uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

u Ottawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Jianming Ke

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

Master of Computer Science

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Role-Based and Action-Driven Access Control for Web Services-Oriented E-learning Portal

TITRE DE LA THÈSE / TITLE OF THESIS

A. El Saddik

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Michael Weiss

Jiying Zhao

Gary W. Slater

LE DOYEN DE LA FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES /
DEAN OF THE FACULTY OF GRADUATE AND POSTDOCORAL STUDIES

Role-Based and Action-Driven Access Control for Web Services-Oriented E-learning Portal

Jianming Ke

A thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements for the degree of
Master of Computer Science

Ottawa-Carleton Institute for Computer Science
School of Information Technology and Engineering
Faculty of Engineering

University of Ottawa
Ottawa, Ontario, Canada, K1N 6N5

Your file Votre référence
ISBN: 0-494-11312-X
Our file Notre référence
ISBN: 0-494-11312-X

Abstract

Researches on access control models have been conducted on a diversity of application areas. For those applications that adopt Web services-oriented techniques and portal principles, access control model is one of their important issues. This thesis presents our research on the access control model for a Web services-oriented e-learning portal system, Ubilearn. The access control model has been designed and developed as an access control portal framework to accommodate a variety of services' components within e-learning system. This portal framework integrates two access control approaches: a role-based access control mechanism is to control the role of individual users' access to a classified group of services via its portal interface; whereas an action-driven policy-based mechanism performs the identity services and policy management in a way that identity declarations can be transmitted and shared among each of the services within the system. Finally we describe the implementation of such access control framework in Ubilearn, a Web services-oriented e-learning portal application.

Acknowledgements

I would like to express my sincere gratitude to my thesis supervisor Prof. Abdulmotaleb El Saddik, for giving me an opportunity to pursue my graduate study and providing valuable support whenever I needed. I greatly appreciate his trust, guidance, enthusiasm, and encouragement towards the completion of this thesis.

I am thankful to my colleagues in the Multimedia Communication Research Laboratory, University of Ottawa, for providing me a pleasant working environment.

I would also like to thank my family members, for giving me endless love, care and support throughout my graduate studies.

Finally the financial support provided by the Communications and Information Technology Ontario (CITO) is also highly appreciated.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ACLs	Access control lists
AD-PBAC	Action-driven policy-based access control
B2B	Business to Business
B2C	Business to Consumer
B2Bi	B2B integration
CA	Certificate Authority
COM	Component Object Model
CORBA	Common object request broker architecture
DAC	Discretionary access control
DCOM	Distributed Component Object Model
ERP	Enterprise Resource Planning
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
JMS	Java Message Service
JMX	Java Management Extension
LDAP	Lightweight Directory Access Protocol
LMS	Learning management system
MAC	Mandatory access control
MOM	Message oriented middleware
OMG	Object Management Group
PBAC	Policy-based access control
PDP	Policy decision point
PEP	Policy enforcement point
RBAC	Role-based access control
RHS	Right-hand side
RMI	Remote Method Invocation
RPC	Remote procedure calls
RSBAC	Rule Set Based Access Control
SOA	Service Oriented Architecture
SOAC	Subject-object access control
SOAP	Simple Object Access Protocol
SSO	Single sign-on
TBAC	Task-based authorization control
UDDI	Universal Description, Discovery and Integration
UI	User Interface
UIM	User identity management
UML	Unified modeling language
URI	Uniform Resource Identifier
WSDL	Web Services Description Language)
WS-ELP	Web services e-learning portal
WSRP	Web Services for Remote Portal (Portlet)
XML	Extensible markup language

Chapter 1 Introduction

1.1 Objectives

More and more Web and Web services-based learning management systems (LMS) have been built in order to satisfy the increased demand for sharing learning resources and reusing learning components across the Internet. Indeed the distributed nature of these systems raise numerous concerns related to data management and controlling users access to different applications.

Web services are self-contained, self-described modular applications that can be published, located and invoked across the Internet. Applying Web services technology for LMS enables applications from different domains to communicate via XML-based interoperable interfaces, where web services based applications adopt XML-based protocols to carry and transmit data through, e.g. HTTP over the Internet.

The main access control issue for those Web and Web services-oriented applications and systems is that the system infrastructures are distributed and usually requires constructing specific authentication and authorization model to make its components easily accessible.

Access is the ability to conduct operations of using, changing, and viewing on a computer resource. The ability of access control can be explicitly enabled or restricted through ways of physical and system-based controls. Computer-based access controls can prescribe not only who (people) or what process (applications) may have access to a specific system resource or service, but also the type of the permitted access [31]. Furthermore, we need to set up an access point to allow users to access the diversity of services across different domains.

A Web portal offers a mechanism that integrates a variety of services directly linked to other sites. The portal principle facilitates the creation of a single entrance to a system so that any

user can access multiple, heterogeneous services through this point. Although portal technology presents a way to encapsulate multiple applications in a single Web browser, we need to formulate a feasible solution on the user identity and access management issues for this application model. The typical concerns focus on the abilities to manage users' identity, to authenticate the users accessing applications, to differentiate user accessing to different functional services, and to render those accessible services. Specifically, an access control mechanism combined with role, policy, attribute, rule, conditions, and context extensions would be constructed for the Web services e-learning portal (WS-ELP) so that the combination of user identity management and accessing to services' components can be easily handled.

1.2 Contributions

This thesis proposes two access control approaches for a Web services-based e-learning portal framework. By integrating the role-based portal access control approach and the action-driven policy-based approach, we created and implemented a portal framework to achieve the emerging needs for distributed WS-ELP to provide an access control management in which all accesses to learning objects or services can be controlled (on the basis of the role of the user and other parameters in terms of services, attributes and access modes). Moreover, the action-driven policy-based approach combines role-based and rule-based access control models to fulfill the separation of policy-checking function from other services components, improving the reusability and maintainability of those other services in Ubilearn. We will give further explanations for these two access control approaches.

The major contributions related to the proposed access control model are briefly summarized in the following:

- The thesis analyzes traits of controlling access to the involved components in the Ubilearn system based on the ways of services are executed and accessed, and the way parameters are transmitted. (Ubilearn is a WS-ELP which has been developed at the Multimedia Communications Research Laboratory at the University of Ottawa, Canada.)

- The thesis describes details of analyzing and designing the proposed access control using Unified Modeling Language (UML).
- Based on the analysis and design, an access control portal-based framework has been created to accommodate a variety of services' components and those two access control approaches have been implemented and integrated into the Ubilearn.
- Publication:
    - Kai Wang, Jianming Ke and Abdulmotaleb El Saddik, "Architecture for Personalized Collaborative E-learning Environment" , In Proceedings of Ed-Media 2005, Montreal, Quebec, June 2005.

1.3 Organization of this thesis

The thesis is organized as follows. In Chapter 2, we give background explanation of the research topic and summarize the diversity of the access control mechanisms. After conducting analysis of the architecture and the general requirements in access control for WS-ELP in Chapter 3, we describe the role-based access control portal framework for Ubilearn which integrates required services by using appropriate UML diagram in Chapter 4. In Chapter 5, we describe the implementation of the action-driven policy-based access control mechanism. Finally, Chapter 6 includes a summary and future work.

Chapter 2 Overview of Access Control

2.1 Basic concepts

Typically, we utilize traditional user-id and password for authentication to restrict unauthorized users from accessing some specific services and resources. A major security risk is that passwords are often easy to guess or infrequently changed, creating additional security risks as a result. For any security system, authentication, authorization and access control are the usually mentioned terminologies to achieve the primary goals of confidentiality, integrity and availability [6].

2.1.1 Common used notions

The following terminologies are often used for identity management processes [38]:

- User: Users are people whose access to systems and identity information must be managed.
- Account: An account is the collection of data used by a system to identify a single user, authenticate a user and control that user's access to resources.
- User-ID: On most systems, accounts are uniquely identified by a short string of characters. This is called the User-ID, login ID or login name.
- Privilege: A privilege is the right to do something on a system. Privileges normally relate either to the ability to access services or data, or the ability to use some features.
- Group of Users: A user group is a list of accounts on a system to simplify administration of privileges (i.e., assign privileges to the group, and actively manage just the membership of the group).
- Managed system: A managed system may be an operating system, database or application where users access some features or data, and where user access must be controlled.
- Resource: Resources are data or functions that users access on systems.

- E-Provisioning: The process of setting up accounts and possibly other resources for users on strictly electronic access. Provisioning is usually triggered by a new user registration or a user's role assignment.

2.1.2 Authentication

The term authentication is usually used to denote identity authentication, which is a process of verifying the identities that users are who they claim they are. Users have to provide an identity pair consisting of a user-id and password so that the application can verify whether the user is allowed to access the services offered by the application or not. In addition, authentication represents a technical procedure to ensure that all messages transmitted across the communication network are from the original source without change in transit.

2.1.3 Authorization

Authorization is to make it valid, legal, binding, or official for a user or application to perform certain actions after being authenticated. Authorization is usually used to find out if the person, once identified, is permitted to access services and resources offered by the system. This is usually determined by finding out whether that person is a part of a particular group and has privileges to access specific services and resources requested.

2.1.4 Access control and access matrix

Access Control is an internal protection mechanism that monitors or controls the users or applications (subjects) access to the system's resources or services (objects). Access can be granted or denied on the basis of a wide variety of criteria. Only authorized parties can access services or resources defined by an access control system [19].

The definition of access control usually includes access matrix, a conceptual model that specifies the rights that each subject accesses to corresponding objects. In the access matrix, a row is used for each subject, and a column for each object. Only those operations that are pre-defined in a cell of the access matrix can be executed by the subject in the row to the object in the column [16]. The information included in it is usually represented separately row-by-row, in the form of access control lists (ACLs), or column-by-column, in the form of capabilities [19].

2.1.5 Access control list

For a specific object, an access control list defines whether a list of subjects can act on the object and what kinds of actions each subject can perform on the object. Each of the permissions defines objects and actions that the subject may perform on the objects, as shown in Figure 2-1.

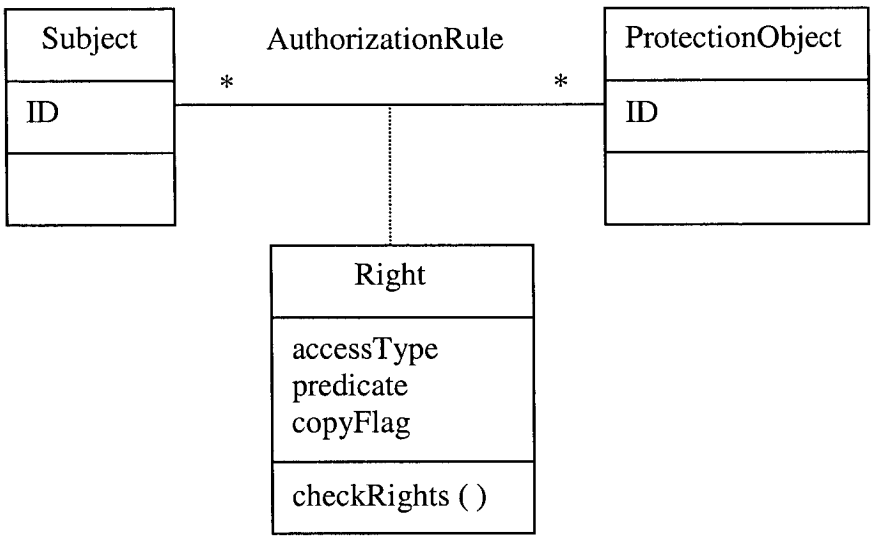

Figure 2-1: Basic authorization pattern [62]

For the purposes of this research, our main interest is in explicit rules and policies that apply on role based access controls. One of the greatest obstacles to the growth of a distributed system is the inability to manage authorization data effectively. Access control lists (ACLs) are constructed for authorization management to control user access to functional services based on the providers' location of those services.

ACLs usually specify a list of named individuals, or groups composed of individual users, with their respective modes of access to those services or objects. ACLs tie every end user to particular objects. In many enterprises environments, end users do not "own" the information to which they are allowed access. In contrast, the organization who utilizes the system is the actual "owner" of system resources or services; hence connecting those objects with the type of controls on the basis of the users may not be appropriate.

Therefore, it is important to make a policy available in classifying information related to subject-based policies with rule agreement and least privilege access. The support of such policies requires the ability to restrict the right of entry on the basis of a user function or role within the distributed system.

The characteristics of ACLs that associates users with permissions make it complicated due to a very large number of user and permission management workload. This situation is even worse when a user takes on different responsibilities, resulting in the selective addition or deletion of user/permission associations on all servers that greatly increase the risk of maintaining residual and inappropriate user access rights.

2.1.6 Permission and policy

Permission is an approval of particular mode of access to one or more objects in the system. Positively defined and granted on their holder the ability to perform an action in the system, permissions can not only apply to one or more objects and but also be specified as concrete actions to those objects.

Policies are high-level guidelines that determine how access is controlled and how access decisions are determined. Usually, policies are divided into authorization policies for access control management and obligation policies for resource management [46]. Access control mechanisms are largely independent of the policy for which they could be used. The desirable goal is to allow mechanisms to be reused in order to serve and support a variety of security purposes such as secrecy, integrity, or availability objectives [5].

In general, there are no policies that are "better" than others; rather, there exist policies that ensure more protection than others. Policies suitable for a given system may not be suitable for another. The choice of access control policy depends on the particular characteristics of the environment to be protected. For instance, very strict access control policies, which are crucial to some systems, may be inappropriate for environments where users require greater flexibility.

2.1.7 Personalization

In the realm of the Internet, personalization is the process of forming pages or generating services based on an individual user's profile, characteristics or preferences. Commonly used to enhance customer service or e-commerce sales, personalization is a means of meeting the customer's needs more effectively and efficiently, making interactions faster and easier and consequently, increasing customer satisfaction and the likelihood of repeat visits. Many portal-based sites, such as Yahoo allow site visitors to customize the page with selected news categories, local weather reports, and other features. To gather specified information on the Web and deliver it in user-specified format and layout, push technology is used for delivering of personalized information. Moreover, the commonly used technologies that apply to personalization include looking up user profile and rule base, data mining, collaborative filtering, and data analysis tools for predicting possible future interactions [7].

2.2 Types of access control

The introduction of Service Oriented Architecture (SOA) such as Web services technology has led to the practice of constructing software systems as collections of heterogeneous distributed components. Such systems are increasingly used for database integration, decision support systems, e-business, e-learning and many other distributed systems. In general, the services provided and resources stored by those systems are sensitive and requires some form of access control approach. This is particularly important as the internet is increasingly used as the basis for distributed systems and as the threat from hackers and malicious software continues to grow. To protect a system from these external threats, access control should also ensure compliance with privacy laws and should ideally guarantee that each user with access to the system uses the information only exactly as required and matched with their role within the organization.

In fact, as a critical functionality, access controlling has been implemented, maintained, monitored, and regulated within most security system to secure organization system and access control services on the application-level. This section briefly describes the classifications of access controls proposed.

2.2.1 Role-based access control

Role management includes concepts such as roles and positions to specify the organizational structure and activities for individuals within an organization. A role can be defined as "a collection of rights and duties" and a position can be described as the status within the organization.

Access rights can be grouped by role name, and the use of resources is restricted to individuals authorized to assume the associated role. The use of roles to access part of the system can be an effective means for developing and enforcing enterprise-specific security policies, and for streamlining the security management process [60].

Role-based access control (RBAC) is a technology that is attracting a great deal of attention because of its potential for reducing the complexity and cost of security administration in distributed systems. Under RBAC, security administration is simplified by using roles, hierarchies and constraints to organize user access levels. RBAC reduces costs within any organizations, which is applicable in a situation that employees or students change more frequently than the duties changes related to positions [2].

With RBAC, access decisions are based on the roles that individual users have as part of an organization. The process of defining roles should be based on a thorough analysis on how an organization operates. Users take on assigned role, for example, roles can be classified as instructor, student, system administrator, and secretary within an educational institute.

RBAC does not permit users to be directly associated with permissions. The fundamental notions of RBAC are user, role and permission, where users are authorized to use the permissions assigned to the roles they belong to, and roles are created based on organization needs for various functions. Permissions are associated with roles, and subjects are assigned roles based on their responsibilities and qualifications within the organization. Figure 2-2 shows role-based access control model. RBAC regulates the access of users to services and resources of an application system based on the users' task requirements within an organization.

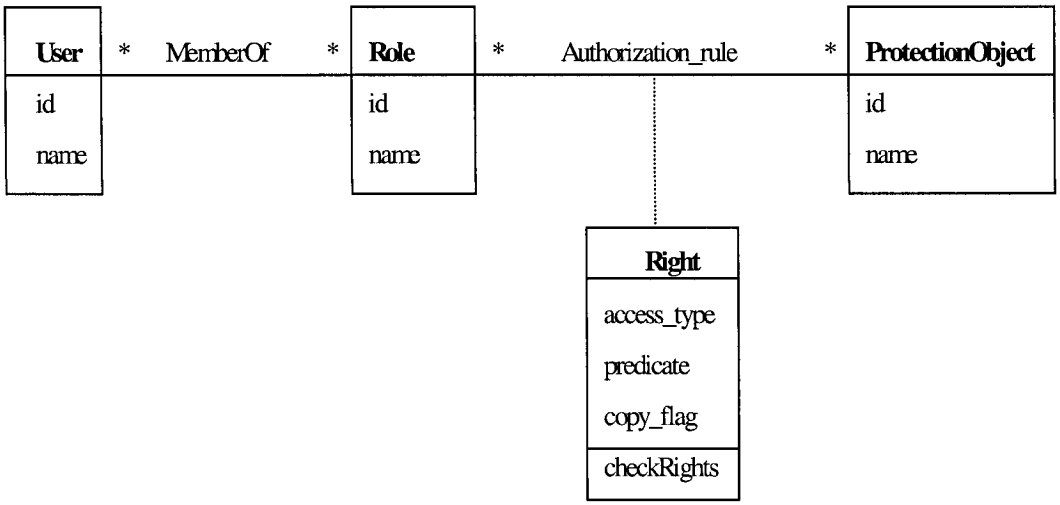

Figure 2-2: Role-based access control model [62]

In addition, separation of duties, an attractive feature offered by RBAC, is the ability to prevent a user from assuming conflicting roles. The enforcement of separation of duty requirements ensures that a single user cannot acquire too much authority. For example, a single individual may be able to assume both "instructor" and "student" roles at different times but not simultaneously.

Based on the role concept, RBAC supports security principles such as least privilege, separation of duties, and data abstraction to provide a fine granularity of access control [16]. This facilitates the management of the access control and reduces the cost of security administration.

2.2.2 Mandatory and discretionary access controls

Mandatory and discretionary access controls are known as MAC and DAC respectively. MAC is a standardized type of categorizing resources and users based on a predetermined set of criteria assigned to subjects and objects. DAC, on the other hand, is a subjective type of allowing decisions made by an individual user that controls access to an object on the basis of a subject's permissions or denials [18]. MAC usually is more secure than DAC but typically less convenient for end users, who rely on another party to identify their security clearance level [18].

Examples of MAC are a specific portal existing in an E-learning system which can only be accessed by a pre-defined group of users. In contrast, an example of DAC in an e-learning system is that when importing a resource, an instructor can define an access control list that regulates access rights on associating user with resources based on actions. This type of system takes away the control from a central authority and leaves the assigning of permission to the resource's creator. Therefore, DAC usually is granted based on a combination of user-ID/password that is linked to a set of permissions.

2.2.3 Rule-based access control

Although role-based access control has satisfied the basic authorization needs which has been implemented as the main user identity management product, it has some significant drawbacks if encountering such those situations that a large number of different roles are needed, and restricting users' access based on the users' attributes and even the environment parameters. Rule-based access control mechanisms allow users to access services and contents based on pre-determined and configured rules. Generally, a rule can be treated as a basic generalization being accepted as true and used as a basis for reasoning or conduct [30]. Thus, rules can be established to restrict accessibility by all end-users coming from a particular domain, host, network, or IP addresses.

