

THE FLORIDA AGRICULTURAL AND MECHANICAL UNIVERSITY

COLLEGE OF ARTS AND SCIENCES

**A Role-based Access Control Security Model for Workflow
Management System in an E-healthcare Enterprise**

By

Lang Zhao

A Thesis submitted to the
Department of Computer and Information Sciences
in partial fulfillment of the
requirements for the degree of
Masters of Software Engineering Science

Fall Semester, 2008

The members of the Committee approve the thesis entitled Role-based Access Control Security Model for Workflow Management System in an E-healthcare Enterprise by Lang Zhao, defended on November 04, 2008.

Dr. Hongmei Chi
Professor Directing Thesis

Dr. Edward L. Jones
Committee Member

Dr. Ken R. Riggs
Committee Member

Dr. Simon Y. Foo
Outside Committee Member

Approved:

Edward L. Jones, Ph.D., Chair, Department of Computer and Information Sciences

Ralph W. Turner, Ph.D., Dean, College of Arts and Sciences

Chanta M. Haywood, Ph.D., Dean, School of Graduate of Studies and Research

To my husband Xiaojun and my daughter Lisa

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to my advisor, Dr. Honemei Chi, for her firm support and guidance in directing my thesis. This work would not have been possible without the constant guidance of her. She taught me to think critically, she encouraged my ideas, and most importantly, she was there whenever I needed help.

I would also like to thank all my committee members: Dr. Ken R. Riggs, Dr. Edward L. Jones and Dr. Simon Y. Foo for their support, advice and encouragement in the completion of this Masters degree. Special thank goes to Dr. Edward L. Jones, Dr. Ken R. Riggs and Mr. Chris Nowicki for their funding support. Without the fund they provided for my study, none of these can be done.

I want to thank all my friends, especially Jian Xu, Zhenyu Lu, Yan Li, Li Fan and Jermaine O'Neil Stewart. Some of you are not named in this document, but my sincere appreciation and love are expressed to you for your unflinching support and encourage.

None of this work would have been possible were it not for the support of my family. I want to thank my parents for their love, encouragement, patience, selflessness and sacrifice.

There is another person for whom I owe everything else. Thanks for bringing me here and your constant encouragement, support and love. Thank you, Xiaojun.

TABLE OF CONTENTS

LIST OF FIGURES	viii
LIST OF TABLES	xi
LIST OF ABBREVIATIONS	xii
ABSTRACT	xiv
Chapter 1 Introduction	1
1.1 Workflow Management Systems (WFMS)	2
1.2 A Case Study for E-healthcare	3
1.3 Security Models	6
1.4 Contributions	9
1.5 Thesis Organization	10
Chapter 2 Security Models for WFMS	11
2.1 Information Assurance	11
2.2 Access Control Models	13
2.2.1 <i>Chinese Wall Model</i>	13
2.2.2 <i>Role-based Access Control Model</i>	15
2.2.3 <i>Task-based Authorization Control Model</i>	16
2.2.4 <i>Context based Security Model</i>	17
2.3 Collaborative environment and E-healthcare	19
2.4 Smart Environment and Future E-healthcare	20
Chapter 3 Security Rules for E-HealthCare	22

3.1	The Security Rule	23
3.1.1	<i>Administrative Safeguards</i>	24
3.1.2	<i>Physical Safeguards</i>	24
3.1.3	<i>Technical Safeguards</i>	24
3.2	The implementation of Security Rules in RBAC security model	27
Chapter 4	Web Services for E-Healthcare	29
4.1	Overview of Web Services	29
4.2	Web Services Security	32
4.3	E-Healthcare and Web Security	33
Chapter 5	RBAC model and Its Implementations	35
5.1	RBAC Model	35
5.2	eXtensible Access Control Markup Language (XACML)	37
5.2.1	<i>XACML Architecture</i>	37
5.2.2	<i>XACML Policy Language</i>	39
5.2.2.1	Policy language components	39
5.2.2.2	An example of XACML Policy	41
5.2.3	<i>RBAC Profile of XACML</i>	43
5.2.4	<i>Current State of the Art: choosing an XACML Implementation</i>	44
Chapter 6	A Case Study of E-healthcare System	46
6.1	Three-Tier Architecture	47
6.2	Database Design of an e-healthcare prototype	48
6.3	Roles and their Applications	50

6.3.1	<i>Administrator Role and its windows applications</i>	54
6.3.1.1	Functions for Administrator Role	54
6.3.1.2	Data Access for the Administrator Role	58
6.3.1.3	GUIs for the Administrator Role	59
6.3.2	<i>The Physician Role and its windows Application</i>	62
6.3.2.1	Functions for Physician Role	62
6.3.2.2	Data Access for Physician Role	64
6.3.2.3	GUI for the Physician Role	65
6.3.3	<i>The Clinic Staff Role and its windows Applications</i>	67
6.3.3.1	Functions for Clinic Staff Role	68
6.3.3.2	GUIs for Clinic Staff Role	70
6.3.3.3	Data access for Clinic Staff Role	70
6.3.4	<i>The Patient Role and its Web Application</i>	71
6.3.4.1	Functions for Patient Role	71
6.3.4.2	GUIs for Patient Role	72
6.3.4.3	Data Access for Patient Role	73
6.4	Authentication ---- Access Control Mechanism	73
6.5	Summary	74
Chapter 7	Conclusions and Future Work	76
REFERENCES		78

