency. In the case of the use of cell phone, we may need to make the granularity of fine-grained actions larger to reduce the wireless network overhead. The shared whiteboard application uses fine-grained actions with the smallest major events as described in section 5.2. When a user requests an image loading action, it may be natural to simultaneously request it with some drawing actions. This natural request with larger-grained action can improve response (delay) time of a request but decrease the amount of concurrency and introduce complexity. The degree for granularity is a balance between responsiveness and concurrency [Bernstein, 87] and between responsiveness and simplicity. Also, without user's point of view [Greenberg, 94] for the granularity of actions, unnatural granularity may violate the principle of least privilege because it may give a user more privilege than needed. The experimental result shows that in future work we need to observe user's behavior with applications in collaboration environment considering responsiveness vs. concurrency, responsiveness vs. simplicity, and responsiveness vs. principle of least privilege.

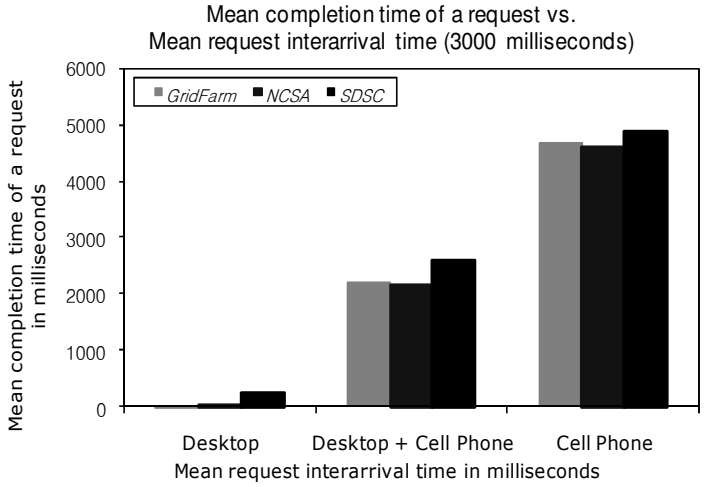

Figure 15: Mean completion time of a request vs. Mean request interarrival time (3000 milliseconds) where Desktop means collaboration using only desktop devices (wired network), Desktop + Cell Phone means collaboration using desktops and cell phones together (wired + wireless network), and Cell Phone means collaboration using only cell phone devices (wireless network)

7 Summary and Future Work

In this paper, we presented the XGSP-RBAC mechanism integrated into our collaboration framework. The XGSP-RBAC uses the notion of role as an intermediate control entity between collaborating users and collaborative applications. The roles in XGSP-RBAC are based on users' privileges and devices' capabilities to allow users to manipulate the protected applications in the collaboration. The use of role simplifies the administrative management of access rights for applications and gives an administrator flexible adaptation to the changes of collaboration environment. Also, XGSP-RBAC mechanism provides flexibility adapting to the state change of collaborative applications that may be occurred from cooperation among collaborators at run time in collaboration system. To specify access control policies and exchange request-response messages of access control for resources, it uses XML because it is easy to understand and use with pre-existing industry standard parsers. Also, fine-grained access control for the instance of individual application as well as for individual user is used.

From our experimental result, in future work we will consider the observation of users' behaviors with a variety of applications in collaboration environment considering responsiveness vs. concurrency, responsiveness vs. simplicity, and responsiveness vs. principle of least privilege.

Also in future work, we will design and implement the authentication service for users joining a conference during roaming with cell phone devices, and the encryption service of messages sent to and from the cell phone devices.

During our experiments with the collaboration framework, one of problems encountered was a failure like network disconnection of a moderator or chairperson node. If a moderator or chairperson node fails or is disconnected, and is not able to recover from the failure for some amount of time, one of participants in collaboration capable of having the role capability of the moderator or chairperson has to be elected. We tested it with an event driven message mechanism. But, when the network connection of a moderator or chairperson node was lost, it did not work since the event messages could not be disseminated in disconnected network. One approach to overcome the problem by exploring different fault-tolerant role delegation mechanism (for example, polling mechanism by heart-beat message between a moderator node and a conference manager) with role hierarchy policy will be considered in future work. We also left in future work support of the role hierarchy policy with the fault-tolerant role delegation mechanism issue.