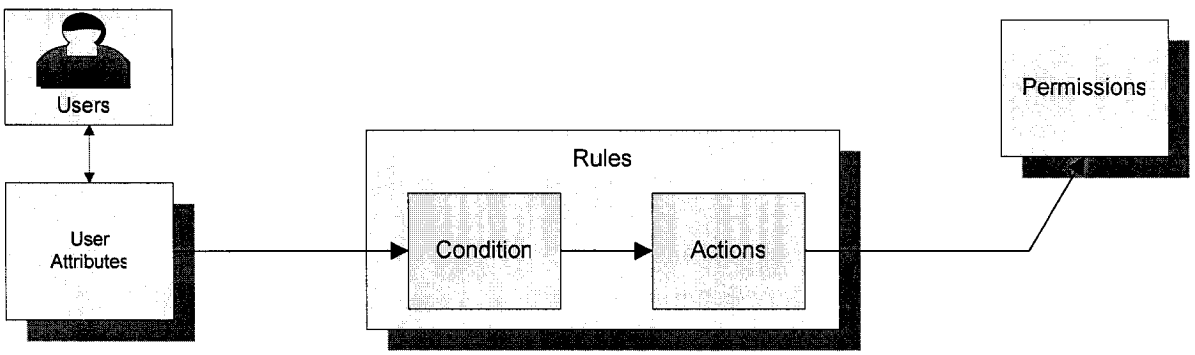

Figure 2-3: Rule-based access control model

Usually, rules consist of a condition in the so-called left-hand side (LHS) and one or more actions on the right-hand side (RHS) [29], that is, when the expression on the LHS is true, the action on the RHS is executed. A rule is often described as If-C-Then-A condition statement, where C represents LHS and A represents RHS.

Being able to retrieve a rule from users' attribute database makes a highly dynamic access control mechanism, rule-based access control has some drawbacks in that it is hard to obtain an overview concerning who is allowed to do what. The capacity of roles to build "business roles" which contain all permissions for a specific service associated to a specific user or a group of users is difficult. Figure 2-3 shows the rule-based access control model. A "rule" consists of the following two things, filter condition and actions. This means that we can define only one filter condition per rule but we can define as many actions as deemed necessary. Filter condition is a combination of different Boolean operators which will combined resulting in an answer that is either "True" or "False"; whereas actions are events that are conducted when a filter condition output is "True".

As mentioned above, a rule can have more than one actions associated with it which means that if a filter condition evaluates to a true value then all of the actions associated with that rule will execute. If the filter condition, on the other hand, evaluates to a false value then all of the defined actions will be skipped.

The rules have to be interpreted by a rule engine at runtime. A rule engine can be understood as a high-performance and specialized interpreter for if-then commands that analyzes the transferred values and objects based on pre-defined rules and then returns values of "True" or "False" or performs actions directly. Thus, the rule engine fetches "rule sets" one by one from the rule repository by picking up the first rule and evaluates its filter condition. If that filter condition is evaluated to "False", all the actions associated with that rule will be skipped and it will pick up the second rule. If the filter condition, on the other hand, is "True", it will execute all the actions that are associated with this rule in the order in which they are defined. After the execution of all the actions, it will pick up the next rule in the current rule set. Once all the rules have been executed, the current rule set will be destroyed and the rule engine will fetch the next rule set if any.

If, however, we associate the rule-sets with users' role, even when users change their role within the organization, their existing authentication credentials remain in effect but they are

allowed to access those services based on the role of the users. Using rules in conjunction with roles adds greater flexibility because rules can be applied to people, as well as devices.

2.2.4 Policy-based access control

Policy-based access control (PBAC) is a strategy for managing user access to one or more systems, where business classification of users is combined with policies to determine user access privileges [Figure 2-4] [40]. Theoretical privileges are compared to actual privileges, and differences are automatically applied to manageable systems [38].

For example, within an E-learning system, a role may be defined for an instructor. Specific types of tasks such as creating course and uploading learning materials and document management system may be attached to this role. Appropriate users are then attached to this role. A policy-based provisioning system would evaluate the actual privileges of every user based on some parameters such as the IP address or time the system is accessed. Moreover, to implement a PBAC model for an application system, new information should be scrutinized and examined. This includes the definitions of the exhaustive set of roles, the access requirements for each user specified in terms of roles, and a periodic process to measure the privilege set possessed by each user and comparison of his/her privileges with the pre-defined access policy.

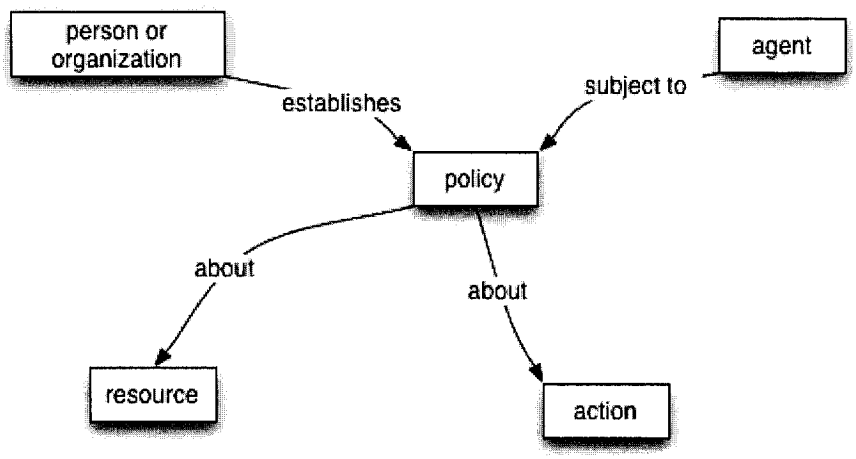

Figure 2-4: Typical policy model

Policies define a relationship between a subject and a target object domain. This relationship expresses either an authorization (what activities the users are permitted or forbidden to perform) or an obligation (what activities the user must or must not perform on the managed objects). Obligation policies are triggered by events while authorizations are considered to be valid until revocation [37].

PBAC is also known as Rule Set Based Access Control (RSBAC). An access control policy is a set of rules that determine users' access rights to resources within an enterprise network (e.g., files, directories, sites, web pages).

One prevailing mechanism for enforcing enterprise policy is the use of access control lists (ACLs). ACLs associate with every resource with a list of users or groups of users and their access rights (i.e., the type of access attempts that should be allowed). ACLs are ineffective in enforcing policy. When using ACLs to enforce a policy, there is usually no distinction between the policy description and the enforcement mechanism - the policy is essentially defined by the set of ACLs associated with all the resources on the network. Having a policy being implicitly defined by a set of ACLs makes the management of the policy inefficient, error prone, and hardly scalable to large enterprises with large numbers of employees and resources. In particular, every time an employee leaves a company or even just changes his/her role within the company, an exhaustive search of all ACLs must be performed on all servers, so that user privileges are modified accordingly.

Coupled with the policy description language there should be an enforcement mechanism capable of intercepting access attempts, evaluating them against the policy, and accordingly granting or denying the access requests.

In addition, policy-based management provides a means for administrators, end-users and application developers to manage and dynamically change the behavior of computing systems. A policy based management architecture based on the Internet Engineering Task Force (IETF) framework consists of four main components: a policy management tool, a policy repository, a policy decision point, and a policy enforcement point.

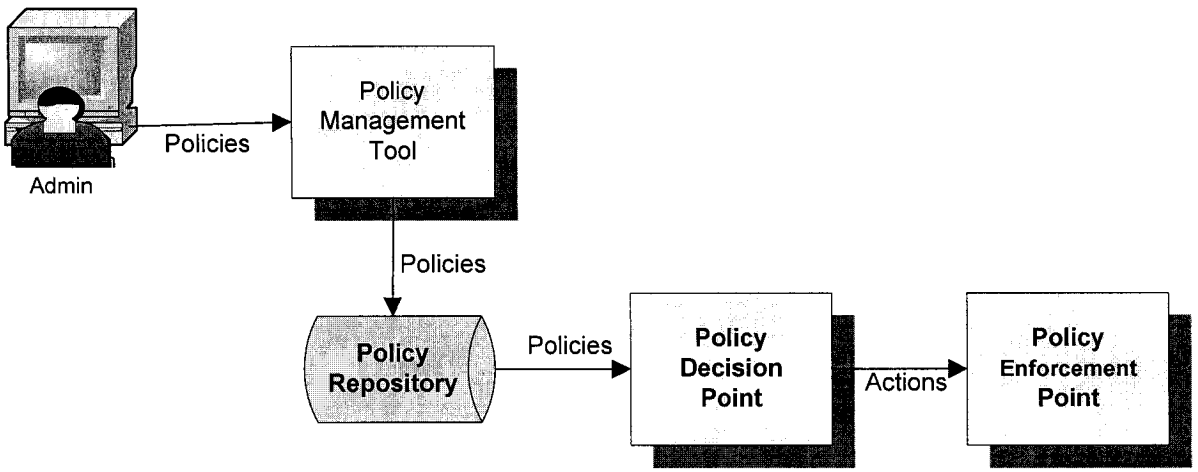

Figure 2-5: Policy-based management architecture

Figure 2-5 shows a general architecture for such a policy based management system [45]. Most current systems employing policies follow such architecture. After a system administrator enters the policies via the policy management tool, these policies are then sent to and are stored in the policy repository and are retrieved by the policy decision points (PDPs). The PDP is responsible for interpreting the policies stored in the repository and communicating them to the policy enforcement points (PEPs). The PEP is the system component that actually applies and executes the policies. The PEP and the PDP may both be located on a single device or they can be on different physical devices [43].

Furthermore, PBAC makes a strict distinction between the formal statement of the policy and its enforcement. Making rules explicit, instead of concealing them in ACLs, gets the policy easier to manage and modify. Such a mechanism is usually based on a specification language which should be expressive enough to easily formulate the policy rules.

Thus, PBAC is not designed with fine granularity or management simplification in mind. In addition, these approaches must be used in combination with other access control approaches for better effectiveness [39].

Chapter 3 Analysis of Access Control for Ubilearn

3.1 Overview technologies for e-learning

The recent influx of "e-" application systems such as e-business, e-commerce, e-government and e-learning has combined conventional business processes, legacy technologies, and Internet-based applications. Emerging as a comprehensive training solution, e-learning systems have been recognized and utilized by many organizations and educational institutes to provide an integrated and scalable teaching and learning technology platform [54]. Operating seamlessly together, the main e-learning components include content, collaboration, testing and assessment, skills and competency, Internet video-audio-based learning, e-payment and learning management system (LMS). Usually, by connecting all other components together, LMS is the infrastructure containing those segments as technology, content and services that accomplish the tasks of tracking, supporting, managing, and measuring e-learning process [10].

From the technical view, usually, an LMS is a Web-based application that consists of three fundamental tiers [Figure 3-1]:

- Presentation-tier: presents data to the end user or system;
- Application-tier: performs the business logic, processes user input, makes decisions, obtains more data and present data to the Presentation tier to send back to the user; and
- Data-tier: stores things needed by the application and act as a repository for both temporary and permanent data.

3.1.1 Traditional distributed computing system

A Web application is a client/server software application that interacts with users or other systems using HTTP. For an end user, a web browser such as Internet Explorer or Netscape Navigator can be treated as the client; a client could also be an HTTP user agent that acts as an automated browser. The end user views web pages and is able to interact with them by

sending choices to and from the system. The functions performed can range from relatively simple tasks like searching a content for a file or reference, to highly sophisticated applications that perform real-time sales and inventory management across multiple vendors, including both Business to Business (B2B) and Business to Consumer (B2C) e-commerce, workflow and supply chain management applications.

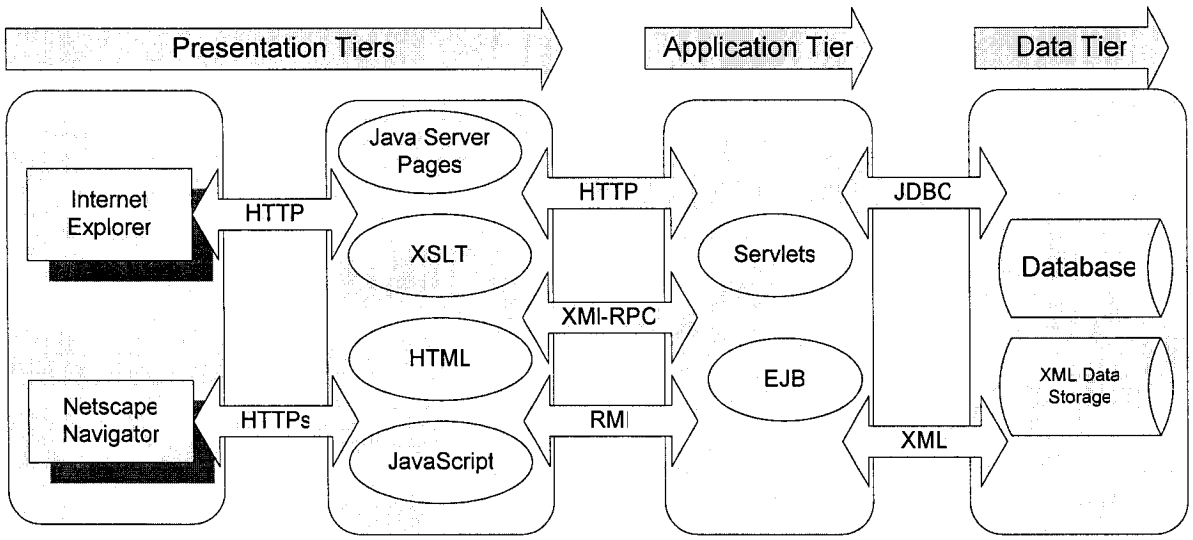

Figure 3-1: Web application multi-tier model

Traditionally, simple applications were built with a common gateway interface application (CGI) running on the web server itself and often connecting to a simple database. Modern applications written in .NET or Java run on distributed application servers and connect to multiple data sources through complex business logic tiers. Moreover, a traditional distributed computing system usually contains several interconnected and relatively independent subsystems that share the computing tasks and service resources through some common communication middleware over a network [22].

The most common used types of middleware in distributed computing are remote procedure calls (RPC), message oriented middleware (MOM), and common object request broker architecture (CORBA).

- The typical communication processes for RPC client-server application include the client-side application sending a request, waiting for response from server-side, and continuing its own processing.

- MOM is a kind of middleware that resides in both the client and the server sides of a distributed system and typically supports asynchronous calls between the client and server applications to exchange messages. Nominally, MOM systems provide a message queue between interoperating processes, so if the destination process is busy, the message is held in a temporary storage location until it can be processed [23].
- Developed by the Object Management Group (OMG), Common Object Request Broker Architecture (CORBA) is a typical architecture which applications utilize to communicate with each other in a decentralized environment. CORBA applications are composed of *objects*, individual units of running software that combine functionality and data and that frequently represent something in the real world [24].

3.1.2 Service-oriented architecture

Within the realm of software engineering, the architecture of a software system describes the components, and interactions between components at a high level. Some problems can be raised when combining heterogeneous components that have been developed for different systems. Enormous dependency problems in scalability occur due to the fact that these components must know too much about the interfaces of each other.

Service-oriented architecture (SOA), in contrast, is an architecture that consists of components designed and built through heterogeneous networks and featured with interoperability and location transparency [8]. Bearing a network-addressable interface and being provided by a component, a service can be dynamically discovered and used by any other component based only on the interface contract. Web Services include HTTP as the primary network protocol, SOAP/XML for the payload format, UDDI (Universal Description Discovery and Integration) for service repository, and WSDL (Web Services Description Language) to describe the service interfaces.

3.1.3 Web services interaction model

The Internet has brought a revolutionary change in the way of conducting businesses. Although the initial focus was primarily on B2C interactions, there is overwhelming evidence that businesses, organizations and companies need not only participate in trading partnerships but also work in close cooperation. B2B integration (B2Bi) is about

coordinating activities by transparently sharing information between organizations to streamline and automate business processes which provide better quality of service [44].

Traditional middleware platforms based on component technologies such as COM/DCOM, J2EE, and CORBA have matured and have been well proven to implement key enterprise requirements such as distributed transaction integrity, workflow automation, security, scalability, and so forth. However, there are some downsides to it. One major drawback of these middleware architectures is their inability to interoperate with each other. This is primarily due to the binary incompatibility between the object models and proprietary protocols used for information exchange. For instance, a COM object running on a Microsoft platform cannot quite easily interact with a Java component or a CORBA object. Yet another problem associated with component technologies is firewall penetration. Object protocols such as DCOM, RMI, and IIOP operate using an arbitrary set of ports on the server, thereby opening up potential security holes. For security reasons, Web administrators are wary of opening ports other than 80.

Web services offers a perfect solution by providing a uniform interface to access business functions using protocols such as SOAP and XML-RPC. Working with XML as the payload over the ubiquitous HTTP, these protocols circumvent both problems mentioned above. Web services include the collection of functions that are packaged as a single entity and published to the network for use by other programs. Web services are building blocks for creating open distributed systems, and allowing companies and individuals to quickly and cheaply make their digital assets available worldwide. One Web Service may use another one to build a richer set of features to the end user.

As a new distributed computing model, Web services model can fulfill application-to-application communication over the Internet, in which different organizations' applications are implemented by using different platforms. A data-oriented service provides a flexible way to allow for the interaction of applications through a well-defined interface by using an XML-based description language. Web services are self-contained, self-describing modular applications that can be published, located, and invoked across the Internet [25].

Specifically, the W3C's Web Services Architecture Working Group has given the definition of a Web service: "A Web service is a software application identified by a URI, whose interfaces and bindings are capable of being defined, described, and discovered as XML artifacts. A Web service supports direct interactions with other software agents using XML-based messages exchanged via Internet-based protocols" [28].

Basic Web services combine the power of two ubiquitous technologies: XML, the universal data description language; and the HTTP transport protocol widely supported by browser and Web servers, that is:

Web services = XML +transport protocol (such as HTTP)

The Web services framework consists of three roles: service requester, service broker, and service provider, whose interactions involve operations of publishing, finding, and binding. Figure 3-2 shows their relationships [26]. In a typical scenario [28], a service provider hosts a network Web services-based accessible module. The service provider defines a service description (using WSDL) for the Web service and publishes it to a requestor or service discovery agency. The service requestor uses a find operation to retrieve the service description locally or from the discovery agency, which is called UDDI repository, and uses the service description to bind with the service provider and invoke or interact with the web service implementation. Service provider and service requestor roles are logical constructs and a service may exhibit characteristics of both.

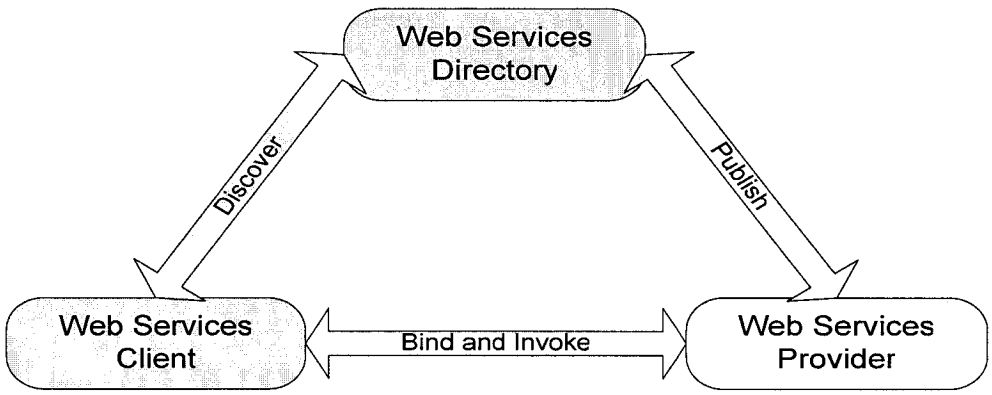

Figure 3-2: Web services: Publish-Discover-Invoke model

A generic Web services architecture is divided into the following four tiers: Service requester tier, Web service tier, Business logic tier, and Backend tier. An independent Web service tier is added to the traditional client-server architecture so that Web service solution can be integrated with an existing distributed infrastructure. Because SOAP is used for standard data exchange between the first two tiers of this Web services architecture, the Web service tier can be called through any kind of application, whether implemented on a Windows or UNIX platform using J2EE Servlets or .NET.