LIST OF FIGURES

Figure 1. Current workflow of Leon county uninsured e-healthcare program in Tallahassee, FL	5
Figure 2. Ideal workflow of Leon county uninsured e-healthcare program in Tallahassee, FL	6
Figure 3. Role-based Access Control	16
Figure 4. Context-based Security Policy	19
Figure 5. Document sharing in the ideal workflow of Leon county uninsured e-healthcare program in Tallahassee, FL.....	20
Figure 6. Health Smart Home	21
Figure 7. Role-based Access Control Model	36
Figure 8. Role Hierarchy RBAC.....	37
Figure 9. XACML Architecture.....	38
Figure 10. XACML Policy Language Model [34].....	40
Figure 11. Permission PolicySet	43
Figure 12. Role PolicySet [37].....	44
Figure 13. A typical three-Tier Architecture	48
Figure 14. RBAC Database.....	50
Figure 15. Data Flow Diagram for Log-in Function.....	52
Figure 16. Data Flow Diagram for Change Password and Set Secure Questions	53
Figure 17. Use Case Diagram for Administrator Role	54

Figure 18. Data Flow Diagram for Users Management.....	55
Figure 19. Data Flow Diagram for Roles Management.....	56
Figure 20. Data Flow Diagram for Delegation Management	56
Figure 21. Data Flow Diagram for Assign Patient to Physician.....	57
Figure 22. Source Code for RetrieveUsers in User Management.....	58
Figure 23. GUI for User Management in Control Panel.....	59
Figure 24. GUI for Role Management in Control Panel.....	60
Figure 25. GUI for Delegation Management in Control Panel.....	61
Figure 26. GUI for Assign Patient to Physician in Control Panel	61
Figure 27. Use Case Diagram for Physician Role	62
Figure 28. Data Flow Diagram for Medical Record Form	63
Figure 29. Source code for Retrieve a Patient Record.....	64
Figure 30. GUI for a Physician	65
Figure 31. Medical Records for a Patient	66
Figure 32. Physician Website	66
Figure 33. Use Case Diagram for Clinic Staff Role	67
Figure 34. Data Flow Diagram for View Medical Record	68
Figure 35. Data Flow Diagram for Discharge and View Inactive Patients	69
Figure 36. GUI for Clinic Office	70
Figure 37. Source code for retrieving patient list of a clinic	71
Figure 38. Data Flow Diagram for Patient Website	72
Figure 39. Patient Website.....	72

Figure 40. Source code for retrieving personal information and medical records..... 73

LIST OF TABLES

Table 1. Summary of Basic Concepts in Workflow	2
Table 2. Notations for Workflow by using YAWL [8]	3
Table 3. Abbreviation of organizations in case study	4
Table 4. Notations for Document Sharing in Case Study	19
Table 5. Specifications of Administrative Safeguards.....	25
Table 6. Specifications of Physical Safeguards	26
Table 7. Specifications of Technical Safeguards	26
Table 8. Characteristics of Web Services	30
Table 9. Layers of the Web Service Stack.....	31
Table 10. XACML Policy Example.....	42
Table 11. Notations for Data Flow Diagram	51

LIST OF ABBREVIATIONS

ANSI	American National Standards Institute
B2B	Business-to-Business
EPHI	Electronic Protected Health Information
HIPAA	Health Insurance Portability and Accountability Act
HSH	Health Smart Home
IA	Information Assurance
OASIS	Organization for the Advancement of Structured Information Standards
PAP	Policy Administration Point
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PHI	Protected Health Information
PIP	Policy Information Point
PPS	Permission <PolicySet>
RPS	Role <PolicySet>
RBAC XACML Profile	RBAC profile of the XACML 2.0 standard
RBAC	Role-based Access Control
SOAP	Simple Object Access Protocol
TBAC	Task-based Access Control
UDDI	Universal Description, Discovery, and Integration

WFMS	Workflow Management Systems
WSDL	Web Services Description Language
WSFL	Web Services Flow Language
XACML	eXtensible Access Control Markup Language
XML	eXtensible Markup Language
YAWL	Yet Another Workflow Language

ABSTRACT

The inability to share information across systems is just one of the major impediments in the health care business that hinders progress towards efficiency and cost-effectiveness. Workflow management systems are very popular and largely being used in a business environment for e-commercials. Workflow management systems are the fundamental support for the interoperability of multiple organizations, such as e-healthcare systems. In this thesis, we study secure inter-organizational workflow management models in e-healthcare. To secure the data in e-healthcare, the workflow management systems have to manage and execute the processes in a secure model. This thesis will discuss the security policy, security access control and the security mechanism to integrate a role-based access control model that is suitable to an e-healthcare system. Role-based access control security specifications are used to control the access to critical resources and avoid security violations. In addition, Microsoft Visual Studio and C# programming language are used to implement the secure workflow model for a real case study.