Acknowledgements

Many thanks to my former colleagues at Community Grids Lab in Pervasive Technology Labs in Indiana University who developed the earlier prototypes of the system described here.

References

[Allcock, 02] W. Allcock, J. Bester, J. Bresnahan, A. Chervenak, L. Liming, and S. Tuecke, GridFTP: Protocol Extensions to FTP for the Grid, Argonne National Laboratory, April 2002.

[Berman, 03] F. Berman, G. Fox, and A. Hey, editors. Grid Computing: Making the Global Infrastructure a Reality, John Wiley & Sons, 2003.

[Bernstein, 87] Bernstein, P., Goodman, N. and Hadzilacos, V. (1987) Concurrency control and recovery in database systems, Addison-Wesley.

[B'Far, 05] Reza B'Far. Mobile Computing Principles: Designing and Developing Mobile Applications with UML and XML, Cambridge University Press 2005.

[Bishop, 04] Matt Bishop. Introduction to Computer Security. Addison Wesley. 2004.

[CGL, 01] Community Grids Lab (CGL), http://communitygrids.iu.edu

[Chadwick, 02] D. W. Chadwick, and A. Otenko. RBAC policies in XML for X.509 Based Privilege Management. SEC 2002, Egypt, May 2002.

[Chadwick, 02] D. W. Chadwick, and A. Otenko. The PERMIS X.509 Role Based Privilege Management Infrastructure. 7th ACM Symposium on Access Control Models and Technologies, 2002.

[Chadwick, 03] D. W. Chadwick and A. Otenko. The PERMIS X.509 Role Based Privilege Management Infrastructure. Future Generation Computing Systems, 2003.

[Chadwick, 03] D. W. Chadwick, A. Otenko, and E. Ball. Role-based Access Control with X.509 Attribute Certificates. IEEE Internet Computing, March-April 2003, pp. 62-69

[CPN,]CPN Tools. CPN Tools Homepage. http://wiki.daimi.au.dk/cpntools/

[Dommel, 95] Dommel H.P. and J.J. Garcia-Luna-Aceves, "Design issues for floor control protocols", In Proceedings of SPIE Multimedia and Networking, (San Jose, CA, USA), pp. 305–16, February 1995.

[Dommel, 97] Dommel H.P. and J.J. Garcia-Luna-Aceves, "Floor Control for Multimedia Conferencing and Collaboration", ACM Multimedia Systems, Vol. 5, No. 1, January 1997.

[Fang, 05] Liang Fang, Dennis Gannon, and Frank Siebenlist. XPOLA: An Extensible Capability-based Authorization Infrastructure for Grids. In 4th Annual PKI R&D Workshop, April 2005.

[Ferraiolo, 92] David F. Ferraiolo and Richard Kuhn. "Role-Based Access Control", Proceedings of the 15th NIST-NSA National Computer Security Conference, Baltimore, MD, 13-16 October 1992.

[Ferraiolo, 95] D.F. Ferraiolo, J. Cugini, D.R. Kuhn (1995) "Role Based Access Control: Features and Motivations", Computer Security Applications Conference - extends the 1992 model.

[Foster, 01] I. Foster, C. Kesselman, and S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of High Performance Computing Applications, 2001. 15(3): p. 200-222.

[Fox, 03] Geoffrey Fox, Wenjun Wu, Ahmet Uyar, Hasan Bulut, Shrideep Pallickara. Global Multimedia Collaboration System in Proceedings of the 1st International Workshop on Middleware for Grid Computing co-located with Middleware 2003, June 17, 2003 Rio de Janeiro, Brazil.

[Global-MMCS, 03] Global-MMCS (Global Multimedia Collaboration System). http://www.globalmmcs.org

[Greenberg, 94] Greenberg, S. and Marwood, D. "Real time groupware as a distributed system: Concurrency control and its effect on the interface," Proceedings of the ACM CSCW Conference on Computer Supported Cooperative Work, October 22-26, 1994. North Carolina, ACM Press.