Furthermore, we can use the Web services technologies to create rapidly reusable application-to-application communication interfaces to be accessed by any of the components at any of the tiers. The typical communication between Web services components is the SOAP communication, which begins by a service requestor sending a SOAP request message to a UDDI registry to retrieve the information of desired services. It then communicates with the Web Service tier application directly through SOAP or XML based RPC. In the Web service tier, a Web service application can publish its service description document called WSDL file to a UDDI registry. Here, an XML handler is designed to parse and compose the request and response SOAP messages. Intercepted data from request messages are routed to the Web services component, in which the desired functional objects are invoked via the underlying middleware. The Web services component then retrieves the result from the middleware, encoding it in XML and sending the response back to the Service request tier.

The middleware that resides in the Business logic tier could be any one of sophisticated technologies such as RMI, JMS, DCOM, CORBA or any other vendor-specified API. The business/application process components are actually the requesters ultimately interested in accessing. The last backend tier, the resources that the requester is ultimately trying to access could be a relational database system, an enterprise resource planning system or a legacy mainframe system [12]. Figure 3-3 shows the communication between Web services client and server [44].

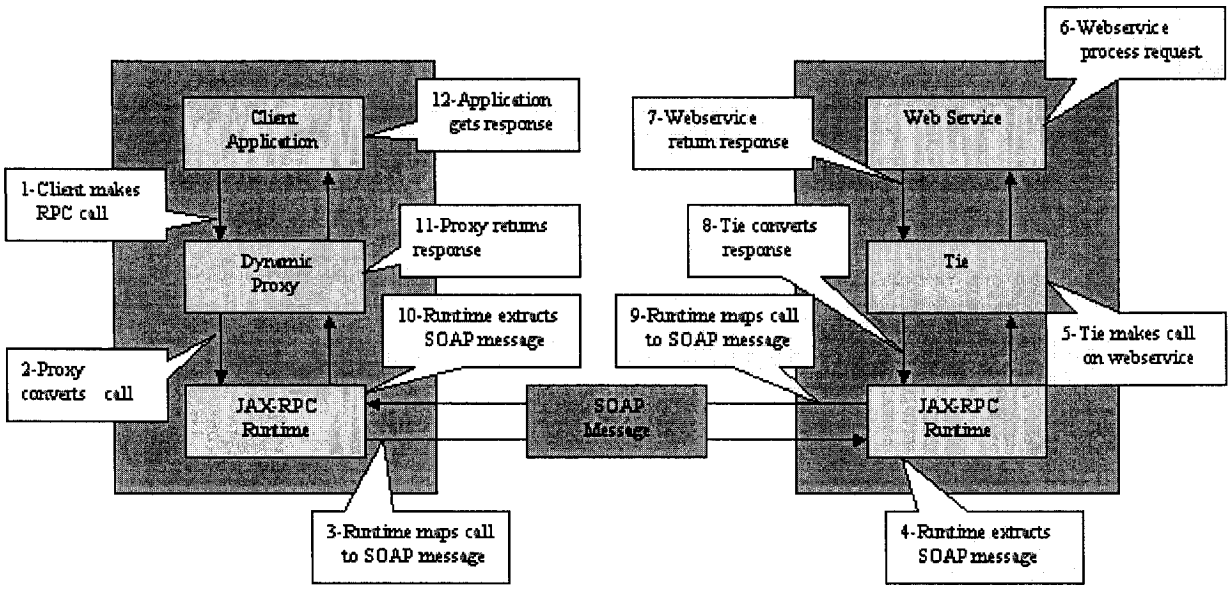

Figure 3-3: Web services client and server communication

An E-learning system can utilize the Web services mechanism to strengthen relationships among the diversity of E-learning components, achieve seamless integration inside and outside educational institute, gain real-time views of user accounts, and increase operational efficiencies and reduce costs.

3.1.4 Portlet and portal

From an end user perspective, a portal is a Web site with pages that are organized by tabs or some other form of navigation. Each page contains a nesting of sub-pages, or one or more portlets-individual windows that display anything from static HTML content to complex interfaces of client-side of Web services components. A page can contain multiple portlets, providing users with access to different information and tools in a single place. The portlet is an extension to the Java Servlet and many aspects of portlet development are common to typical Web application development. A portal, on the other hand, is a Web-based application that provides personalization, single sign-on, and content aggregation from different sources, and hosts the presentation layer of information systems.

Usually, portlets can be shared and exchanged by various portlet containers. Once the portlets contained in a portal receive a request from a user, they process requests and

generate dynamic contents, showing all the services that the user can access. Portals use portlets as pluggable user interface components that provide a presentation layer to information systems [Figure 3-4].

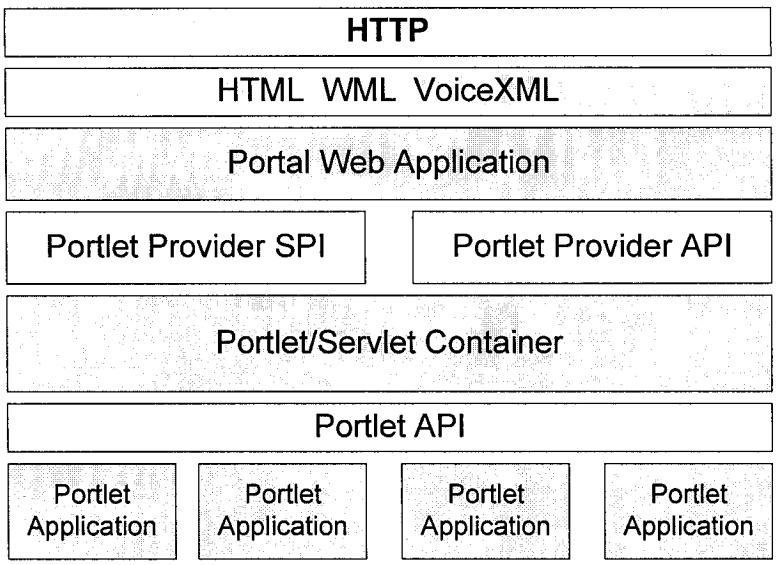

Figure 3-4: Relationship of portal and portlet

Generally, a portal, which organizes portlet-set into portal pages, may include sophisticated personalization features in order to provide customized content to different users. From the point of view of access control, it is necessary to set up a mechanism to control users' access to services (representing via a portlet) based on the group the user belongs to.

Precisely, a portlet represents active components of a portal. Portlets are responsible for the generation of widgets of a graphical user interface. A portal is defined by an individual set of portlet modules.

Whenever the interface is newly created, all portlet modules of the portal are loaded and each portlet module is asked to create an object using the new method which is provided by the base class. Moreover, actions are portlet-specific activities that need to be performed as the result of the incoming request. Figure 3-5 shows how actions are created and executed.

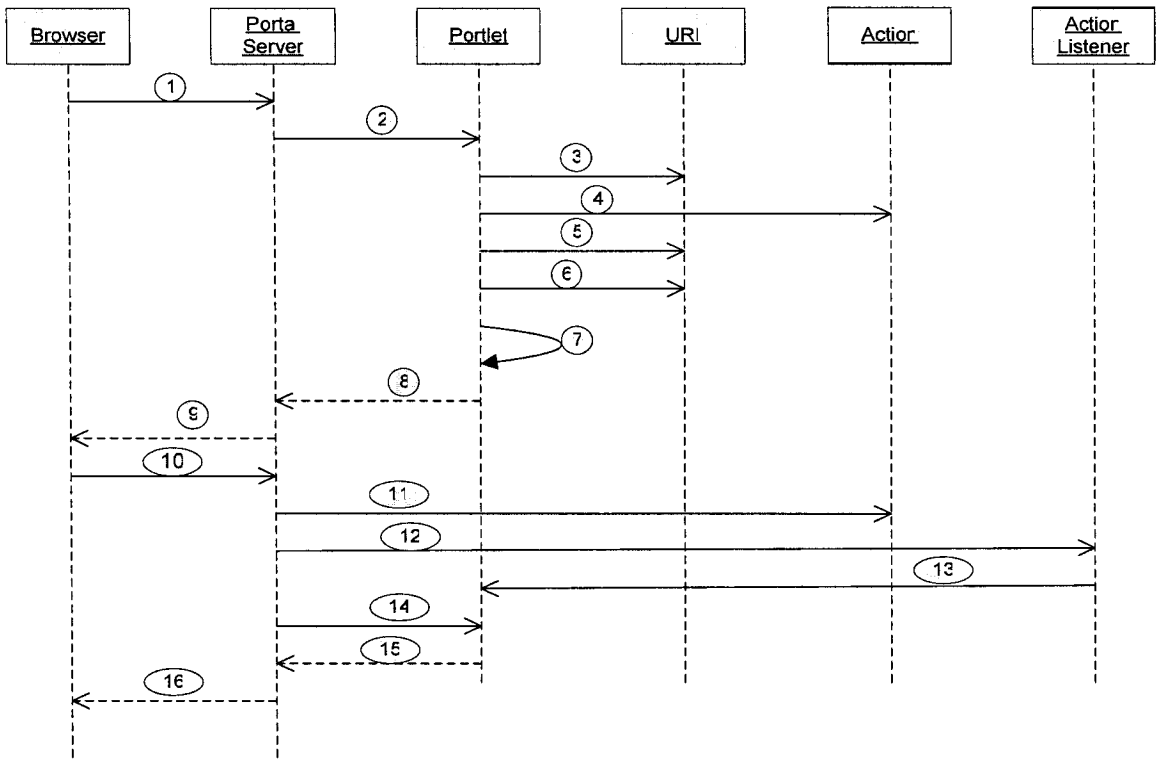

Figure 3-5: Creations and executions for a typical portlet action

The procedure for the creation and execution of a typical portlet action are as follows [20]:

- The portlet creates a URI pointing to itself;
- The portlet creates an action and attaches it to the URI;
- The portlet writes the URI as the HREF of a URL in its HTML markup;
- The portal aggregates this markup to a page;
- The user clicks on the link;
- The browser sends the request to the portal server;
- The portal server notifies the action listener of the portlet of the action event;
- The action listener can change the state of the portlet; and
- The portal server asks the portlet for its markup in order to generate the response to the request.

Table 3-1 Message transmitting table

Series #	Method name	Series #	Method name
1.	get portal page()	9.	return page()
2.	request service	10.	click on URI()
3.	createURI()	11.	get action for event()
4.	createAction()	12.	action performed()
5.	addAction()	13.	change state()
6.	toString()	14.	request service
7.	write HTML of URI()	15.	return fragment()
8.	return fragment()	16.	return page

Generally, a portal framework consists of a presentation component, a portal engine component, and a portlet container.

- The presentation component provides customized and personalized pages for users through aggregation. The page content is composed of a combination of content from a variety of sources and applications.
- The portal engine component provides a pure java engine whose main responsibility is to incorporate content from different resources and serve those incorporated contents to multiple devices. It also provides a mechanism to decouple the presentation layer of the portal from the portlet implementation details.
- The portal container can be treated as a container of Web services requester which includes several local portlets called service requester and remote portlets.

3.1.5 Naming and directory services

A Naming Service provides a mechanism for assigning names to objects that can be retrieved and used without knowing the location of the object. Objects can be located on any machine accessible from the network, not necessarily the local workstation. Naming Services not only provide an indispensable mechanism for de-coupling the provider of a service from the consumer of the service, but also allows the supplier of a service to register the service against a name. Therefore users or clients of the service only need to know the name of the service in order to use it.

The Lightweight Directory Access Protocol (LDAP) is an information model and a protocol to query, manipulate, and access information directories such as organizations, individuals, phone numbers, and addresses. Usually, the information directories reside on the so-called LDAP-based Directory Server.

An LDAP directory is organized in a simple "tree" hierarchy to be distributed among many servers. Each server can have a replicated version of the total directory that is synchronized periodically. An LDAP server that receives a request from a user takes responsibility for the request, ensuring a single coordinated response for the user.

Roles within a Directory Server entry mechanism are similar to the concept of a group. Just as a group has members of roles, a role has members of users and a profile list of user is an LDAP entry that is said to possess the role. The criteria for the role itself is defined as an LDAP entry with attributes, identified by the Distinguished Name (DN) attribute of the entry.

3.1.6 WSRP

Web Services for Remote Portals (WSRP) are visual, user-facing web services centric components that plug-n-play with portals or other intermediary web applications that aggregate content or applications from different sources. Since WSRP includes presentation, service providers can determine how their content and applications are visualized by end-users and to what degree are adaptation, transcending, and translation allowed.

WSRP decouples portlet applications from portals, which offers significant benefits for managing large portal deployments. Instead of bundling all portlets within one single portal application, we can choose to deploy our portlets on a remote server, and let the local portal consume those portlets through the WSRP mechanism. For most large portal development projects, this decoupling eases team development, upgrades, and administration. The WSRP protocol defines a set of web services that WSRP Producers implement. WSRP consumers can send messages to Web services to view and interact with the UI. The WSRP protocol translates the typical browser-server interaction protocol into a protocol that is suitable for applications (consumers) to act as clients for other applications (producers) that host the UI.

An important point to realize is that WSRP web services are synchronous and UI-oriented. Unlike business logic-oriented or data-oriented web services, UI-oriented web services offer a more coarse-grained application reuse and are more resilient to change [27].

WSRP natively supports portlet concepts, such as modes (view, edit, help, preview, and custom), window states (maximized, minimized, solo, custom), and portlet preference management. This makes it much easier to integrate portlet applications in a portal paradigm. Furthermore, WSRP addresses some of the common integration issues, such as transparent session management, timeout handling, caching, and support for different markups, which integrates applications as services rather than fragile links. Moreover, WSRP applications are built over the SOAP stack that can be published into public or corporate service directories (UDDI), thus, not only can they leverage enterprise Web service management infrastructure, including service metering, routing, prioritization, and life-cycle manageability, on versioning, monitoring, and upgrade, but they can also be discovered from a UDDI registry and secured using Web service security standards. For transport-level security, WSRP can be used with SSL.

To accomplish these goals, the WSRP standard defines a web services interface description using WSDL and allows WSRP services to be implemented in very different ways, whether as a Java/J2EE based web service, a web service implemented on Microsoft's .NET platform, or a portlet published as a WSRP Service by a portal. The standard enables use of generic adapter codes to plug in any WSRP service into intermediary applications rather than requiring specific proxy codes.

Carrying reusable presentation, interoperability and portability, WSRP has combined the power of Web services and portal technologies and is fast becoming the major enabling technology for distributed portals in an enterprise. In addition, WSRP possesses features such as visually rich functionality, presentation-oriented and reusable user interfaces, interactive applications with complex flow on multi-step and multi-page user interaction code sharing at the portlets level across different applications, and no re-implementing of the presentation layer on each portal.

3.1.7 Federated portal

Today, software applications running on distributed environments are becoming more complex with increasing business processes and there is a need to render a unified face by using an enterprise portal. Thus, it is definitely a challenge to consolidate different portals, information islands and business processes, and bring information and data together from different business portals into a centralized enterprise portal for a variety of reasons. The challenge of realizing a true enterprise portal becomes obvious when it is imperative to consider the governance, information ownership models, and trust within an enterprise for information life-cycle management. The availability of the right technology and standards is important to realizing truly distributed portal information domains and simultaneously providing a seamless integration and delivery of information across the enterprise. Thus, the infrastructure for service on demand portals has been developed the use of SOA and WSRP [35].

Let us consider an educational institute that has several existing departmental portals. Each of them could be residing on a different technology. Some of them may not even be on a portal product. Each portal may or may not have the same look and feel or navigation, but all departmental portals participate under a central Single-Sign-On (SSO) umbrella. Each departmental portal is self-contained and provides unique functionality specific to its product or services.

By utilizing the Federated portal model, we can create a main portal that serves as an entry point or gateway to the enterprise. The main portal is a thin layer that sits on top of the individual departmental portals and leverages them. A federated portal allows organizations to provide a common entry point, but, at the same time, maintain the individual departments' sovereignty so they may develop, maintain, and control their released schedules. A truly federated portal needs to integrate with transactional services and information across the enterprise, provided by various specialized portals and service infrastructures. Bearing a more elegant, service-oriented approach by integrating into an industry-standard manner, WSRP is an ideal technology for building federated portals [35].

The main portal acts as a facade [Figure 3-6] serving up a common log-on and home page. It authenticates users against the central SSO infrastructure to display a home page, which is composed of multiple portlets that display information from various departmental applications exposed as WSRP services. Each portlet, acting as a WSRP consumer and interacting with the WSRP producer on the other end, achieves departmental services integrated seamlessly into the main portal. It is convenient for a portal developer to rebuild the application in the main portal by adding a WSRP Producer layer on top of existing applications [35].

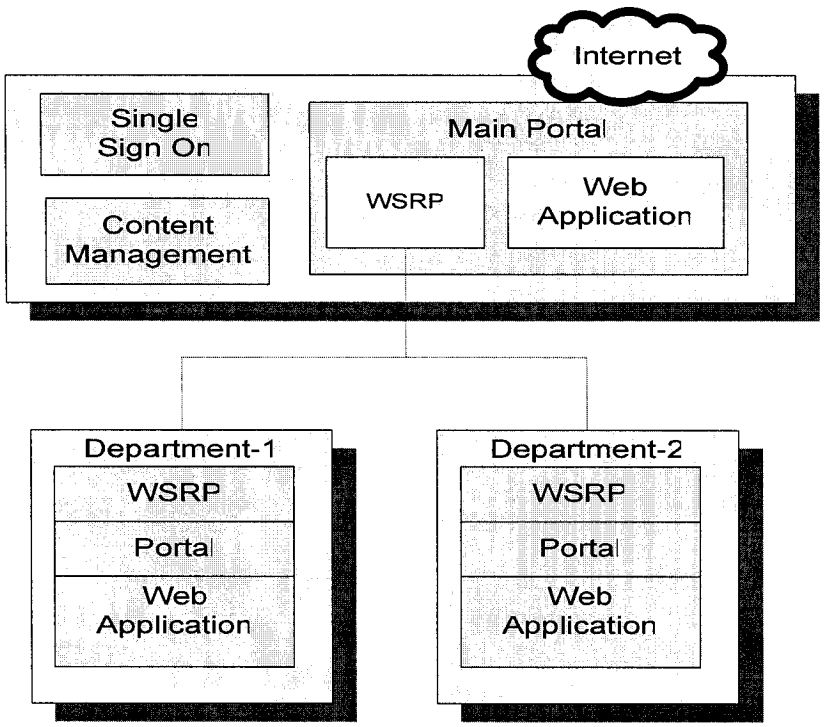

Figure 3-6: Federated portals

Moreover, there should be some restrictions on the responsibilities of WSRP consumers and producers. The WSRP consumers may only conduct content/application service assembly. Portal (top Level) personalization and customization, authentication and services may only be accessed through the departmental portal; whereas the producers provide business logic, content rendering, portlet-specific customization, and personalization, and access permission checks.

3.2 Architecture of e-learning system

3.2.1 Architecture layers

We designed and implemented our Ubiquitous Learning (Ubilearn) system by combining traditional and new distributed computing techniques. We exploited portal principle to centralize the starting place for access and to consolidate the functions through well designed interfaces with portlet applications. As shown in Figure 3-7, a generic architecture layer model to which our Ubilearn system is adopted consists of multiple tiers including a portal Web application, a portlet container in which portlet interfaces connect to a variety of applications such as Web services-based components, non Web services-based applications, Web business logic tier components, and WSRP components.