Keyword: workflow management system, e-healthcare, inter-organization, security models, Web Service, role based access control, security policy

Chapter 1 Introduction

Healthcare is increasingly a cooperative business that involves many individuals and organizations [3]. With the rapid development of information technologies, it is increasingly convenient and efficient to provide good healthcare information services. E-healthcare is a term for healthcare practice that is supported by electronic processes and communication it and provides a way for medical informatics, public health and business to be delivered via the Internet [5].

In e-healthcare, personal medical information is being entered, processed, stored and transmitted electronically and the private health information should be protected in a secure way. In 1996, the Health Insurance Portability and Accountability Act (HIPAA) was enacted as Public Law 104-191, to specify “the privacy, security and electronic transaction standards with regard to patient information for all health care providers” [1]. The introduction of the Privacy and Security Rules of the HIPAA of 1996 is just one such step that the US Federal Government has taken to protect the privacy of patients in the health care industry. HIPAA helps to establish national standards for electronic health care transactions and national identifiers for providers, health plans, and employers.

Due to the popularity of e-healthcare, most healthcare organizations have already been employing some kind of workflow management techniques to help improve the efficiency of their work processes [2]. It is not enough that the healthcare organization has the e-service in-house. Only when organizations can collaborate, as over the Internet, do we achieve real e-healthcare.

1.1 Workflow Management Systems (WFMS)

Workflow [7] at its simplest is the movement of documents and/or tasks through a work process. Workflow problems can be modeled and analyzed using graph-based formalisms such as Petri nets. A workflow system contains two basic elements: modeling and execution components. Table1 gives the basic concepts of workflow derived from the Workflow Management Coalition [7]:

Table 1. Summary of Basic Concepts in Workflow

Activity	It is a discrete process step performed either by a machine or human agent and may consist of one or more tasks [7].
Business process	It consists of a sequence of activities and has distinct inputs and outputs and serves a meaningful purpose within an organization or between organizations [7].
Workflow	It is the automation of a business process in whole or in part, during which documents, information or tasks are passed from one participant to another according to a set of procedural rules [7].
Workflow Management	It is the progress of automated business procedures performed by a company, industry, department or person [4].

Yet Another Workflow Language (YAWL) [9] is an open source and very powerful and new workflow language. Its structure is inspired by an advanced concurrency control language, Petri Nets, which are in turn backed up by 40 years of research [8]. YAWL aims to provide a support environment to specify, analyze and execute business processes, and to simplify the specification of executable business processes without the distraction of "unnecessary" technical considerations [9]. Table 2 gives some of the fundamental notions in YAWL.

Table 2. Notations for Workflow by using YAWL [8]

 Atomic task	 AND-split task	 XOR-split task
 Input condition	 AND-join task	 XOR-join task
 Output condition	 OR-split task	 Cancellation
 Condition	 OR-join task	 Composite task

1.2 A Case Study for E-healthcare

In this thesis, we will provide a case study to address the security issues of workflow management systems for an e-healthcare enterprise. We consider a real but conventional case. In Leon County, the health department has two clinic centers: Bond Community Clinic Center and Lincoln Neighborhood Clinic Center. These two clinic centers provide healthcare and discount prescription medicine to non-insurance people who are also not qualified for Medicare. In the weekend or at night, these people can go to Tallahassee Memorial Hospital Emergency Room or to Capital Regional Medical Center Emergency Room. Later, they have to go to Lincoln Neighborhood clinic center to take continue health care and/or pick up the discount prescription drugs. There are two departments in each clinic center: clinic department and pharmacy department. Now the data is manually passed from clinic department to pharmacy department. Because of the security issues, these two pharmacies are independent in each other and cannot connect or share the data directly. In this scenario, some patients can get the discount prescription drugs from both pharmacies. All the drugs are bought by funding from government and the patients only pay a little for the drugs. It is a big problem for these two pharmacies

since they are limited in funding. Through meetings and discussions, we determined the current workflows for this system of two pharmacies, two clinics, the Emergence Room's patients, Department of Health (DOH) and related organizations in Leon County; FL. Table 3 lists the participants with abbreviations.

Table 3. Abbreviation of organizations in case study

Bond Commu Clinic	Bond Community Center Clinic
FAMU Phar2	FAMU Pharmacy2 in Bond Community Center Clinic
LNHS Clinic	Lincoln Neighborhood Service Center Clinic
FAMU Phar1	FAMU Pharmacy1 in Lincoln Neighborhood Service Center
TMH ER Pat	Tallahassee Memorial Hospital Emergency Room Patient
CRMC ER Pat	Capital Regional Medical Center Emergency Room Patient
RX30	Pharmaceutical operation system

Figure 1 shows the current workflow of Leon county uninsured e-healthcare program in Tallahassee, FL. The patient is referred to one of the clinics (the LNHS clinic or the Bond community Center Clinic) according to a division of zip codes. The patient who goes to LNHS Clinic picks up the discount prescription drugs in FAMU Pharmacy1, while the patient from the Bond Community Center Clinic goes to FAMU Pharmacy2. The related patient's information and prescriptions are stored on the FAMU RX30 that connects with Stored DOH Server or LNHS RX30 that connects with Donated DOH Server. The TMH ER patient and the CRMC ER patient may only go to FAMU Pharmacy1 to pick up their discount prescription drugs. The data on patients go ahead as the arrowhead directions are showed in figure. According to the HIPAA policies, patient records at each pharmacy are now stored on a DOH server that is one of the most secure servers in Florida.