[GSI] Globus Grid Security Infrastructure (GSI). http://www.globus.org/toolkit/docs/4.0/security

[JDOM] JDOM. http://www.jdom.org/

[Jensen, 97] K. Jensen, Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use, vol. 1, Basic Concepts. Monographs in Theoretical Computer Sciences. Springer-Verlag, 1997.

[J2ME] J2ME. http://java.sun.com/javame/index.jsp

[kXML] kXML. http://kxml.objectweb.org/

[Moore, 65] Moore's Law. 1965. http://en.wikipedia.org/wiki/Moore's_Law

[NaradaBrokering, 01] NaradaBrokering, 2001. http://www.naradabrokering.org

[Pallickara, 03] Shrideep Pallickara and Geoffrey Fox. NaradaBrokering: A Middleware Framework and Architecture for Enabling Durable Peer-to-Peer Grids. Proceedings of the ACM/IFIP/USENIX Middleware Conference. 2003. pp 41-61.

[Pallickara, 05] Shrideep Pallickara, Harshawardhan Gadgil and Geoffrey Fox. On the Discovery of Topics in Distributed Publish/Subscribe systems Proceedings of the IEEE/ACM GRID 2005 Workshop, pp 25-32. November 13-14 2005 Seattle, WA.

[Pallickara, 06] Shrideep Pallickara, Marlon Pierce, Harshawardhan Gadgil, Geoffrey Fox, Yan Yan, Yi Huang. A Framework for Secure End-to-End Delivery of Messages in Publish/Subscribe Systems. Proceedings of the 7th IEEE/ACM International Conference on Grid Computing (GRID 2006). Barcelona, Spain, 28-29 September 2006.

[Pearlman, 02] L. Pearlman, et al., A Community Authorization Service for Group Collaboration. In Proceedings of the IEEE 3rd International Workshop on Policies for Distributed Systems and Networks. 2002.

[Pearlman, 03] L. Pearlman, et al., The Community Authorization Service: Status and Future. CHEP03, March 24-28, 2003, La Jolla, California.

[Pereira, 06] Anil L. Pereira, Vineela Muppavarapu, and Soon M. Chung. Role-Based Access Control for Grid Database Services Using the Community Authorization Service. IEEE Transactions on Dependable and Secure Computing, vol. 3, no. 2, April-June 2006.

[Saha, 03] D. Saha and A. Mukherjee. Pervasive Computing: A Paradigm for the 21st Century. Published by the IEEE Computer Society, Vol. 36, No. 3. pp. 25-31 March 2003.

[Sandhu, 96] Sandhu, R. S., Coyne, E. J., Feinstein, H. L., and Youman, C. E. Role-Based Access Control Models. IEEE Computer 29, 2 (Feb. 1996), pp. 38-47.

[Shen, 92] Shen, H, and Dewan, P. Access Control for Collaborative Environments. In ACM Conference on Computer-Supported Cooperative Work. 1992, p. 51-58.

[SVG] Scalable Vector Graphics (SVG). http://www.w3.org/Graphics/SVG/

[SVGArena, 03] SVGArena. http://www.svgarena.org/

[Treo 600] Treo 600. http://en.wikipedia.org/wiki/Treo_600

[Qiu, 03] Xiaohong Qiu, Bryan Carpenter, Geoffrey Fox, Collaborative SVG as a Web Service, SVG Open 2003 Conference and Exhibition, Vancouver, Canada, July 2003.

[Qiu, 05] Xiaohong Qiu. "Message-based MVC Architecture for Distributed and Desktop Applications" Syracuse University PhD March 2 2005.

[Weiser, 91] M. Weiser. "The Computer for the Twenty-First Century," Scientific American, September 1991.

[Wu, 04] Wenjun Wu, Geoffrey Fox, Hasan Bulut, Ahmet Uyar, Harun Altay. "Design and Implementation of a collaboration Web-services system", Special issue on Grid computing in Journal of Neural Parallel and Scientific Computations (NPSC), Volume 12, pages 391-408 (2004).

[X.509, 01] ITU-T Recommendation X.509 (2001). The Directory: Authentication Framework.

[Yao, 01] Yao, W., Moody, K., and Bacon, J. A Model of Oasis Role-Based Access Control and Its Support for Active Security. In ACM Symposium on Access Control Model and Technology, ACM. Chantilly, VA. 2001.