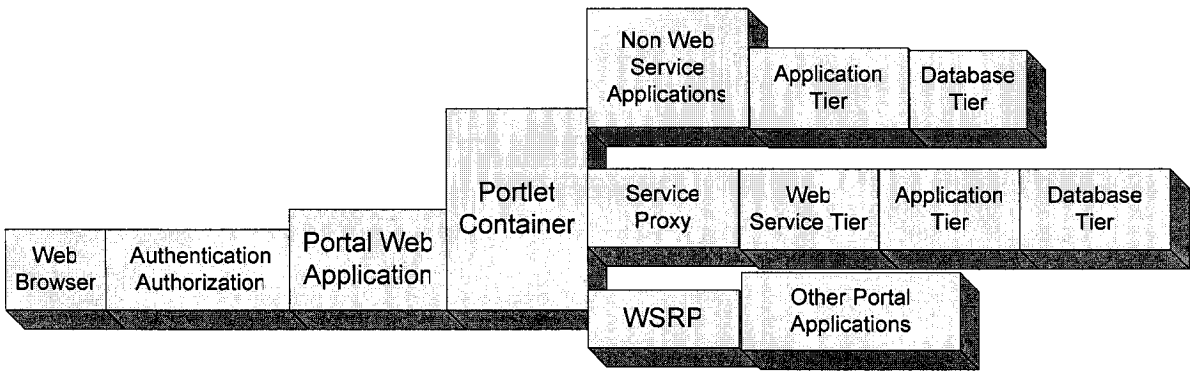

Figure 3-7: Generic architecture layer model

3.2.2 Ubilearn system

Figure 3-8 illustrates the Ubilearn system diagram for a multiple-tier e-learning Web services-oriented portal framework running in distributed environments.

Here, we give an explanation on organization of Ubilearn. The Ubilearn system framework includes a portal presentation layer, portal engine layer, service provider layer, service broker layer, and service requester layer. The portal presentation service layer provides customized and personalized pages for users through aggregation model. The page content is aggregated from a variety of sources via content and applications. The portal presentation framework simplifies the development and maintenance of the portal by defining the page structure independent of the portlet definition. Portlets can be changed without an impact on the

overall portal page structure. Aggregation Modules for the requesting device render the overall portal page by collecting information from all the portlets on the page and adding standard decorations such as title bars, edit buttons, etc.

Providing a portal engine layer, the main responsibility for a portal is to aggregate content from different sources and to serve the aggregated content to multiple devices. The Portal Engine also provides a framework that allows the presentation layer of the portal to be de-coupled from the portlet implementation details. This allows the portlets to be maintained as discrete components.

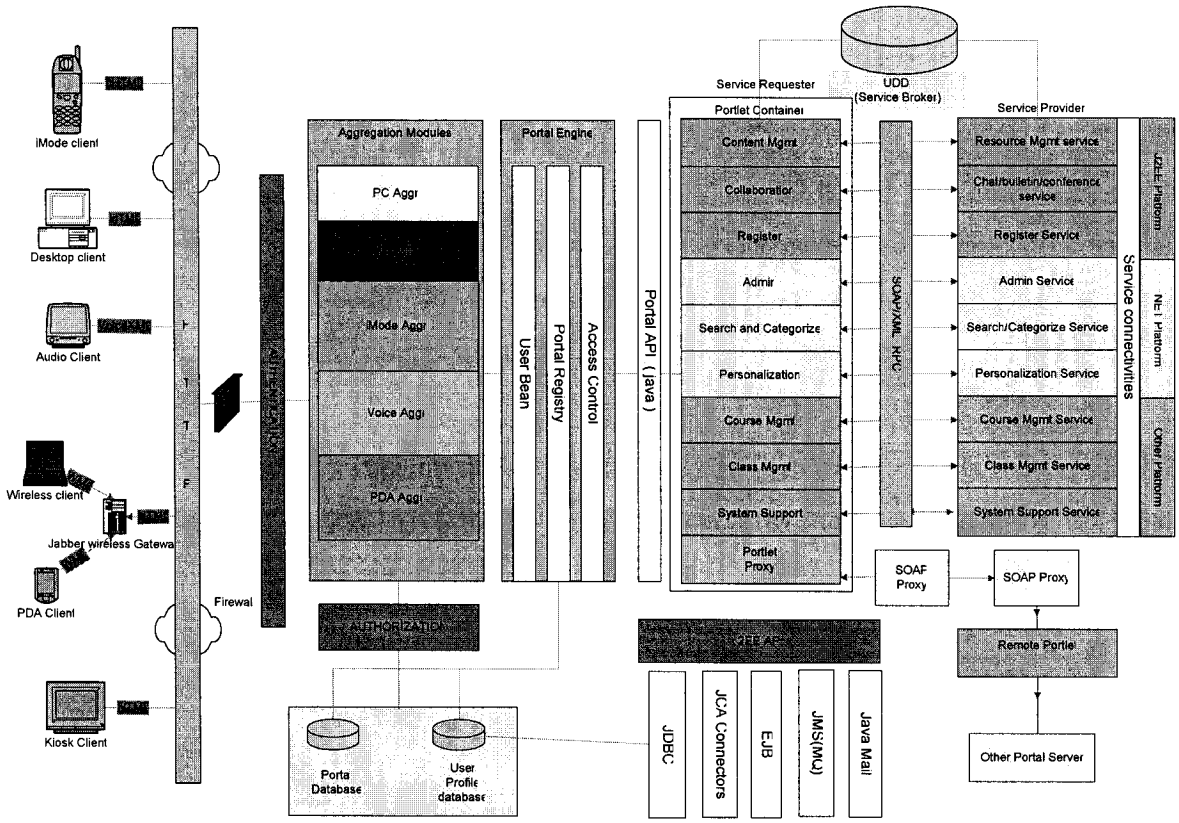

Figure 3-8: Ubilearn

In our learning context, a Service provider provides any learning information, material, or process as a self contained, self-describing modular service across different platform (J2EE, dot net and others). It could contain one or more of the following:

- Learning Objects and Learning Contents: are reusable learning resources as defined by SCORM Content Model [53] and can be any content stored in a database repository or a file system [55].
- Learning Management Applications: can be any program or function considered as shareable services by businesses or educational institution such as register management, course management, and so on.
- Learning Collaboration: includes both synchronous and asynchronous communications among different parties. Synchronous communication could be instant messaging/chat, shared whiteboard, or teleconferencing, etc. Asynchronous communication could be email, discussion forum, or news group, etc.

A Service broker provides a universal Web services registry (UDDI), in which service provider and service requester can publish learning services and find desired learning services respectively. Once the learning objects, contents, and applications are built as services, they are described by a WSDL document to let requesters know how to invoke them. Then, the WSDL file and XML metadata of the services are wrapped in a SOAP message that will be sent to the UDDI. All the necessary information for discovery is registered in the UDDI with directory and key word supports, just like a "Yellow Page".

A Service requester can be any application that requests a particular service. Based on the binding information of the services found in UDDI, the service requester can directly contact the services regardless of what kinds of platform used in each side, since SOAP is the standard communication protocol. In our framework, service requester can be a local portlet or a remote portlet in a portal e-learning system, which needs some specific services to integrate learning contents, remote course registration, and shared whiteboard together.

Chapter 4 Role-Based Portal Access Control Framework

4.1 Access control requirement for Ubilearn

4.1.1 Access control mechanism criteria

To choose the appropriate access control mechanisms for our Ubilearn, we must follow the following preliminary steps to help expedite and clarify the design process [32]:

- Clearly outline the types of role driven functions that must be implemented.
- Determine the relative interaction between data owners and administrative users.
- Specify the process for granting and revoking user access control rights on the system, whether it is a manual process, automatic upon registration or account creation, or through an administrative front-end tool.
- Try to quantify the relative value of information to be protected in terms of Confidentiality, Sensitivity, Classification, Privacy, and Integrity related to the organization as well as the individual users.
- Try to support access control mechanisms that, as closely as possible, adhere to the organization's security policy, such as acceptable time of day of certain data access, types of users allowed seeing certain data or performing certain tasks, etc.

Generally, a single case study is not sufficient to motivate the design of an access control mechanism. On the other hand, considering applying the principle and mechanism to satisfy the need for a typical example is instructive. To be able to accurately describe the access control model listed in Chapter 2, an ideal mechanism must fulfill a number of generic security criteria such as the following: conciseness, clarity, being aspect-oriented, fundamentality, being positive, level of need-to-know, and efficiency [34].

- "Concise" means that we must express the constraints and access rules in an easy and specific way. In general, mechanical repetition or extraneous effort in the expression of the rules is indications that something is amiss.

- "Clarity" is the second most important criteria in that it should be apparent what rule is saying what and whether it is correct, where access rules or constraints must be expressed and checked against corresponding conditions or situations.
- For a component based system, "fundamental" means that access control should be integral to the middleware rather than an optional add-on. One advantage of this is that developers are forced to address access control questions from the very start.
- "Positive" means that an access right should be expressed in terms of what a subject is allowed to do rather than what he/she is not allowed to do, which ensures that the default is that no access at all is allowed and that permissions must be explicitly listed.
- A strict "need-to-know" approach to access control means that users have a right to expect that no more of their data is being revealed than is absolutely necessary for a particular service.
- "Efficient" means that we must keep the overhead at a reasonable level in performing access control checks.

4.1.2 Analysis of access control for portal framework

A framework, as a reusable, semi-complete application that can be specialized to produce custom applications, provides developers with a set of backbone components that have the following characteristics:

- They are known to work well in other applications.
- They are ready to use with the next project.
- They can also be used by other teams in the organization.

While portals can bring an organization many needed benefits, they can also pose new IT challenges. Many of these challenges relate to the overall process for provisioning system resources and privileges to users who must access a portal. Therefore, designers of portal applications must address specific issues such as:

- How do new users get provisioned into the portal?
- What portal applications can each user view?
- How can a consistent approval process be used for granting portal access?

- How can identity information be synchronized across distributed applications and systems?
- Is there a mechanism by which the users can change their password through the portal?
- What happens when a user's privileges need to be revoked?

These challenges have common themes: access control and identity management. To manage the complexity of security administration in distributed systems such as e-learning, e-business and e-government, access control has become one of the most challenging issues. Access control systems usually provide authentication, authorization, and administration. Authentication is a process in which users are challenged for identity credentials so that it is possible to verify that they are who they claim to be. Once a user has been authenticated, authorization determines what resources a user is allowed to access. Administration refers to the ability to add, delete, and modify user accounts and user account privileges [31].

Under the RBAC portal framework, users are granted membership into roles based on their rights and responsibilities. The operations that a user is permitted to perform are based on the user's role. User membership into roles can be revoked easily and new memberships established as job assignments dictate. This simplifies the administration and management of privileges; roles can be updated without updating the privileges for every user on an individual basis.

When a user is associated with a role: the user can be given no more privilege than is necessary to perform the job. This concept of least privilege requires identifying the user's job functions, determining the minimum set of privileges required to perform that function, and restricting the user to a domain with those privileges and nothing more, which is not applicable to the e-learning portal application.

Furthermore, when implementing the Ubilearn system, we adopt principles of separating rules and roles from services' functionality. In this way, when applicable rules of processing logic have been changed at any time, some parts of the project do not need to be changed if

rules have not been embedded with the application, where a service is indifferent to who will use it.

As our Ubilearn system is implemented with portal and Web services techniques that combine service components together, we can apply access control on two levels. One is to apply role-based access control on portal resources like pages and page groups or on individual portlet by basically determining which portal-users are allowed to see which pieces of content. The second utilizes action-driven, condition-based access control behind each portlet interface using a Web services gateway.

An E-learning system running on distributed environment, especially implemented with techniques of Web services, portal, and WSRP, would also need to consider this issue. Each educational institute has its own regulations for every aspect of its functionality, such as course registration management, accessing course material, marking assignment, testing and grading, and teaching evaluation. Thus, extracting those "who or how to use" policies from functional components play an important role in making those e-learning service components portable, maintainable, flexible, and usable by any educational institute.

Therefore, the goal and purpose of the research is to create an e-learning portal framework embedded with role-based access mechanism. We create portal application interfaces with respected to the roles within the whole system. Thus, we do not need consider who will be the user of a service component when we develop it. We only need to create a portlet interface for the service component and integrate the interface into the corresponding portal. Moreover, every service in an E-learning system is independent from the other, and it is necessary that we implement an action-driven, policy-based access control (AD-PBAC) model to satisfy the personalization and collaboration of the distributed e-learning system. We explain the AD-PBAC model at Chapter 5.

4.1.3 User identity management

In the simplest terms, an identity is some method that verifies a user who distinguishes him/herself from others. The identity interacts with a process that determines what level of

security the user may possess and which services or activities the user may participate in after completing an authentication challenge. Associated with this process is a set of global attributes that contain information about the user that might be the user name, email address, pin/user password, credit card number, or Social Security number.

Identity management is becoming an important technology for managing users, accounts, passwords, provisioning processes, and deactivating processes for enterprise-scale organizations. Identity management typically lets agency administrators define user roles and grant access to resources according to tasks a user is allowed to perform. Identity management is a shared platform which provides consistent processes for managing information about users: who they are, how they are authenticated, and what they can access. Usually, an identity is created when a user initiates a relationship and gets an initial set of rights or privileges. As the relationship grows, rights may be added or subtracted, with preferences and affinity groups modified or removed.

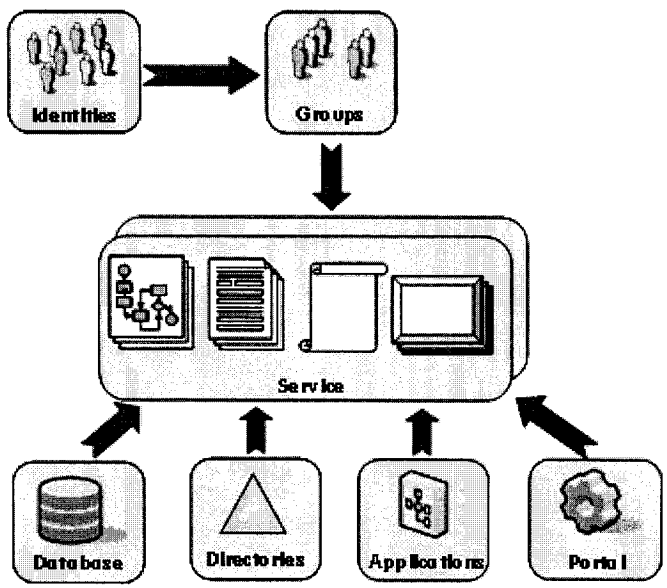

Figure 4-1: User identity management

The user identity management (UIM) within Ubilearn provides several features for user registration, role management, user account provision, service designation and notifications management, and self-services such as password management [Figure 4-1] [41]. The user

registration allows any new user to register to the system by choosing a user-id and password; the role management provides a functionality to an administrative user assigning a role to the user who has been registered to the account that has not been activated; the user account provisioning capabilities automate the creation, update, and deletion of accounts on distributed systems across the enterprise; service designation and notifications management allows high-level users to define access policies for low-level users. The UIM also provides self-service capabilities, enabling users to initiate access to specific services, change their password, or update their identity information.

4.1.4 Single sign-on

Single Sign-On (SSO) is a key feature of the Enterprise Portal that eases user interaction with the many system components available to the user in a portal environment. Once the user is authenticated to the enterprise portal, he/she can use the portal to access external applications. With SSO in the Enterprise Portal, a user can access different systems and applications without having to repeatedly enter his/her user information for authentication.

To apply SSO with WSRP, SSO allows the propagation of user identity from the consumer to the producer. That is, the consumer provides authentication, and the WSRP stack ensures that the same user identity is established on each producer with which the user is interacting.

Moreover, by utilizing signed certificates, producers authenticate consumers through the use of client certificates in conjunction with SSL/TLS. Therefore, if you are relying on SSO and allow users to log-in to the Consumer portal, as recommended, the Producer must trust that Consumer. To establish this trust, the Consumer needs a certificate of authentication signed by an approved certificate authority (CA), such as VeriSign, Inc. The Java keytool utility can be used to generate a self-signed certificate.

4.1.5 Generic actors and their targets

The main components of Ubilearn e-learning community can be classified by roles of users and functionalities, as UML use case diagrams show in Figure 4-2 through 4-5. For administrators, components include managing user profiles and user roles, updating user

identity to LDAP directory and controlling access policies; for instructors, components include course creation, teaching material management and assignment publishing, quiz question generation, automatic grading and assessment; for learners, components include course registration, course material reviewing, and assignment uploading and online testing; and for anonymous users, we can have components as bulletin board, forum, and news publish board.

The shared components include "my contact", "my task", "my email" and "my notebook"; the scopes and types of e-learning content can be chosen from basic Web pages and documents to fully interactive courses, events, assessments and simulations; for personalization realm within e-learning context, content is formatted to suit individual needs and preferences.