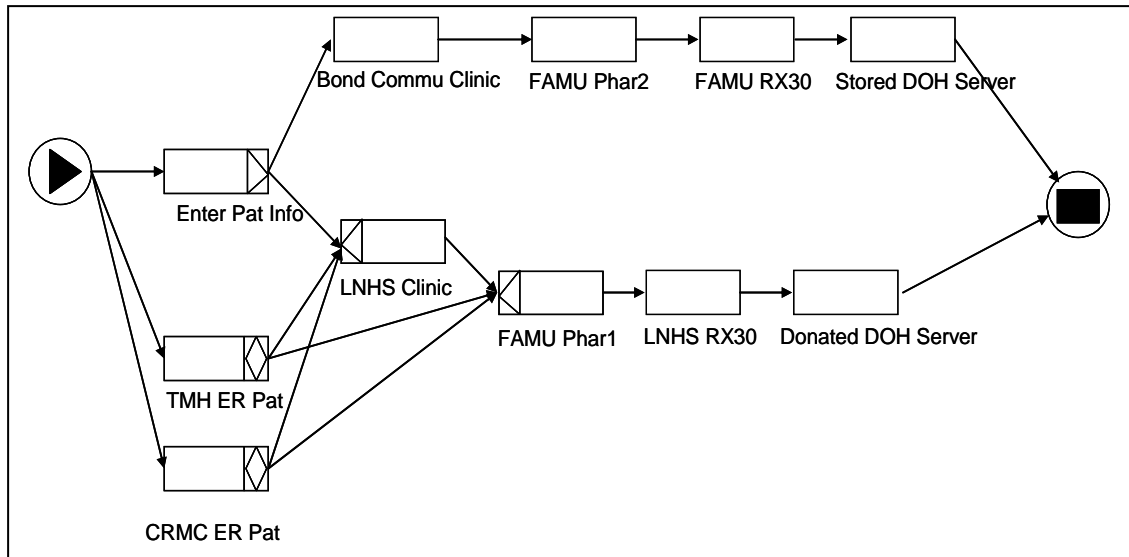


Figure 1. Current workflow of Leon county uninsured e-healthcare program in Tallahassee, FL

There are two problems in the case study:

1. No automatically electronic transactions among departments in an organization.
2. No systematic transactions among organizations.

Figure 2 below shows the Ideal workflow. The two clinic centers have systematic transactions with Tallahassee Memorial Hospital Emergency Room or Capital Regional Medical Center Emergency Room. In each clinic center, there are automatically electronic transactions between the clinic department and pharmacy department. The systematic transactions among organizations prevent duplicating prescriptions. The pharmacies can save funding wasted duplicate prescriptions and have more funding to support other people. Using the Internet is an economic method which is especially suitable for the Health Department with limited funding. Managers of the Health Department can use the laptop anywhere via the Internet.

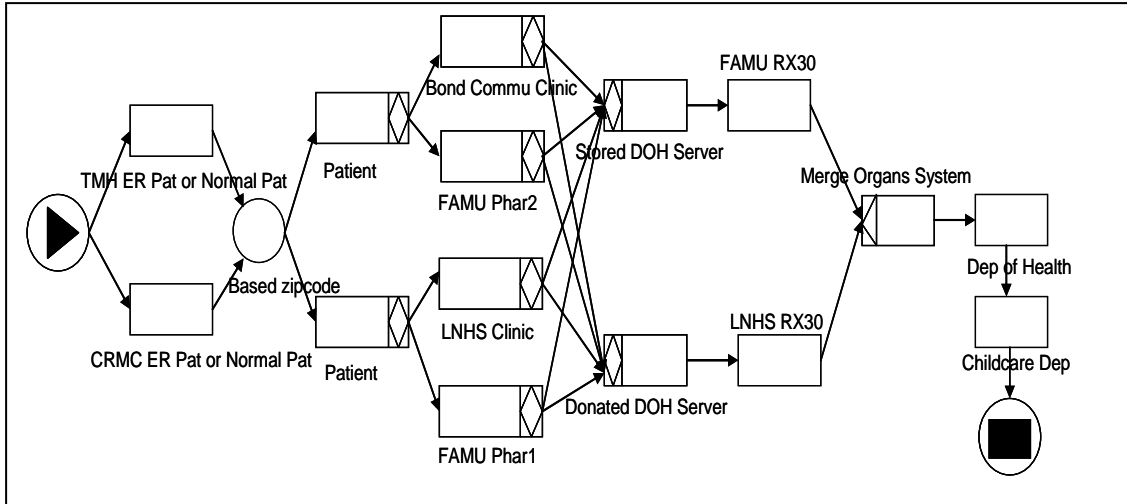


Figure 2. Ideal workflow of Leon county uninsured e-healthcare program in Tallahassee, FL

As the Web service emerges, an inter-organizational workflow management system becomes easier to achieve. In this thesis, we achieve automatically electronic transactions among departments in an organization. Also, we have built a web services which provide a way to visit distributed data servers.