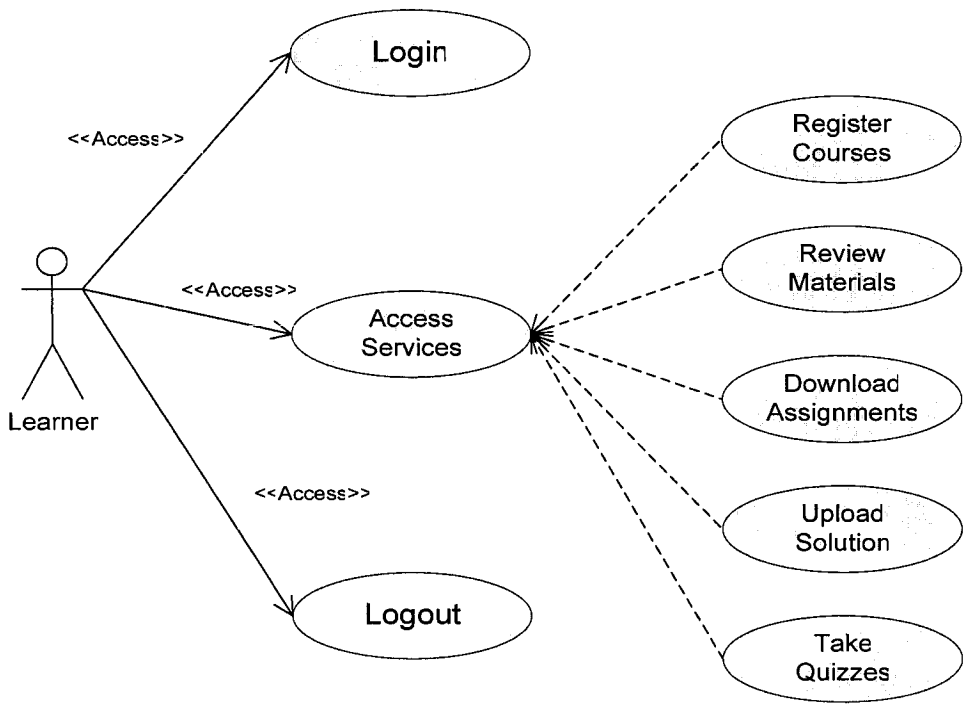

Figure 4-2: Learner use case

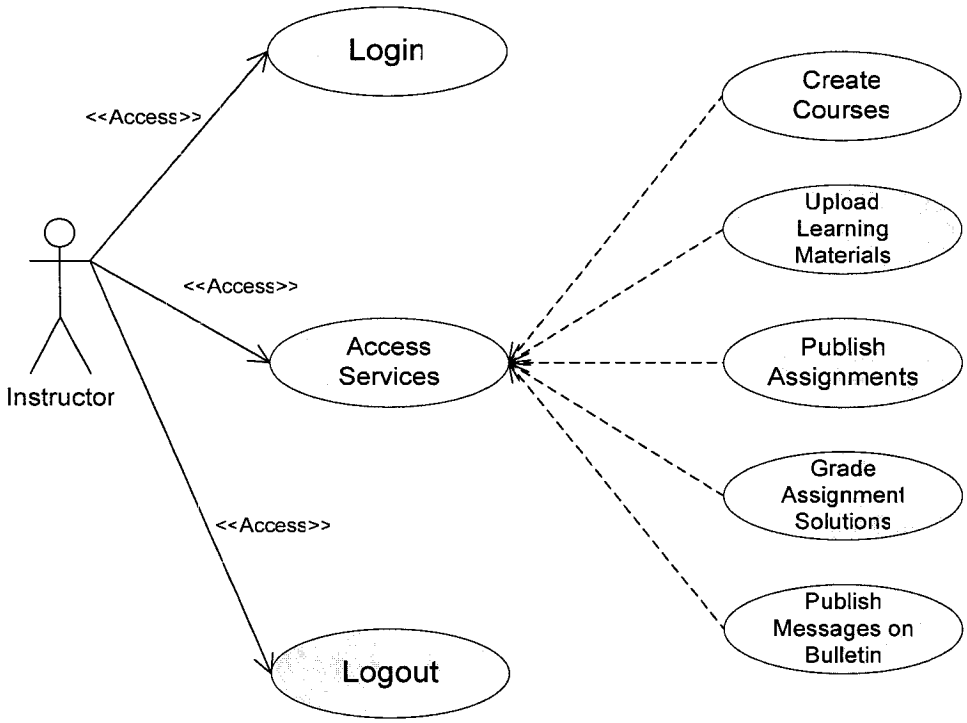

Figure 4-3: Instructor use case

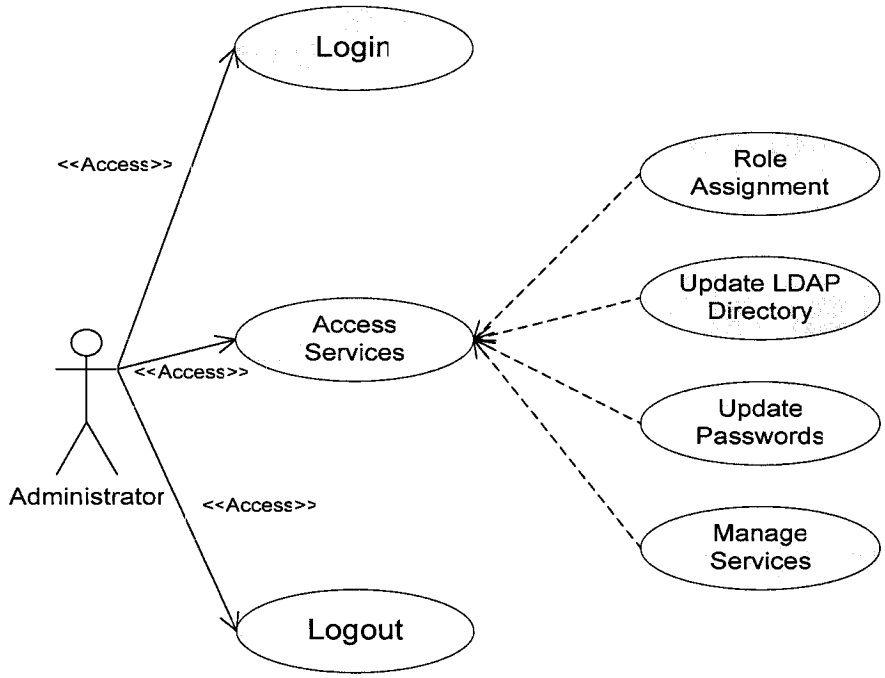

Figure 4-4: Administrator use case

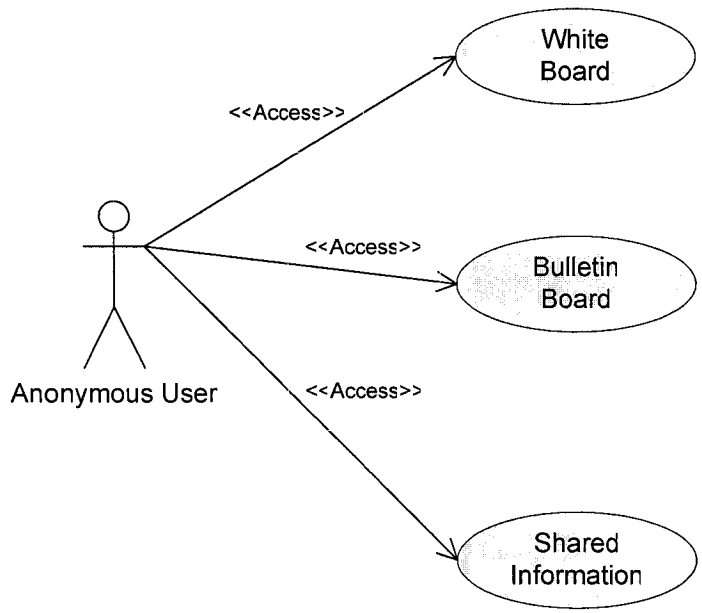

Figure 4-5: Anonymous use case

4.1.6 Manage users and groups

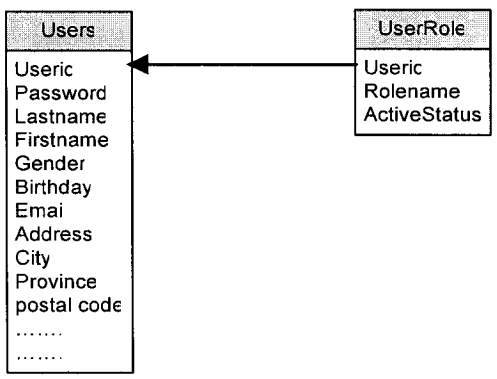

Figure 4-6: User and role entity relationship diagram

Generally, a user represents a person, system, or Java client; a group represents a static collection of users; whereas a role represents a dynamic collection of users. Obviously, the groups within any educational institute are relatively fixed, such as an instructor, learner or administrator; and each user must be assigned a role in order to access a service. Here, we do not differentiate between the definitions of role and group; for example, an instructor user who belongs to the instructor group is also assigned the role of instructor. First, we create tables, Users and User-Role within a database as a user identity repository. When a user

profile has been entered, the placement of a user in a group, or a role, rests with an administrator. Also, we can add an attribute to UserRole table to indicate whether the user's role is activated or not. The entity relationship diagram is shown in Figure 4-6.

To implement our Ubilearn as a portal application, we utilize a group-based approach as a default way of controlling access to a portal. Group-based access control is typical of many portal products; it is most useful when the user group structure changes infrequently, and the access to portlets and pages maps well to user groups. Although Database and LDAP can fulfill the task of storing and retrieving user-role data, relational and other databases exercise extreme measures to ensure that data is consistent during write/update cycles by using transactions, locking and other methods.

4.2 System prototype

Services within a generic e-learning system can be classified as functional and non-functional (i.e. system-supporting) services. Functional services include creating a course, uploading course material, publishing assignments and grading assignments, registering a course, reviewing course material, submitting assignments, taking quizzes, registering user profile, assigning roles to users, and activating users' accounts. Non-functional services include personalization, chatting, directory, internationalization support, and integration services.

To make an E-learning system more robust in terms of flexibility, portability, and maintainability, we design and implement those functional services independent from other restrictions of roles and rules, where the service itself does not care as to who will use it. Based on the requirement analysis of access control for Ubilearn system, we can construct a functional system utilizing portal principle and Web services techniques to fulfill the rule and role based access control. Figure 4-7 illustrates the mechanism on how we organize the services into different portals.

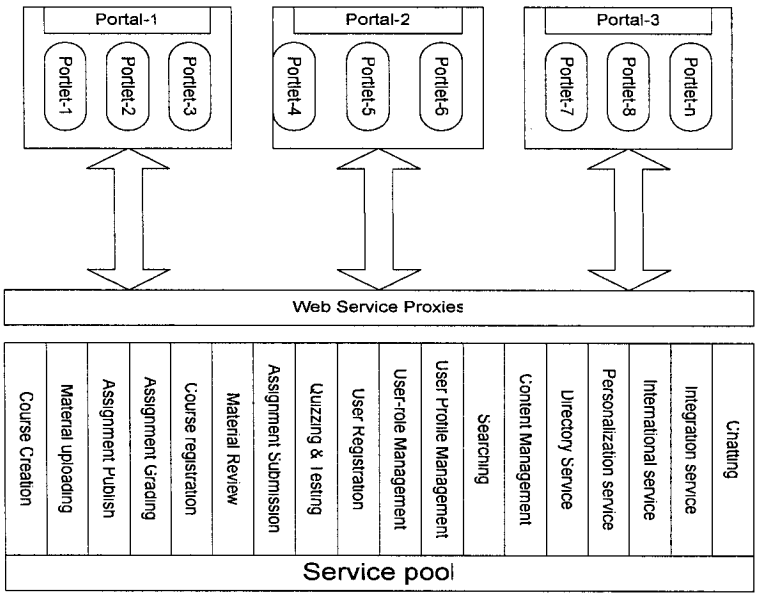

Figure 4-7: System prototype framework

In order to reach the purpose that once the user's role is acknowledged upon his/her login, we realize the need for a protocol that system will direct a user, when browsing from the login page to the portal interface corresponding to the role of the user. Figure 4-8 shows the user-role-portal relationship to implement role-based access control on the Ubilearn portal application.

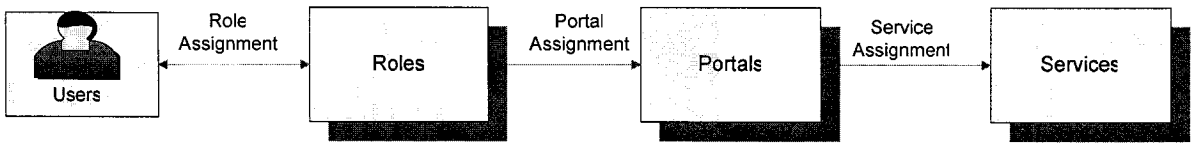

Figure 4-8: Role-based portal access control model

From Section 2.2, we describe the basic components of a simplified RBAC model. These are namely Users, Roles, Permissions, User-Role (U-R) relationship, and Role-Permission (R-P) relationship. In the role-based portal access control model, we replace the permission and role-permission relationship from RBAC model with portal, service, role-portal relationship, and portal-service relationship. By doing so, a user is only allowed to access those services belonging to a specific portal based on the role of the user.

We implemented our Ubilearn by utilizing Web services with portal technologies. Extending from Java Servlets, portlets can be used as a user interface connected to all services located on local or remote servers. A portal, being a container for portlets, represents a user interface for those accessible services which provide a single entrance for any user to access multiple and heterogeneous services through this point.

4.3 Design of Ubilearn role-based portal framework

4.3.1 Functional description

To solve the common challenges that e-learning faces on securing portal solutions with integration to back-end applications, we have implemented several functional components such as portal controller, user identity, and access management for users provisioning across the enterprise, permitting or denying access to resources based on the policies and users/groups who access the resources (authorization), determining who is accessing the site (authentication) and only requiring a single sign-on (SSO) to access resources to which they have been granted access; and an audit trail to ensure proper use of the system.

More specifically, we have developed several portlet applications for user and account management and approval of user provisioning requests by utilizing the JMX APIs and policies, to provide a user with an integrated SSO solution using an LDAP directory manager for authenticating once and access services within Ubilearn portal application, in order to manage user access control through role mapping with products that have access models.

The identity and access management includes such functional components as user registration, role assignment, new user creation, setting new members to a group matched to the role of the new user, and a portal controller to direct user's browser to the appropriate portal application that the user is entitled to.

4.3.2 Use case model

This section describes the actors in the use cases, presents the use case diagram, and summarizes the use cases. For the Ubilearn system, we define two types of actors, general users – including instructors and learners, access to the main e-learning functional services,

and administrative users – including administrators and managers, and finally, access to user provisioning components. Actors are defined in Table 4-1. The description of each of the use cases is shown in the Table 4-2. Figure 4-9 depicts the use case model that outlines how users will interact with the system.

Table 4-1 Actors

Actor	Description
General user (Instructor, or Learner)	All general users within Ubilearn system. All general users have to enter his/her profile information.
Administrative user	All administrative users within Ubilearn system. All administrative users manage all general user provisioning.

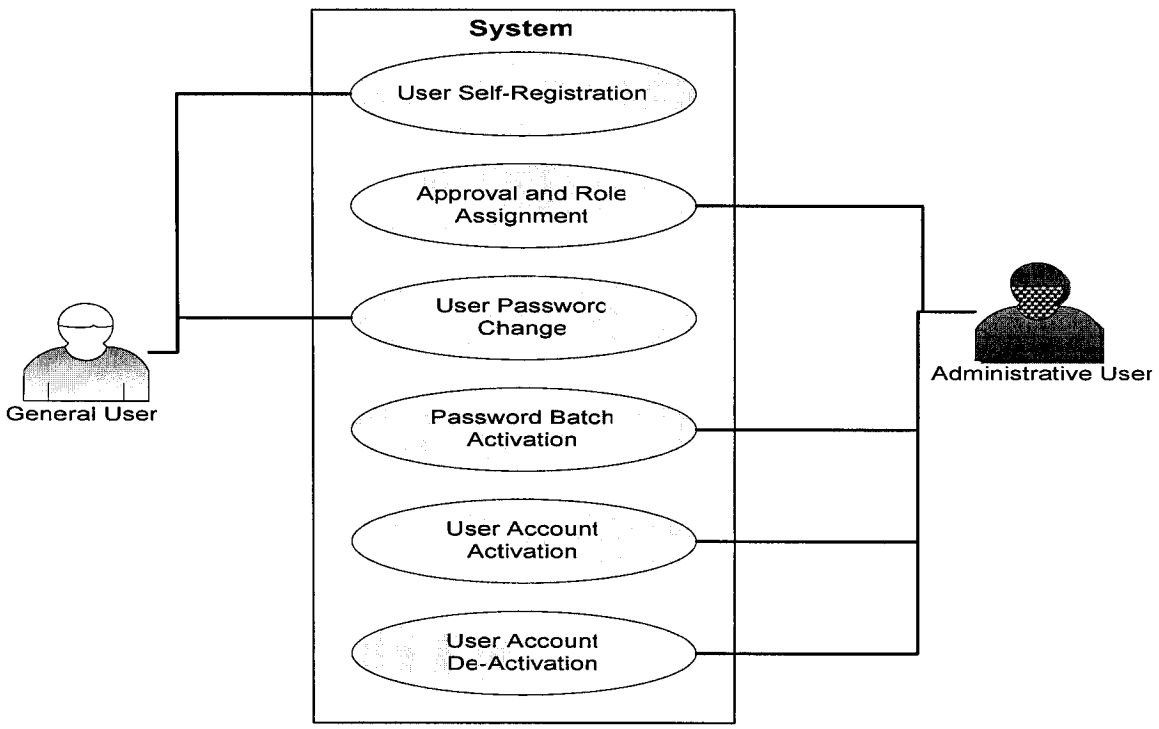

Figure 4-9: User provisioning use case diagram

Table 4-2 User provisioning use case description

Use Case ID	Use case name	Goal in context	Actors
UP01	User self-registration	Each new user have to enter his/her user profile information to get the identity pair (user-id and password)	General users
UP02	Approval and role assignment	Assign role to a newly registered user	Administrative users
UP03	User account activation	Activate user account by adding the user identity pair to the specific group in LDAP directory based on his role	Administrative users
UP04	User password change	After a user account is activated, a user can change the password	General users
UP05	Password batch activation	Update the password to the user identity pair in LDAP directory.	Administrative users
UP06	User account de-activation	De-activate the user account by removing the user identity pair from the specific group in LDAP directory	Administrative users

Table 4-3 Use case details

Table 4-3 (1) User self-registration:

Use case ID: UP01	UP01: User self-registration
Description	Any new user has to create the initial profile in the Ubilearn application. The user enters his/her name and personal geographic, contact, organization information. After which the user information is sent to "User registration service" to create the user's record and generate a user-id and password.
Precondition	The actor does not have a role. The actor's name does not exist in the user profile database.
Primary actors	General users
Secondary actors	None.
Main scenario	1. The new user navigates to the registration portlet with the login portal which includes login portlet and registration portlet; 2. The new user enters his/her personal profile for initial registration; 3. Registration service generates and displays the user's identity pair.
Alternatives	None.

Table 4-3 (2) Approval and role assignment:

Use case ID: UP02	UP02: Approval and role assignment
Description	An admin user selects and assigns a role a newly registered user.
Precondition	UP01
Primary actors	Administrative users
Secondary actors	None.
Main scenario	1. The administrative user login through the Ubilearn login portal (his user identity pair has been predefined.) 2. The Ubilearn controlling system directs the admin user's browser to the administrative portal 3. The admin user navigates to the "user role assignment" portlet and chooses a role, then chooses a user who does not have a role, and clicks submit button.
Alternatives	None.

Table 4-3 (3) User account activation:

Use case ID: UP03	UP03: User account activation
Description	An administrative user add users who have a role to the group in the LDAP server
Precondition	UP02
Primary actors	Administrative users
Secondary actors	None.
Main scenario	1. The administrative user login through the Ubilearn login portal (his user identity pair has been predefined.) 2. The Ubilearn controlling system directs the admin user's browser to the administrative portal 3. The admin user navigates to the "user account activation" portlet and chooses a role, then chooses users who have a role, and clicks submit button.
Alternatives	None.

Table 4-3 (4) User change password:

Use case ID: UP04	UP04: User password change
Description	A general user changes his/her password
Precondition	UP03
Primary actors	General users
Secondary actors	None.
Main scenario	1. The general user (learner or instructor) login through the Ubilearn login portal, after his user identity pair has been activated. 2. The Ubilearn controlling system directs the admin user's browser to the learner or instructor portal 3. The general user navigates to the "Change password" portlet and enters old password, then enters new password twice, and clicks submit button.
Alternatives	None.

Table 4-3 (5) Password batch change:

Use case ID: UP05	UP05: Password Batch change
Description	An administrative user updates general users' passwords to the LDAP directory
Precondition	UP04
Primary actors	Administrative user
Secondary actors	None.
Main scenario	The administrative user login through the Ubilearn login portal The Ubilearn controlling system directs the admin user's browser to the administrative portal The administrative user navigates to the "Password Batch Change" portlet and clicks submit button.
Alternatives	None.

Table 4-3 (6) User account de-activation:

Use case ID: UP06	UP06: User account de-activation
Description	An administrative user remove a general user identity from the LDAP directory
Precondition	UP04
Primary actors	Administrative user
Secondary actors	None.
Main scenario	The administrative user login through the Ubilearn login portal The Ubilearn controlling system directs the admin user's browser to the administrative portal The administrative user navigates to the "User account de-activation" portlet and chooses a user from the user list, and clicks submit button.
Alternatives	None.

4.3.3 Manage relationship between users and services

By analysis of role characteristics of a generic online institute, we classify Ubilearn users into three groups (roles): administrators, instructors, and learners. Moreover, we associate the group names with the virtual portals. Thus, we are able to categorize services by integrating them into those pre-defined portals according to the relationships between service functionalities and roles pre-assigned to users. In the Figure 4-10, represented by pre-organized names of services, permissions can be accessed by only the users according to their roles.

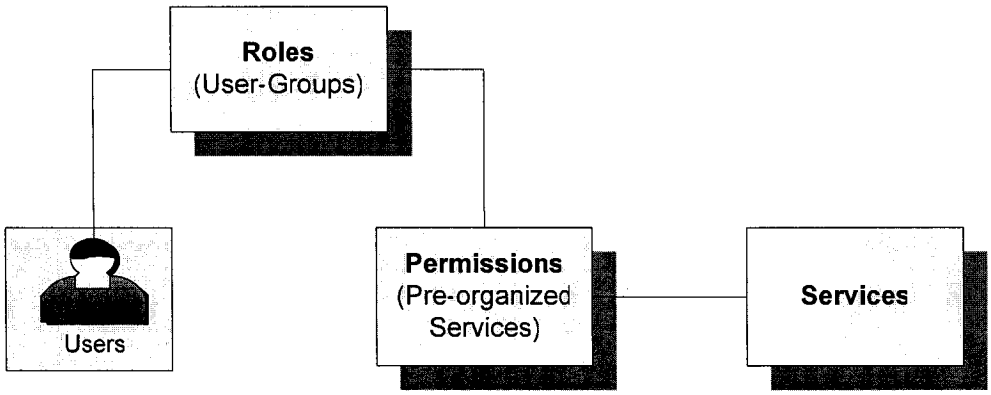

Figure 4-10: Relationships between users and services [1]

This portal access control mechanism provides excellent advantages by virtue of its functionality. Since each user is linked with what that user is allowed to access, once a user is authenticated and the user's role data is retrieved, the portal container directs the user's browser to the corresponding portal the user is entitled to. Another advantage depends heavily on the concept of inheritance; any change made to a group's permissions will automatically be inherited by all members of that group. Furthermore, changes to a group's permissions to access services can be made very easily by adding or removing portlets within that portal. Also, it is easy to accommodate the switch of an individual user to another group simply by removing the old role and assigning a new role.

4.3.4 Utilize LDAP in portal

However, the performance of Ubilearn system is a priority issue and LDAP is characterized as a 'write-once-read-many-times' service; using a simple asynchronous replication process, we take advantage of LDAP by mapping user and role relationship into a directory server for better user-role entries retrieval. Figure 4-11 shows component diagram for user identity and access management among the portal server and the application server.

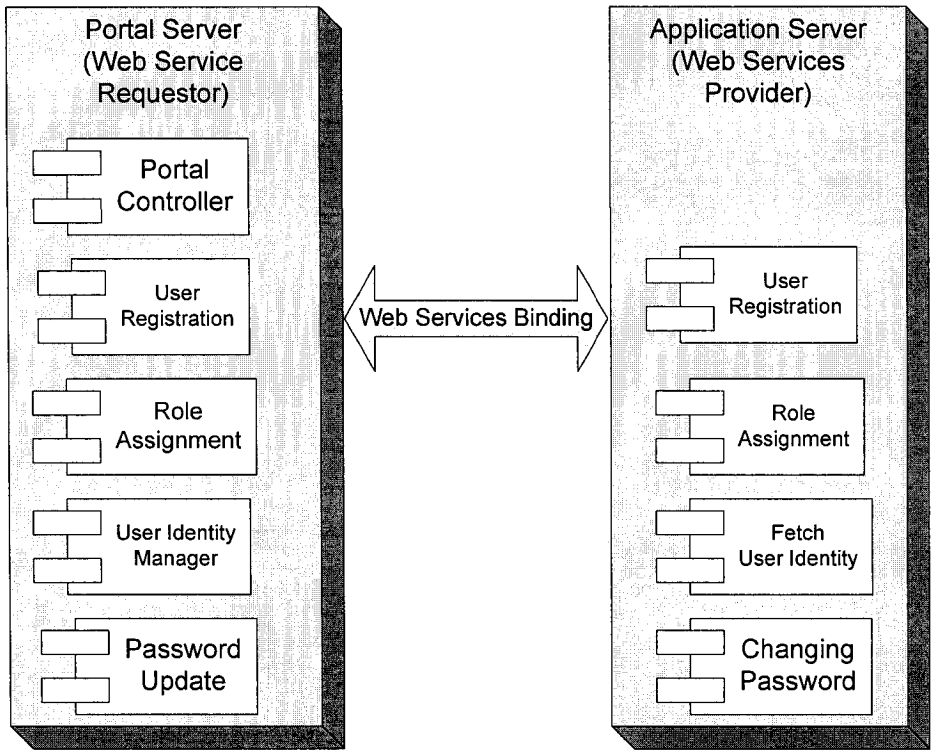

Figure 4-11: Component diagram for user identity management

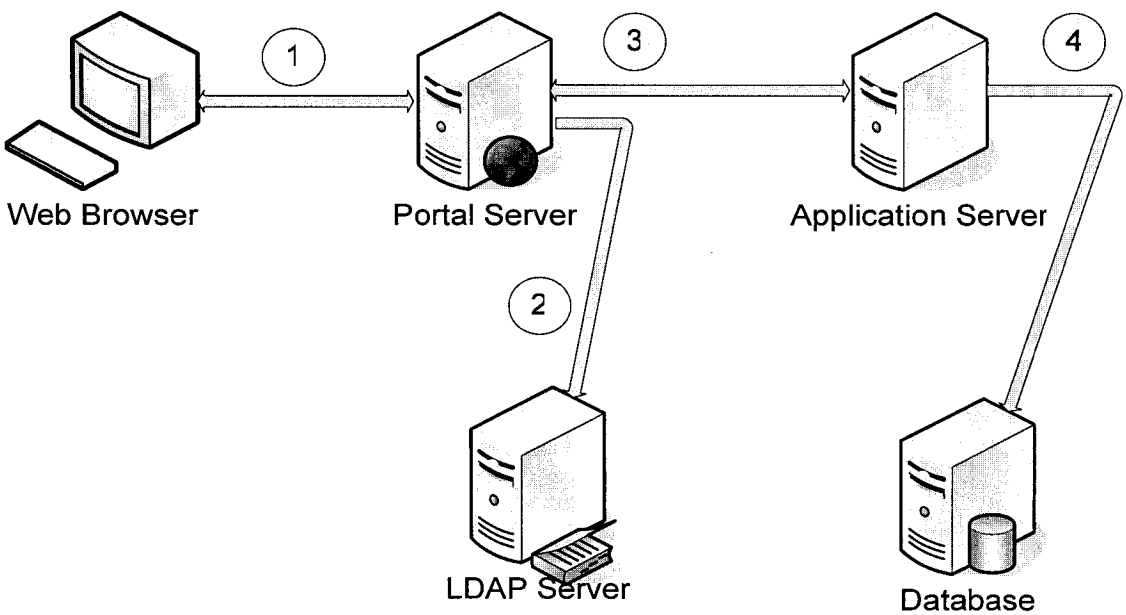

Figure 4-12: Servers working together

Figure 4-12 shows a general architecture of a portal server, application server, LDAP server and a database server working together. A brief description is provided below:

1. User enters credentials: When a user needs to access the E-learning management system, the user enters the Ubilearn Web site URL in a Web browser. The portal controller within portal server will direct the Web browser to the login portal where the user will use the login portlet to enter the user ID and password credentials.
2. Validate user credentials with embedded LDAP Server: The portal server will validate that the user credentials by communicating with the embedded LDAP Server.
3. Successful authentication and authorization: If the credentials are valid, the user will be authenticated to the system and then the Web browser will be directed to a portal corresponding to the role of the user, where the portal illustrates a group of portlets interfacing to those services that reside on the application server.
4. Manage user identity: When identity management is conducted, the identity management service will fetch the user identity data from Database and update data with the LDAP server through the service component with portal server.

Moreover, an embedded LDAP in a portal server container being provided by some portal development platform products makes it easier for developing portal applications. Figure 4-13 shows a portal server container.

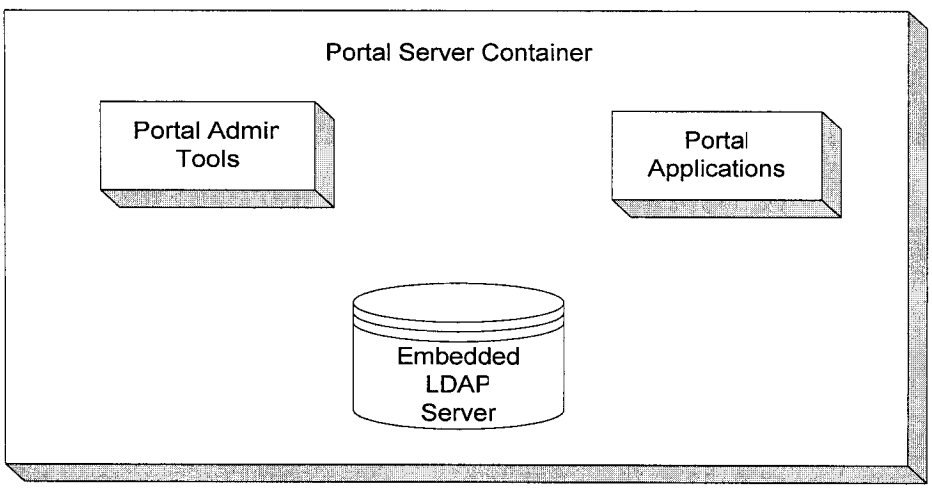

Figure 4-13: Portal server container

Portal authentication and authorization is based on a pre-defined Server Security Applications (SSA) implemented for a controlling access protocol. When a user logs into the portal, the SSA handles the authentication of that user through either the embedded LDAP database or an external authentication provider. Once authentication is complete, based on the pre-defined protocol, the SSA determines which portal the user can access based on the role of the user. In other advantageous implementation cases, we can implement a mechanism to determine what portlets the user can view.

4.3.5 Portal controller

The Ubilearn system provides single sign-on authentication by including a basic authentication and authorization flow for any user to access to the services within Ubilearn application. Figure 4-14 shows a UML activity diagram for a user login procedure under the Ubilearn portal controller, a top level component to control and direct a user's Web browser to the appropriate portal based on the role of the current user. Figure 4-15 show a diagram for portal controller.

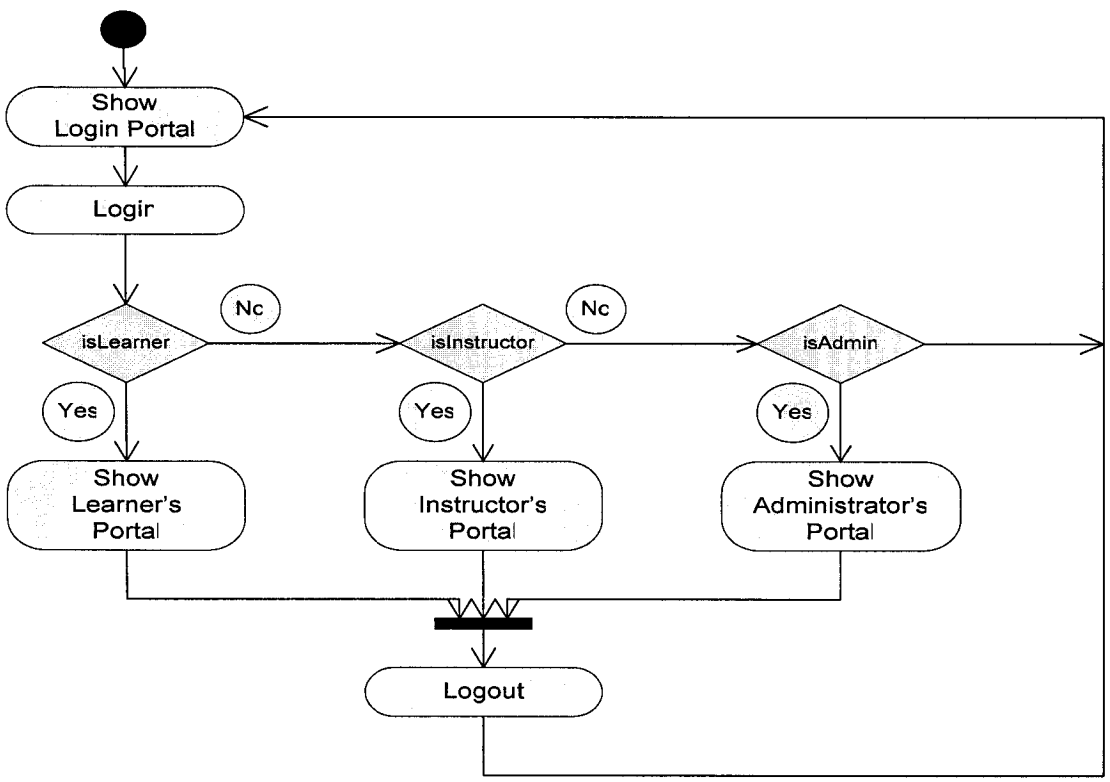

Figure 4-14: UML activity diagram

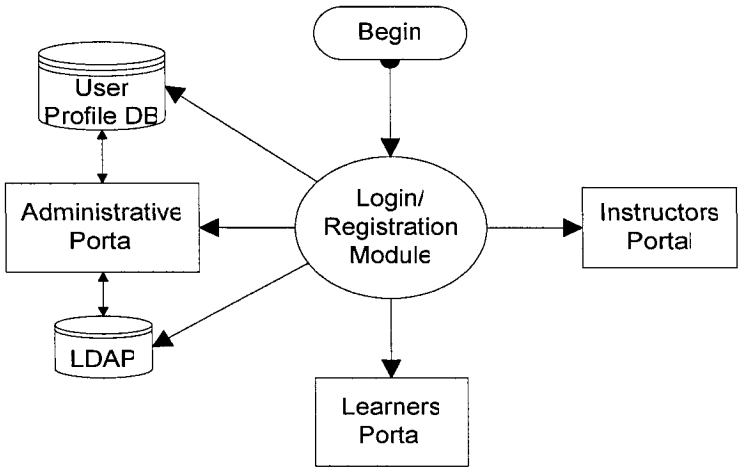

Figure 4-15: Portal Controller

4.3.6 User provisioning

In addition to assigning a role to a general user based on the user account ownership policy, the Ubilearn identity and access management keeps general user data stored in user profile database and keeps the data synchronized with embedded LDAP server by assigning and adding members to the specific group in LDAP directory. Once this action has been done, user account is activated.

More precisely, we utilize MBeans of JMX [42] as the communicative agents to place user identity objects into embedded LDAP server. Those operations include creating user identities, deleting user identities, changing user passwords, adding user members to a specific group, and checking whether a user is a member of a specific group.

4.3.7 Role-based portal framework for Ubilearn system

Based on the analysis above, we describe our research result as a role-based access control portal framework to integrate services' interfaces together. The Unilearn system is comprised of four layers: access control layer, portal layer, web service proxy (agency) layer, and service layer.

- The access control layer: A role based access control directs different users to different portals that include different portlets based on the user role. Once a user has been authenticated, authorization determines what resources a user is allowed access.

Administration refers to the ability to add, delete, and modify user accounts, and user account privileges.

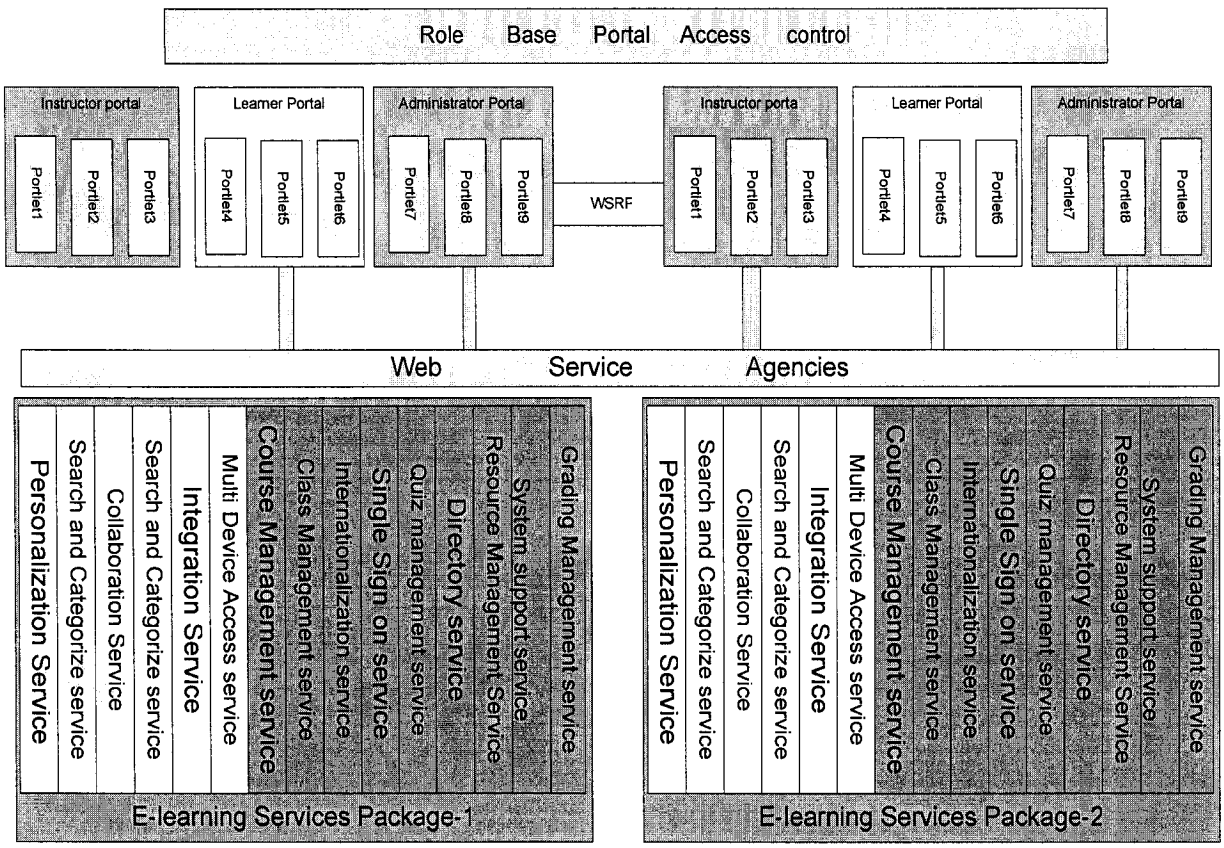

Figure 4-16: Portal access control framework

- The portal layer: this layer can be treated as a container of web services that consist of many local portlets called service requesters and some remote portlets. A learner portal can call a remote portlet from another e-learning system as its portlet through web services for remote portlets to share system functionality. Being integrated with a Web services-proxy, a portlet can call a Web service inside its system or the other e-learning system. Figure 4-16 shows two Ubilearn systems where both carry Web-services components with exposed Web service proxies via portlets placed in the portal containers. Users of Ubilearn-1 can access services provided by Ubilearn-2 through WSRP. That is, WSRP allows portals to display remotely-running portlets inside their pages without requiring any additional programming by the portal developers. Here, WSRP exposes UIs from remote components located in remote

domains, relieving the burdens of a system administer that are caused by having all service components located in the same service domain. Access to such components must follow the rules located in the remote domain. Moreover, both Producers and Consumers of WSRP can control access by using the implemented security measures. Consumers can restrict end-users' access to portlets and limit specific operations within those portlets. Producers can implement access control mechanism programmatically through the use of facilities such as an authenticated user identity.

- The web services proxy (agency) layer is a service engine that provides a pure Java engine whose main responsibility is to aggregate content from different sources and serves the aggregated content to multiple devices. It also provides a framework that allows the portlet layer of the portal to be decoupled from the service implementation details.

- The service layer provides a set of Web services' components to be accessed through the SOAP protocol.

When a user logs in with a user ID and password, access control will verify the user ID and his/her user role and direct the user to the relative portal. A user can personalize his/her portlet through WSRP and locates some portlet interfaces that he/she prefers. The web services agencies then receive the request from the portlet and help the portlets to search the proper web service. Then web services agencies send the response back to portlet, if they find the relative web service.

Our framework enables a true flexible integration among educational institutions or businesses.