1.3 Security Models

Security issues are critical part of implementing an e-healthcare enterprise. Security requirements approached at the enterprise level initiate the need for models that capture the organizational and distributed aspects of information usage. Access control is an indispensable part of e-healthcare enterprise. A security model provides a formal representation of the access control security policy and its working method. The formalization allows the proof of properties on the security provided by the access control system being designed [6]. Since authorization-to-user is the most important thing and access control decisions are often determined by employee functions in e-healthcare.

Access control of e-healthcare focuses on employee functions and access right. Employee function, in another word, is *role*. There are several fixed roles in e-healthcare, such as administrator, physician, office staff, patient, etc. There are many current access control models such as the Chinese wall model, Role-based Access Control (RBAC), Task-based Access Control (TBAC), Context-based security model, etc. we summarize the state-of-the-art for these access control models.

The Chinese Wall model [12] aims to prevent sensitive information concerning a company being disclosed to competitor companies through the work of financial consultants. The Chinese Wall Model dynamically establishes the access rights of a user based on what the user has already accessed [12]. But the strict enforcement of the access rules result too rigid and restrictive. The Chinese Wall model will be discussed in more detail in next Chapter.

The essence of Role-based access control (RBAC) [14] is that permissions are assigned to roles. This simplifies the security management and helps to determine efficiently which permissions are authorized for what users in a large enterprise system. But the nature of roles is static, so RBAC lacks flexibility and responsiveness. RBAC does not encompass the overall context associated with any collaborative activity [16]; it is a passive security system that serves the function of maintaining permission assignments. RBAC lacks the ability to specify a fine-grained control on individual users in certain roles and on individual object instances [16], so it is not enough for collaborative environments. RBAC will be discussed in more detail in next Chapter.

The TBAC, Task-based authorization control [15], is task-oriented and an active model that allows for dynamic management of permissions as tasks progress to

completion [16]. In TBAC, permissions are associated with contextual information about ongoing activities when evaluating an access request, and authorizations have a strict runtime usage, validity, and expiration characteristics [16]. But there are no contexts in relation to activities, tasks, or workflow progress and it only keeps track of usage and validity of permissions. This is insufficient for collaborative systems that require a much broader definition of context. More fine-grained components need to be defined to support dynamic environments motivated by TBAC [16]. TBAC can be used effectively in an application or enterprise. But for most collaborative environments, TBAC is used within other access control models. TBAC will be discussed in more detail in next Chapter.

The Context-based security model [16] is an active security model. It uses contextual graphs as a modeling tool to provide a convenient way for specifying security requirements in pervasive environments and as a security management tool that eases security administration for complex environments with many heterogeneous services and devices. It also includes contextual information, fine-grained control, groups of users, policy specifications, and policy enforcement characteristics. However, this model is fairly new and requires further testing within the collaborative systems domain [10]. The Context-based security model will be discussed in more detail in next Chapter.

E-healthcare Enterprise is a large-scale system and does have many cooperative organizations instead of competing organizations. So the Chinese Wall model is not suitable for e-healthcare system. In e-healthcare, authorization focuses on user, neither task, nor context. The task-based authorization control and context-based security model are not suitable for e-healthcare system too. Also, access control decisions are often

determined by the roles and control of protect resources is no more based on data ownership in e-healthcare. Users are easy to be divided into groups by employee functions in e-healthcare. Employee function, in another word, is role. Users in the same role do have the same permissions. And the number of roles is largely less than the number of users. Assigning permissions to a role may reduce the complexity of security administration. The Role-based Access Control (RBAC) provides a means of naming and describing many-to-many relationships between individuals and rights. And RBAC regulates the access of users to the information and the system resources based on the roles granted for users. Therefore, RBAC Model is suitable for the e-healthcare enterprise.

1.4 Contributions

The contributions of this thesis are listed below:

- A detailed analysis using workflow technology for the correlations among different organizations of a real-life e-healthcare example at Tallahassee, which includes hospital emergency rooms, clinics, pharmacies, the Department of Health, and the Department of Childcare.
- A selected security model that is suitable for e-healthcare enterprise. The security model is role-based, windows-form-deployed and web-deployed. There are only a few roles, such as patient, physician, pharmacist, manager, in this e-healthcare enterprise. This simplifies the management of authorization. And the web-based services which provide a way to access other servers in distributed locations can be visited via Internet by authorized users.

- A prototype e-healthcare system which includes several windows applications and web application developed in Microsoft Visual Studio 2008 using the C# program language.

1.5 Thesis Organization

In the rest of the thesis, Chapter 2 describes the detailed security models for Workflow Management System. The security rules are researched in Chapter 3. Web services for the e-healthcare and the security of the web services are discussed in Chapter 4. Chapter 5 describes the Role-based security model and its implementation. Chapter 6 presents the implementation of the e-healthcare system. The final chapter 7 presents the conclusions and discusses the future work.

Chapter 2 Security Models for WFMS

The 21st century has brought about a new lifestyle where the networking of all sorts of devices is prevalent. How to protect the sensitive information becomes a big challenge. In this Chapter, firstly we will introduce the Information assurance. Secondly, we will examine the popular existing access control models and computing environment for current and future e-healthcare enterprise. And as part of this examination, we present an overview of the access control models and merits of each approach as well as identify potential shortcomings. Finally we choose an access control model that is suitable for e-healthcare enterprise case study.