4.3.8 Login scenario

We have two methods for specifying a role to the user: one is to link the user's profile to a static role manually; the other method is to link the user's profile to roles automatically based on the user profile attributes. We apply the first method to our role-based portal access

control model and explain it as follows. For example, when a user, let's say, Jill, browses our Ubilearn welcome page via the "Login/Registration Module", she can use either registration, if she is a new user, or login, if her identity has been already activated. Suppose Jill is a new user and she wants to access services and learning resources as a learner, she has to register and let an administrative user, lets' say, John, link her profile to a role before she can access those services through a portal.

Upon Jill's registration of her profile (which stored in the user profile database), John will link the user profile to a role she is entitled to and he will update the identity of Jill into the LDAP server; so that Jill can access those entitled services via a portal. Once Jill enters her user-id and password pair, the login module will check with the LDAP directory to see whether Jill's credential are matched, and if yes, the login module will direct Jill's browser to the learner portal, where a group of portlets connect with those corresponding services and through those services, Jill can access her expected learning resources.

Chapter 5 Action-Driven Access Control

5.1 Background

Traditionally, access control has been expressed in terms of read/write access. It is applicable for the standard file system and for the attributes associated with a record in a database system. However, a service component in a distributed environment only exposes one or more interface methods and encapsulates its details within it, and each method represents a pre-defined operation, deeming operations more complex than just a matter of read and write.

Moreover, group or role-based access control is not sufficient if access to portlets or pages is dependent on a more complex set of conditions than simply the group membership, or if it may need to change dynamically, based on changes to the user profile, without the involvement of an administrator. With traditional group-based or role-based access control, the placement of a user in a group or a role rests with an administrator. Where a more dynamic, flexible entitlement behavior is needed, policy-based entitlements become helpful. With policy-based entitlements, the business policy behind an entitlement becomes a self-documenting business policy that governs whether a user is given access to services through portlets or portal pages in a portal interface. Here arises an access control problem to prevent unauthorized users from invoking methods for which their access parameters do not satisfy some specific conditions.

In Chapter 4, we described the portal access control approach utilizing the role-based access control model on Ubilearn portal to fulfill the authentication and authorization pertaining to the role of users. Next, we discussed issues that arise from restricting users' actions based on the environment parameters.

As we stated in Section 3.3, the functionality of each service is independent of the restriction of rules and roles. Thus, to restrict users' access to specific services or actions within each of

the services, it is necessary to apply an access approach that associates policy-parameters with each action.

Generally speaking, policies are rules of conduct that characterize the behavior of a group of people involved in a certain enterprise. Accordingly, a policy refers to the actions being regulated, the role of participants that engage in such actions, and the applicable conditions.

Since all services being integrated within a portal exposed to a user, and each service includes several actions, there arises a dilemma: for we do not know which service or action the user will request and we cannot define the access order for the user, nor can we assign the accessing rules within the user session. The policy-assignment information is much bigger if too many services are integrated.

5.2 Action-driven policy-based access control

Here we propose an Action-Driven Policy-Based Access Control (AD-PBAC) approach, a modification of policy-based access control model, by combining role-based access control models, as the solution for the issue mentioned above. Our goal of the AD-PBAC approach is to simplify and automate the administration of IT environments by allowing the system administrator to specify only the conditional parameters that are to be met for one or more services, rather than having to know the internals of the system and how the actions and service are to be executed or accessed. By associating conditional parameters with role assignment, this approach enriches and compensates the defects of the role-based portal access control approach. Those condition sets can be defined as XML segments to represent access parameters such as, the location of all end-users, their roles, the service (or service sub-functions) they want to access, whether to apply time restrictions to the time periods, etc. All condition-sets are stored within an XML file independent from platform.

We implement the AD-PBAC as a Web services component in that it can be accessed by any type of application, either from other Web services components, or server-side or client-side application. The AD-PBAC component consists of three sub-components, an XML file to define a condition hierarchical structure, an XML parser to read the XML file, and an action

evaluator to compare the pre-defined XML policy corresponding to the user's action. Figure 5-1 presents system diagram and Figure 5-2 the component diagram of AD-PBAC model.

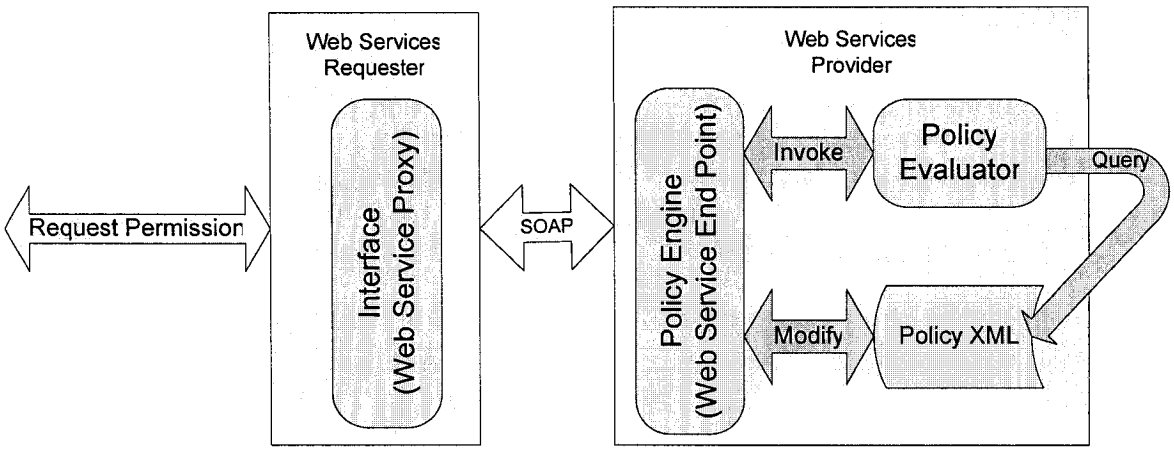

Figure 5-1: System diagram for AD-PBAC

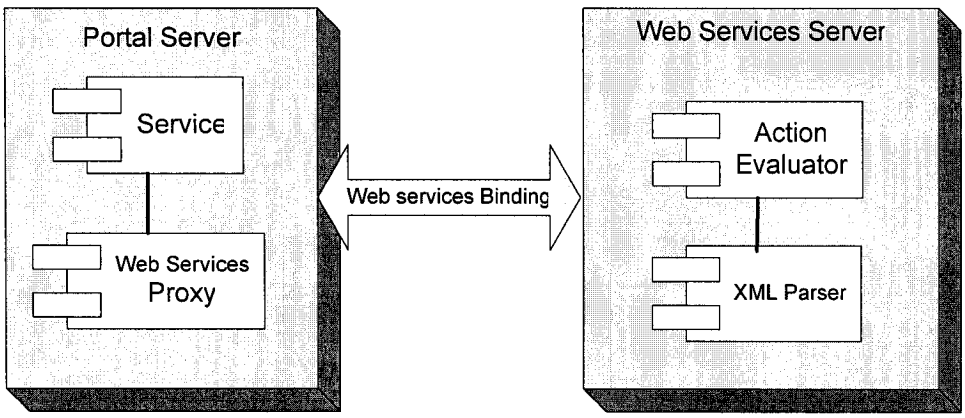

Figure 5-2: Component diagram of AD-PBAC

A typical access control policy is defined by three elements: a service identifier; a set of parameter specifications, represented by parameter-name and parameter-value-range; a set of conditions against service identities.

5.2.1 Policy-parameter definition

To define a policy-set for each of actions invoked by the client application, the following parameters need to be considered based on our discussion in Chapter 3. Although the AD-PBAC model can deal with any condition parameters associated with the actions or services

in Ubilearn system, we focus on defining those parameters based on the environment attributes and user attributes such as action-identifier, role, domain, location, and time. Here,

- Action-identifier: identify the actions uniquely, such as service name, or action id;
- Role: defining the actions and activities assigned to a person or group
- Domain: representing what department or unit in Ubilearn system a user works, for example, an instructor who belongs to Computer Science Unit is not allowed to create a Medical course under the domain.
- Location: representing where the user is accessing services from. Location information is usually used in several types of rules and policies. One type uses location to identify the trust domain, where the user is accessing information services from. A reasonable policy would deny access to any sensitive information to anyone accessing it from such areas, off-campus, for instance.
- Time: time constraints specify the validity periods for a policy.

5.2.2 Workflow description

In order to find all possible policy parameters, we examine the services and actions of our Ubilearn system with operational environments. Ubilearn is a Web services-based portal system, in which individual operational service is implemented as a Web services component, interfaced as a portlet, and integrated within a portal. The implementation of the action evaluator follows the module of Ubilearn. Figure 5-3 presents a sequence diagram for the AD-PBAC approach.

For each action, we define an action-id (service-name) which corresponds to a policy segment within the XML policy file. The procedure from requesting action evaluation to returning permission in the sequence diagram of AD-PBAC lists as follows:

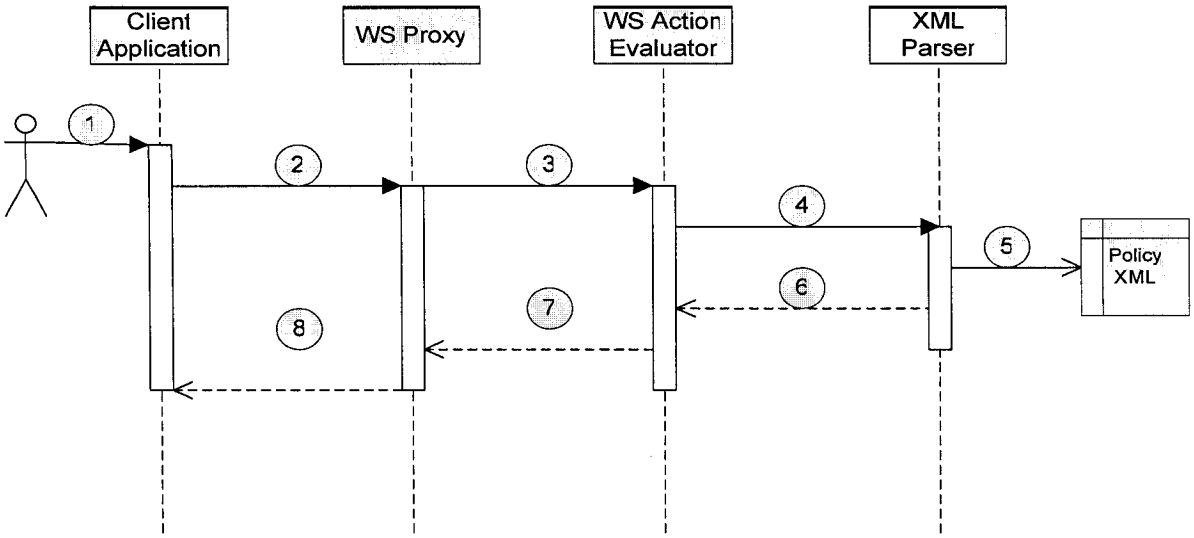

Figure 5-3: Sequence diagram of AD-PBAC approach

1. When a user invokes an action within the client-side application;
2. A method calls to Web Service Proxy indicating the action-id.
3. Web services Proxy generates a method call to Web services Evaluator with submitted attributes such as action-id, the role of the current user, caller's IP address and time-restriction stamp;
4. The WS Evaluator calls the XML parser;
5. The XML parser reads and parses the policy XML file to fetch the policy segment matched with the action-id.
6. The XML parser returns the policy vector to WS Evaluator, which will compare the submitted attributes with the policy vector.
7. If matched, the WS evaluator returns "True"; otherwise it returns "False".
8. WS proxy in turn returns the decision of permission or denial based on the return value.

Figure 5-4 shows an UML activity diagram for a general user to access a function from a portlet interface (within a service) guarded by an evaluating policy applied to the user.

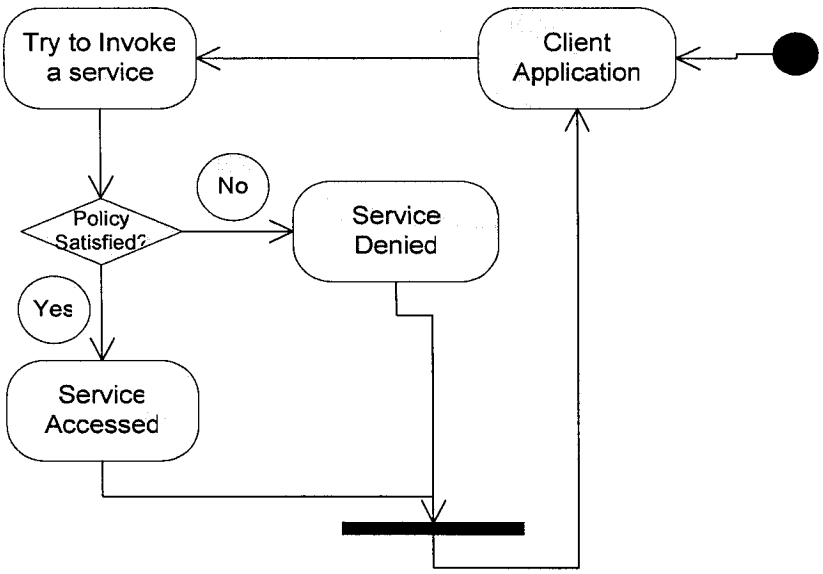

Figure 5-4: UML activity diagram for a general user

5.2.3 Scenario

Suppose a student Jill wants to use the Chatting Service in Collaboration Tool (CSCT) and talk with her Professor Ron. When an action request is sent to the CSCT, the CSCT will invoke a call to the AD-PB component with several conditional parameters, including action name "CSCT", the role of user "Student", the IP address within campus, and the time.

Table 5-1 Snippet of the XML rule for CSCT

```
<Rule Service="CSCT">
  <Condition RoleName="professor">
    <parameter name="ipaddress" datatype="string" operation="Greater" value="000.000" />
    <parameter name="starttime" datatype="string" operation="GreaterOrEqual" value="18" />
    <parameter name="endtime" datatype="string" operation="LessOrEuqal" value="20" />
  </Condition>
  <Condition RoleName="student">
    <parameter name="ipaddress" datatype="string" operation="StartWith" value="137.122" />
    <parameter name="starttime" datatype="string" operation="GreaterOrEqual" value="18" />
    <parameter name="endtime" datatype="string" operation="LessOrEuqal" value="20" />
  </Condition>
</Rule>
```

The AD-PB component will check the conditional parameters requested against the XML rule defined with the component. For example, suppose Professor Ron appoints the Chatting

time period between 6pm to 8pm and he already creates a chatting room for a specific course through the Chatting Service during appointed time frame. If Jill logins to the Ubilearn system and click the Chatting Service within Collaboration Tool, say at 6:15pm, within the campus, then she can talk with Professor Ron.

Since the AD-PB component does not retain those session information; thus when the time is out of the appointed time frame, the chatting will be interrupted until the CSCT checks with the AD-PB component again.

Charter 6 Conclusions

6.1 Related work

In Chapter 2, we described a variety of access control models such as access control list, discretionary and mandatory access controls, user-based access control, team-based authorization [11] and subject-object access controls, rule-based access control, and policy-based access control. However, those models are not appropriate to solving our issue mentioned above, especially for a situation that involves crossing university consortium, where e-learning systems are short of a user identity management mechanism integrated to provide access control services. Only few e-learning systems are architected specifically to satisfy the necessary requirement from users in any educational system, and supply sufficient learning and reusable services and material. Here, we briefly describe two of them.

6.1.1 OASIS and RAED system

One of the access control mechanisms is named "The Open Architecture for Secure Inter-working Services (OASIS)" [49] [50], an implementation of distributed RBAC to support parameterized policy elements and their definitions, as well as session-based distributed operation utilizing a given role credential with attributes. That is, the role activation and privilege authorization rules are parameterized, which takes operations such as database lookup, or temporal checks. Those policy rules are specified in simplified clause logic and implemented in XML format.

Unlike many RBAC models including a notion of privilege delegation, the OASIS architecture employs the more expressive and secure notion of prior arrangement. This prior arrangement models the intuition that the principal-A can indirectly authorize other principals to perform tasks principal-A does not have the privileges to perform within the OASIS network. Distributed OASIS services share users' authentication credentials through SOAP and X.509 certificates over HTTPS connections.

Among the few, the Role-Based Access Control for the Evolution of Distributed Courseware (RAED System) [51] provides a role-based access secured infrastructure for globally

distributed electronic courseware. The client-side presentation of the RAED system is demonstrated as a set of pages within a browser. The pages are dynamically generated, based on a set of rules for each role coupled with results from database queries [56]. For example, when a student logs in, the system will be presented with pages according to which course he/she is enrolled in, and any related material to which the student is allowed access as specified by a local tutor.

The RAED system utilizes a database driven mechanism to generate Web pages dynamically on the basis of requests sent from browsers. That is, the HTML code is compiled by a server-side set of programs. In this way, materials from several distributed sources are completely encapsulated so that system users see a single unified Web page; even a single Web page has combined information from distributed sources across two or more domains. This system is implemented with server-side technologies and database access technology on databases.

Figure 6-1 shows the RAED architecture with two domains, one at the University of Strathclyde (SU) and one at Glasgow Caledonian University (GCU). Both SU and GCU domains are running individual instances of web servers, OASIS servers and databases. A client wishing to access the SU system must present a valid password-protected X.509 certificate in order to set up a secure TLS-based HTTPS session between the client and the server. When a client attempts to access a secured resource, the request is first passed onto the OASIS server, which checks the local database to determine whether the client has the appropriate privileges.

The OASIS server then checks an XML policy file and performs environmental lookups. If the client request is authorized, the OASIS server informs a server-side script in the web server, which will dynamically generate the appropriate web pages for the client. When a client connected to GCU wishes to access data governed by the SU domain, the request propagates from the GCU OASIS server to the SU OASIS server. Thus, a shared policy must be negotiated between the two institutions. The SU OASIS server checks the policy file plus any environmental constraints and decides whether to allow the access to the local data.