2.1 Information Assurance

Information assurance (IA) [40] is the practice of managing information-related risks. More specifically IA protects and defends information and information systems by ensuring their **availability, integrity, authentication, confidentiality, and non-repudiation**. These measures include providing for restoration of information systems by incorporating protection, detection, and reaction capabilities. In e-healthcare, personal medical information is being entered, processed, stored and transmitted electronically and the private health information should be protected in a secure way. Protection of information systems against unauthorized access to or modification of information, whether in storage, processing or transit, and against the denial of service to authorized users, including those measures necessary to detect, document, and counter such threats.

Practically, in order to make information secure, most organizations have a risk assessment periodically. A risk assessment is carried out by a team of people who have knowledge of specific areas of the business. The ISO/IEC 27002:2005 [41] Code for information security management recommends the following be examined during a risk assessment:

- Security policy,
- Risk Management
- Organization of information security,
- Asset management,
- Human Resources security,
- Physical and environmental security,
- Communications and operations management,
- Access control,
- Information systems acquisition,
- Development and maintenance,
- Information security incident management.

It is easy to see that access control is one of part of those procedures. Access control alone is hardly to carry out all tasks of IA. The foundations on access control mechanisms are built start with identification and authentication.

- **Identification** [40] is an assertion of who someone is or what something is.
- **Authentication** [40] is the act of verifying a claim of identity.

Along with other IA procedure, access control is the process of ensuring that authorized users have access to authorized information at the authorized time. Access to protected information must be restricted to people who are authorized to access the information. The computer programs, and in many cases the computers that process the information, must also be authorized.

2.2 Access Control Models

Access control models are the critical part of e-healthcare enterprise. In this section, an overview of various access control models is described. The goal of this section is to find a suitable security model for our e-healthcare enterprise case study.

2.2.1 *Chinese Wall Model*

The Chinese Wall policy [12] was introduced by Brewer and Nash in 1989 [11] as an attempt to balance commercial discretion with mandatory controls [6]. It combines elements of Discretionary Access Control and Mandatory Access Control. The purpose is to prevent sensitive information concerning a company from being disclosed to competitor companies through the work of financial consultants. In the Chinese Wall model, individual items of information are grouped into basic single objects, where each object concerns a single corporation. These objects are then grouped into company datasets, where every object belongs to a single dataset, on behalf of all the information about a single corporation. The company datasets are then classified into conflict of interest (CoI) classes that refer to competing companies. The Chinese Wall policy restricts access according to the following two properties [6]:

Simple security rule: A subject S can be granted access to an object O only if the object O:

- is in the same company datasets as the objects already accessed by S, that is, “within the Wall”, or
- belongs to an entirely different conflict of interest class.

(The term subject can be interpreted as user.)

Access Rules: are divided into Read rules and Write rules [12].

Read Rule: A subject S can read an object O if:

- O is in the same Dataset as an object already accessed by S,

OR

- O belongs to a CoI from which S has not yet accessed any information

Write Rule: A subject S can write an object O if:

- S can read O according to the Read Rule

AND

- No object has been read by S that is in a different company dataset from the one on which the write is performed.

Chinese Wall Model dynamically establishes the access rights of a user based on what the user has already accessed [12]. The simple security rule blocks direct information leakages that can be attempted by a single user, while access rules blocks indirect information leakages that can occur with the collusion of two or more users. But the strict enforcement of the access rules may be too rigid and restrictive. A user that has read objects from more than one dataset is not able to write any object; the user can only read and write objects from a single dataset [12].

2.2.2 *Role-based Access Control Model*

Role-based access control (RBAC) [14] has rapidly emerged in the 1990s as a technology for managing and enforcing security in large-scale systems [13]. The basic opinion of RBAC is that the permissions are organizationally associated with roles, and users are administratively assigned to appropriate roles. RBAC ensures that only authorized users are given access to certain data or resources [13]. This simplifies the management of authorization while providing an opportunity for flexibility in specifying and enforcing enterprise-specific protection policies [14]. In RBAC, a role is a function within the context of an organization with an associated semantics regarding its authority and responsibility [14]. User is defined as a human being, a machine, a process, or an intelligent autonomous agent, etc [14]. Permission is an access mode that can be exercised on objects in the system. Both objects and access modes are domain dependent [14]. System administrators can create roles, grant permissions to those roles, and then assign users to the roles on the basis of their specific job responsibilities and policy. Hence, role-permission relationships can be predefined, making it easy to assign users to the predefined roles [13]. RBAC really helps to determine efficiently which permissions are authorized for what users in a large enterprise system.

Figure 3 shows the mapping relationships among users (A, B, C, D, E, etc), roles (R1, R2, R3, R4, R5, etc.), and permissions (P1, P2, P3, P4, P5, etc) in an RBAC model. Every user has one or more roles. Every role can be assigned to one or many users. They are one-to-one, one-to-many, or many-to-one relationship. Every role has one or many permissions, and permission can be assigned to one or many roles according to the policy. Here we choose the tables (T1, T2, T3, T4, T5, etc) of the database as the

particular data or resources for access. Permission gives the right to do an action (such as read or write) on a table of database.

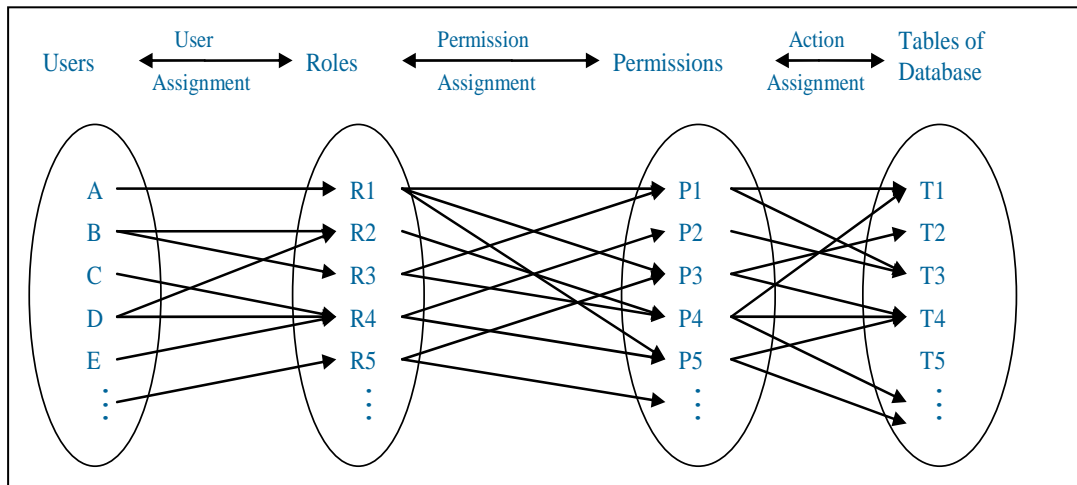


Figure 3. Role-based Access Control

However, RBAC has several weaknesses. Firstly, the nature of roles is static, so RBAC lacks flexibility and responsiveness to the environment in which they are used. Secondly, RBAC does not encompass the overall context associated with any collaborative activity [10], so it is a passive security system that serves the function of maintaining permission assignments. Thirdly, RBAC lacks the ability to specify a fine-grained control on individual users in certain roles and on individual object instances [10], so it is not enough for collaborative environments.

2.2.3 Task-based Authorization Control Model

Task-based Authorization Controls (TBAC) [15] is a task-oriented model for access control and authorization [13]. It is an active security model that is well suited for agent-based distributed computing and workflow management that allows lots dynamic information processing activities with multiple points of access, control, and decision

making. An active security system takes into account the impact of context as it emerges with progressing tasks and distinguishes task-based and context-based permission activation from permission assignment [10].

TBAC focuses on security modeling and enforcement at the application and enterprise level. In the TBAC paradigm of access control, permissions are associated with contextual information about ongoing activities when evaluating an access request. And permissions are checked-in and checked-out from protection states in a just-in-time fashion based on activities or tasks and synchronized with the processing of authorizations in progressing tasks. Thus, TBAC dynamically manages permissions as authorizations by progress to completion and minimizes the vulnerabilities in a system. TBAC keeps track of the usage and consumption of permissions, thereby preventing the abuse of permissions through unnecessary and malicious operations [15]. However, in TBAC, there are no contexts in relation to activities, tasks, or workflow progress and it only keeps track of usage and validity of permissions. This is insufficient for collaborative systems that require a much broader definition of context. More fine-grained components need to be defined to support dynamic environments motivated by TBAC [10]. TBAC can be used effectively in an application or enterprise, but for most collaborative environments, TBAC is used within other access control models.

2.2.4 Context based Security Model

The Context-based security [16] emerged recently as a new approach to cope with the new types of security problems introduced by the high mobility of pervasive systems and the heterogeneity of devices used in these types of environments [16]. A Context

based security model treats context as a first-class principle both in the specification and enforcement of policies [17]. It models and represents the contexts in which agents operate and to which policies are associated, defines what actions are permitted or forbidden on resources in specific contexts, defines the actions that must be performed on resources in specific contexts, and dynamically associates agents with contexts [17].

In the context based security model, contextual graphs help specifying context-based security policies in a pervasive environment and are used as a management tool that eases security administration for complex environments with many heterogeneous services and devices [16]. The exploitation of context as a mechanism for grouping applicable policies (not as a limit to applicability of already retrieved policies as in traditional access control solutions) simplifies access control management by increasing policy specification reuse and by simplifying policy update and revocation [16]. It also includes fine-grained control, Policy specifications, and Policy enforcement characteristics. However, this model is pretty new and requires further testing within the collaborative systems domain [10].

Figure 4 describes the conceptual model of context-based security policy, which is defined as an association between a context and an action in pervasive environment. When the authorized user enters a certain security context, the context will be associated with the corresponding action automatically. The obligated user cannot access the security context. Here context can be any useful information about the world, such as the user location, the characteristics of the underlying device, relationship with other users and many others. Actions can be acted itself and the specific applications.