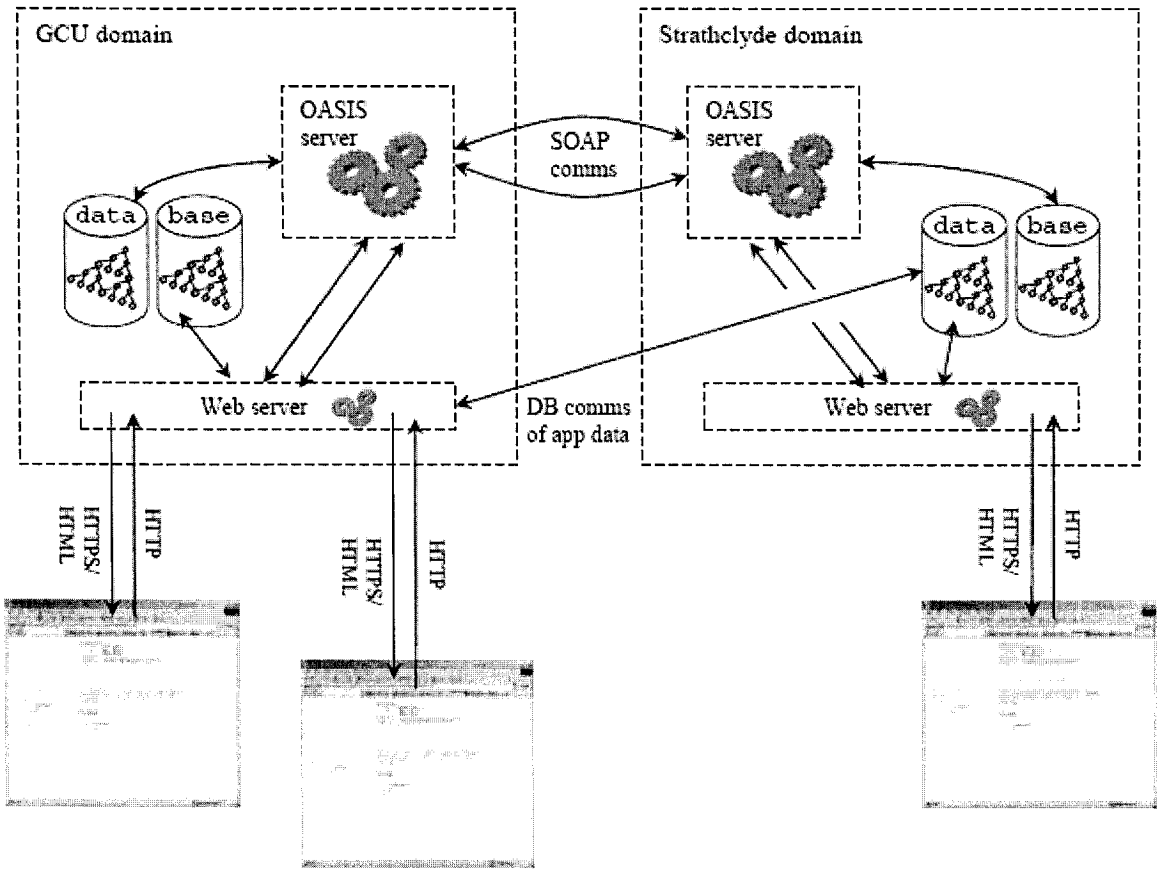

Figure 6-1: The RAED system architecture [51]

The RAED system confirms that separating system and application administration simplifies the overall task and minimizes the risk of errors, particularly when distributed sites are cooperating. The tasks for a system administrative staff include registering individuals and recording in a database, or certifying, their employment or group memberships. The tasks for the application administrative staff include the expression and enforcement of role activation and authorization policies, including service-level agreements between distributed institutions [58]. For example, if GCU students are accessing material at SU, it is sufficient for the SU system to accept a GCU certified student certificate as a credential for activating a student-on-course role. One institution need never be involved with details of the individuals involved with the other. The fact that OASIS defines service-specific roles that are activated within sessions, as opposed to generic, persistent roles which are used for a wide variety of purposes, allows access control to be defined precisely, according to the principle of least privilege [51].

6.1.2 LCDM and LPEE

Learning Process Definition Model (LCDM) and a Learning Process Execution Engine (LPEE) [52] specify and execute learning process models, which represent instructional modules in the forms of *activity trees*. LCDM is a model implementation for distributed sharable learning service components (s*oftware objects)* and resources (*content objects*). This model is implemented with Web services and Event-Trigger-Rule (ETR) technology for achieving Web services-based, dynamic and collaborative e-learning; and for making learning process models active, adaptable, flexible, and customizable. Through the integration of event, event filtering, event notification, condition-action rule processing technologies with computer mediated communication (CMC) tools, the interaction, coordination, and communication among people and software systems that are involved in distributed e-learning is enhanced.

To store and manage different types of rules, an ETR Server is integrated with the LPEE to make learning process models active, flexible, customizable, and adaptable. It is also used to facilitate the interaction and coordination among learners, administrators, authors, and other personnel involved in collaborative e-learning [52]. Those rules are triggered for processing when events are posted during the execution of an instance of an activity tree. This approach allows updating of any types of rules and adding new rules, without requiring changes to the system itself. Event-injected activities, condition action rules and triggers, which link events to rules, can be separately specified by different people or organizations and may be managed in a distributed fashion. This separation is important for achieving the dynamic properties at run time to be elaborated upon next.

By allowing users involved in e-learning to define and register events in a Web-service registry, those users can also browse or programmatically search the registered events and subscribe them. Through this way, users can define rules and triggers and store them at their corresponding sites with security and privacy access control. Those rules and triggers will be processed by the replicas of the Event-Trigger-Rule Server that are installed at these sites. The processing of distributed triggers and rules is activated by the event notification mechanism. Event, event filtering, and event notification can be powerful means to tie

loosely coupled distributed systems and users together. By integrating them with the rule specification and processing mechanisms, learning process modeling and execution mechanisms, and CMC tools, they can together form a powerful information infrastructure for achieving advanced distributed learning.

6.1.3 Comparison

We can make a table to compare my thesis work with RAED and LCDM model in terms of

- variety of functional services (VFS) vs. courseware delivery (CD),
- easier in integration of services corresponding to the role of users,
- easier in management of user identity,
- access to local and remote services,
- security of information transformation,
- role-based access control portal framework, and
- action-driven policy-based access control.

Table 6-1 Comparison with related work

Feature	RAED	LDCM	My work
Variety of functional services (VFS) vs. courseware delivery (CD)	CD	VFS	VFS
easier in integration of services based on the role of users	N/A	N/A	Yes
easier in management of user identity	N/A	N/A	Yes
access to local and remote services	Yes	N/A	Yes
Message-level agreement	Yes	N/A	Trust assumed
role-based access control portal framework	N/A	N/A	Yes
action-driven policy-based access control	After the user logon, assigned to session.	Action trigger	Action trigger.

6.2 Summary

This thesis describes a successful implemented role-based access control portal framework with action-driven policy-based access control component, which matches the initial requirements for Web-services oriented e-learning system, Ubilearn.

Based on the experience with the implementation of access control approach and mechanism for Web services-oriented e-learning portal system, the following conclusions can be drawn with respecting to the criteria and requirements of access control as discussed in Chapter 3.

Single sign on

It is practical to use a portal as a single access point, which allows a user to access those services within the portal page with one-time login. Each virtual portal page can incorporate many portlet applications that conduct as user interfaces for service components across the distributed environments. All virtual portal rendering is controlled by a component called portal controller, where, when a user has provided his/her identity, the login module will direct the user's browser to the corresponding portal on the basis of the role assigned to the user.

Easy of deployment and integration

When a new service component is ready, it is easy to integrate it into the role-based access control portal framework by importing the component package to the portal application and adding a new portlet user interface to the corresponding virtual portal. By combining with WSRP, portlets residing on remote servers and working as service providers can be incorporated into the local virtual portal and consumed by the local portlet, which increases the reusability of those remote service components.

Simplification of identity management

The mechanism of associating the role of users to the virtual portal in a portal controller simplifies user identity management and access control to each of the services that the user is entitled to. In this process, the management of access rights to the services only needs to be

conducted via assigning a role to a user, if the user's identity is in effect, or remove the role from the user if the user's identity is expired.

Separation of environment parameters from functional component

The action-driven policy-based access control provides a mechanism that makes a separation of controlling access to those services based on the environment parameters from the functional services themselves. This improves not only the feasibility and maintainability to development and implementation of a service component - where we do not care under which situation the service component will be accessed; but also the accessibility management - where the environment parameters can be modified without interfering the service component.

6.3 Future work

This thesis presents our design and implementation for an application-level access control mechanism that combines an action-driven policy-based approach with a role-based portal access control model for Ubilearn, a Web services-oriented e-learning portal. These distributed system security issues include access control, identity management, authentication, password management, authorization, encryption of confidential information on each node, and integrity of information passing between nodes.

Although, Web services are best applied at facilitating the inter-connection between providers and consumers of new custom application functions, the fact that client applications access those Web services-based components through their exposed interfaces demonstrates a risk from unauthorized user accesses since the presented access control mechanism is based on the assumption of the trust relationships between Web services-oriented functional components having been constructed.

WSRP allows user interface to be transmitted over a web services protocol and enhances the reusability. However, this mechanism is increasing risks where those WSRP producers allow their portlet interfaces to be consumed by the WSRP consumers without creating proper trust relationships. Both WSRP producers and consumers are thin user interfaces, which are lack

the ability to create reliable trust management contracts between them, nor provision for passing along -- such as a user token from an SSO system or other means of authenticating a user to the producer.

To solve the issue, we have to consider in creating trust relationship among the Web services-oriented applications and business logic tiers. Since some of the Web services could be "on the top" of other Web services, thus, creating trust relationships can be done among the Web services components. To secure creation of trust relationships, message-level security on encryption of confidential information is a big issue for Web services. Though the Web-based security technologies -- basic authentication and SSL -- have been incredibly effective for all sorts of online transactions works well for Web services components; and those technologies focus on securing the end-to-end transportation rather than just the SOAP message transmitted between two intermediary points-two Web services components.

In addition, more security issues for Web services should be considered, as such, whether the Web services are talking to the correct endpoints, whether the communications are kept confidential between all the distributed Web services components, whether the message integrity is maintained at all times, and whether a Web services component to be invoked is clearly identified, known, trusted, and authorized by all other authorized services [48].

Therefore, the future work for our Ubilearn system is to create message-level security services components as an intermediary to solve all issues while invoking Web services components within Web services-based portal application.

Bibliography

[1] T. Marston, "A role-based access control (RBAC) system for PHP", available at: http://www.tonymarston.net/php-mysql/role-based-access-control.html, last visited on April 11, 2005.

[2] "Standards Body Takes on Security", available at: http://www.automationworld.com/cds_print.html?rec_id=506, last visited on October 15, 2005

[3] Eduardo B. Fernandez and John C. Sinibaldi, "More patterns for operating systems access control", available at http://www.polaris.cse.fau.edu/~security/public/OSAccessControl.pdf, last visited on October 14, 2005.

[4] "E-learning community", available at http://www.lansbridge.com/elearning/overview.php, last visited on April 12, 2005.

[5] H. Harney, Andrea Colgrove, and Patrick McDaniel, "Principles of Policy in Secure Groups", available at http://www.isoc.org/isoc/conferences/ndss/01/2001/papers/mcdaniel.pdf

[6] R Atkinson, "Security Architecture for the Internet Protocol", RFC1825, Internet Engineering Task Force, August 1995.

[7] "Personalization", available at http://searchcio.techtarget.com/sDefinition/0,,sid19_gci532341,00.html, last visited on October 15, 2005.

[8] M. Stevens, "Service-Oriented Architecture Introduction", available at: http://www.developer.com/services/article.php/10928_1010451_2, last visited on October 15, 2005.

[9] Shi Kuo Chang, Polese, G., Cibelli, M., "Visual Authorization Modeling in E-commerce Applications", Multimedia, IEEE, Volume 10, Issue 1, Jan.-March 2003 Page(s):44 – 54

[10] B. J. Brockbank, "Next-Generation Management Systems", available at: http://www.workindex.com/editorial/train/trn0308-s-01.asp, last visited on October 15, 2005.

[11] R.K. Thomas and R.S. Shandu, "Task-Based Authorization Controls (TBAC): A Family of Models for Active and Enterprise-oriented Authorization Management", Proc. IFIP WG11.3 Workshop on Database Security, Chapman & Hall, 1998, pp. 166-181

[12] Xu, Z., Yin, Z., and Saddik, A.E., "A Web services oriented framework for distributed e-learning", Electrical and Computer Engineering, 2003. IEEE CCECE 2003. Canadian Conference Volume: 2, On page(s): 943- 946 vol.2, May 2003.

[13] "Web Services for Remote Portlets Specification", available at: http://www-106.ibm.com/developerworks/webservices/library/ws-wsrp/, last visited on October 15, 2005.

[14] Allison, C., Bain, A., Ling, B. and Nicoll, R., MMS: "A User Centric Portal for eLearning", in 14th int. workshop on database and expert systems applications, (Prague, Czech Republic, 2003), IEEE CS, 292-297

[15] Web Services for Interactive Applications (WSIA), available at http://www.service-architecture.com/web-services/articles/web_services_for_interactive-applications_wsia.html, last visited on October 15, 2005.

[16] R.S. Sandhu and P. Samarati, "Access control: principle and practice". Communications Magazine, IEEE, 32:40–48, September 1994.

[17] R. Sandhu, E. Coyne, H. Feinstein, and C. Youman, "Role-Based Access Control Models", IEEE Computer, 29(2):38–47, 1996.

[18] "HIPAA Security: You Can Run, But You Can't Hide", available at http://www.texmed.org/cme/pms/ec_pmsem/hipaa/tss.asp, last visited on October 15, 2005.

[19] Pekka Nikander, "An Architecture for Authorization and Delegation in Distributed Object-Oriented Agent Systems", Helsinki University of Technology, Department of Computer Science, Telecommunications Software and Multimedia Laboratory, FI-02015 TKK, Espoo, Finland

[20] C. Steindl, "WebSphere Portal: Portlet Concepts and Guidelines", Consulting IT Architect and Method Exponent, IBM Global Services

[21] "What is portal software", available at http://searchcio.techtarget.com/sDefinition/0,,sid19_gci525918,00.html, last visited on October 15, 2005.

[22] S. Allamaraju, "Professional Java Server Programming J2EE Edition", ISBN: 1861004656, Wrox Press, Inc. 2000.

[23] "Message-Oriented Middleware (MOM)", Software Engineering Institute at Carnegie Mellon University: http://www.sei.cmu.edu/str/descriptions/momt.html, last visited on October 15, 2005.

[24] Object Management Group FAQ (OMG), available at http://www.omg.org/gettingstarted/corbafaq.htm, last visited on April 4, 2005.

[25] Web services, available at http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci750567,00.html, last visited on April 4, 2005.

[26] Robert J. Brunner, Frank Cohen, Francisco Curbera, Darren Govoni, Steven Haines, Matthias Kloppmann, Benoit Marchal, K. Scott Morrison, Arthur Ryman, Joseph Weber, Mark Wutka, "Java Web Services Unleashed", ISBN: 0-672-32363-X, Sams Publishing, 2002.

[27] S. Allamaraju, "Inside WSRP", available at http://dev2dev.bea.com/pub/a/2005/03/inside_wsrp.html, last visited on October 15, 2005.

[28] "Web Services Architecture", available at: http://www.w3.org/TR/2002/WD-ws-arch-20021114/ last visited on April 4, 2005.

[29] E. Friedman-Hill, JESS in Action. Rule-based Systems in Java, Manning Publications, Greenwich (USA), 2003

[30] "WordNet Search - 2.1 beta", available at http://www.cogsci.princeton.edu/cgi-bin/webwn2.1, last visited on June 15, 2005.

[31] "An introduction on role-based access control", available at http://csrc.nist.gov/rbac/NIST-ITL-RBAC-bulletin.html, last visited on June 17, 2005

[32] "A Guide to Building Secure Web Applications", available at http://www.cgisecurity.com/owasp/html/ch08.html, last visited on June 20, 2005.

[33] R.J. Hulsebosch, "Context sensitive access control", *SACMAT'05,* June 1–3, 2005, Stockholm, Sweden.

[34] Mark Evered and Serge Bögeholz, "A Case Study in Access Control Requirements for a Health Information System", Australasian Information Security, Workshop 2004 (AISW 2004), Dunedin, New Zealand.

[35] Rajul Rana and Sai Kumar, "Service on Demand Portals", available at http://weblogic.sys-con.com/read/46317.htm, last visited June 23, 2005

[36] V. Ungureanu, "A Policy-Based Access Control Mechanism for the Corporate Web", 16th Annual Computer Security Applications Conference (ACSAC'00).

[37] E.C. Lupu, "A policy based role framework for access control", 1st ACM/NIST Role Based Access Control Workshop, Gaithersburg, USA, Dec. 1995

[38] "Identity Management Terminology", http://idsynch.com/docs/identity-management-terminology.html, last visited on June 25, 2005

[39] Camelot access control whitepaper, "Differentiating between access control terms", available at http://www.secinf.net/authentication_and_encryption/Differentiating_Between_Access_Control_Terms.html, last visited on June 25, 2005.

[40] "Web Services Architecture", available at http://www.w3.org/TR/ws-arch/, last visited on June 25, 2005.

[41] "Identity management in portals", available at http://devresource.hp.com/drc/resources/id_mgt_si/index.jsp, last visited on June 26, 2005.

[42] JMS and J2EE, available at http://dev2dev.bea.com/pub/a/2002/06/336.html?page=last, last visited on July 3, 2005.

[43] The IETF Policy Framework Working Group: Charter, available at the URL: http://www.ietf.org/html.charters/OLD/policy-charter.html, last visited on July 4, 2005.

[44] Simon John, "A Dynamic e-Business Application Using Web Services", available at http://www.developer.com/tech/article.php/10923_2211381_3, last visited on July 1, 2005.

[45] RFC 3060: Policy Core Information Model -- Version 1 Specification, available at http://www.faqs.org/rfcs/rfc3060.html, last visited on July 4, 2005

[46] Sloman, M., Lupu, E. "Security and Management Policy Specification", IEEE Network, Vol. 16, no. 2, 2002, p 10-19.

[47] Kai Wang, Jianming Ke and Abdulmotaleb El Saddik, "Architecture for Personalized Collaborative E-learning Environment" , In Proceedings of Ed-Media 2005, Montreal, Quebec, June 2005.

[48] Jothy Rosenberg and David Remy, "Securing Web Services with WS-Security: Demystifying WS-Security, WS-Policy, SAML, XML Signature, and XML Encryption" Book published by Sams.

[49] J. Bacon, K. Moody, and W. Yao, "A model of OASIS role-based access control and its support for active security", ACM Transactions on Information and System Security, 5(4):492–540, November 2002.

[50] R. Hayton, J. Bacon, and K. Moody, "OASIS: Access Control in an Open, Distributed Environment". In Proceedings IEEE Symposium on Security and Privacy, pages 3–14, Oakland CA, May 1998.

[51] Steve Neely, Helen Lowe, David Eyers, Jean Bacon, Julian Newman, Xiaofeng Gong, "Engineering e-learning systems (ELS): An architecture for supporting vicarious learning in a distributed environment", Proceedings of the 2004 ACM symposium on Applied computing, March 2004

[52] Stanley Y. W. Su Gilliean Lee, "Web-service-based, Dynamic and Collaborative E-learning", Proceedings of the PGL DB Research Conference, PGLDB'2003, pp. 68-77, 2003.

[53] Matteo Baldoni, Cristina Baroglio, Viviana Patti, and Laura Torasso, "Reasoning about learning object metadata for adapting SCORM courseware" Societ a Reale Mutua di Assicurazioni | Servizio Informatico-S.I.Ge.A. Via Corte d'Appello 11, Torino (Italy)

[54] Khan, B. H. (2005), "Managing e-learning: Design, delivery, implementation, and evaluation", Hershey, PA: Information Science Publishing. (Website: http://BooksToRead.com/elearning)

[55] Osvaldo A. Santosa, Fernando M.S. Ramosb, "Proposal of a framework for Internet based licensing of learning objects", aEscola Superior de Tecnologia, Instituto Polite´cnico de Castelo Branco, 6000-767 Castelo Branco, Portugal Universidade de Aveiro, Departamento de Comunicac¸ Arte/UnICA, 3810-193 Aveiro, Portugal

[56] R.M. Carro, E. Pulido, and P. Rodruez, "Dynamic generation of adaptive internet based Courses", Journal of Network and Computer Applications, 22:249--257, 1999.

[57] T. Mayes and F. Dineen, "Developing Tertiary Courseware through capturing Task Directed Discussions", In Proceedings of ED-MEDIA, Seattle, USA, 1999.

[58] X. Gong and J. Newman, "Selecting a Security Architecture for a New Model of Distributed Tutorial Support", In Proceedings of 11th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, pages 89–94, Los Alamitos, CA, June 2002, Computer Society Press.

[59] J. Bacon, K. Moody, D. Chadwick, and O. Otenko "Persistent versus Dynamic Role Membership", In IFIP WG 11.3 Working Conference on Database and Applications Security, Estes Park, Colorado, August 2003.

[60] G. J. Ahn and R. Sandhu, "Role Based Authorization Constraints Specification", ACM Transaction on Information and System Security, 3(4), November 2000.

[61] J. F. Barkley, "Implementing Role Based Access Control using Object Technology", In First ACM Workshop on Role Based Access Control, November 1995.

[62] Eduardo B. Fernandez, "Security patterns and secure systems design using UML", available at http://www.iceis.org/iceis2005/tutorials.htm, last visited on October 14, 2005.