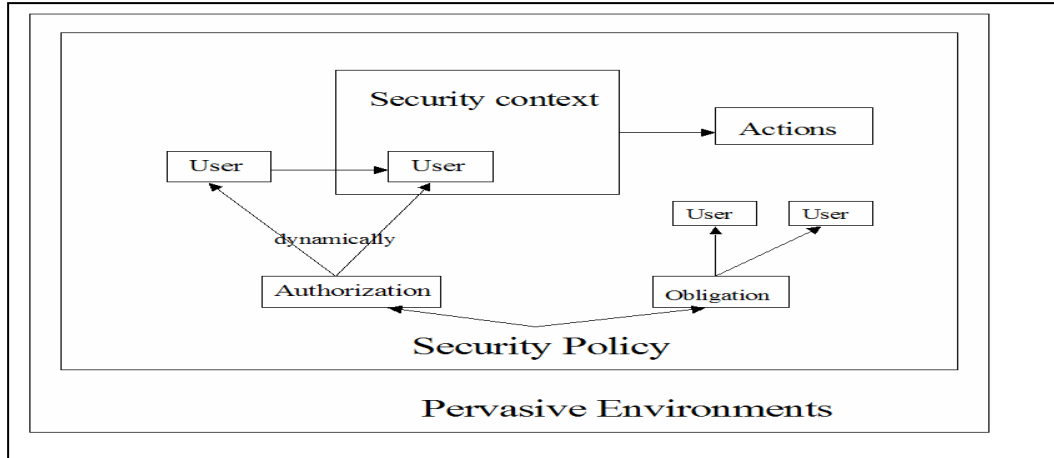


Figure 4. Context-based Security Policy

2.3 Collaborative environment and E-healthcare

Collaborative environments provide systems that allow groups of users to communicate and cooperate on common tasks in distributed locations. Collaborative systems include a wide range of applications such as collaborative document sharing/editing, conferencing, distance learning, workflow management systems, and so on [10]. The applications deployed in such systems create, manipulate, and provide access to a variety of protected information and resources [10].

E-healthcare involves in a wide range of individuals and cooperative organizations. Some organizations (such as clinics or medical insurance companies) in the hospital are collaborative and yet competing each other. In order to better understand the collaborative environments, Figure 5 shows document sharing in an ideal workflow for the Leon county uninsured e-healthcare program in Tallahassee, FL.

Table 4. Notations for Document Sharing in Case Study

Bond Community Server	Bond Community Center Clinic Stored DOH Server
LNHS Server	Lincoln Neighborhood Service Center Clinic DOH Server

TMH Server	Tallahassee Memorial Hospital Emergency Room Server
CRMC Server	Capital Regional Medical Center Emergency Room Server

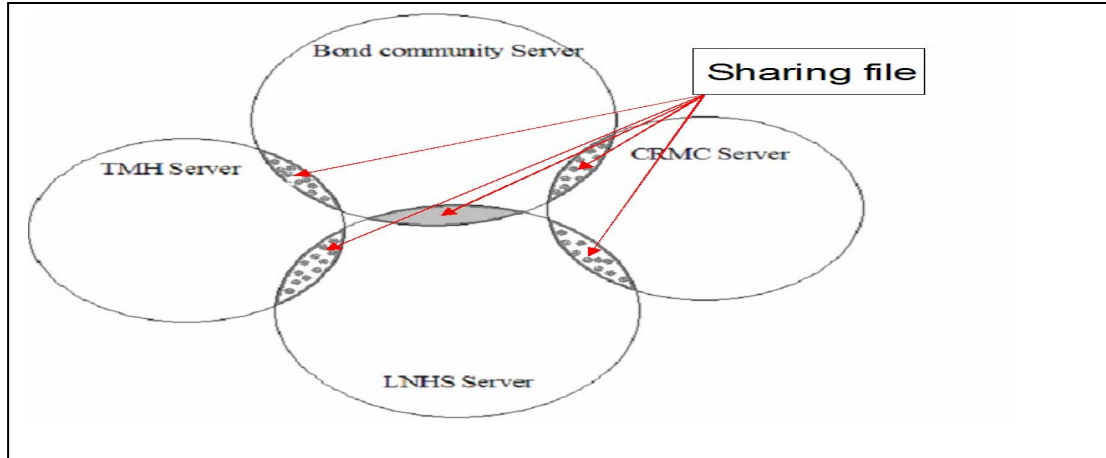


Figure 5. Document sharing in the ideal workflow of Leon county uninsured e-healthcare program in Tallahassee, FL

In figure 5, the servers, which belong to different organizations (see Table 4), store the patients' data. The dotted areas are the file sharing between different collaborative organizations. The TMH ER Server and LNHS Server share the uninsured patients' medical information showed in dot painted part. And it is same for the TMH ER Server and the Bond community Server, the LNHS Server and the CRMC ER Server. The grey area indicates the sharing of prescriptions of some patients between LNHS Server and Bond community Server.

2.4 Smart Environment and Future E-healthcare

In the future, e-healthcare will exist in smart environments. Smart environments -- pervasive computing environments -- are those in which people and devices are mobile and use various wireless networking technologies to discover and access services and devices in their vicinity [18]. They consist typically of a large

heterogeneous collection of networked devices [19]. And a large number of wirelessly networked heterogeneous users, services and semi-automated entities of varied capabilities are populated to these environments. Such pervasive environments are extremely dynamic in nature.

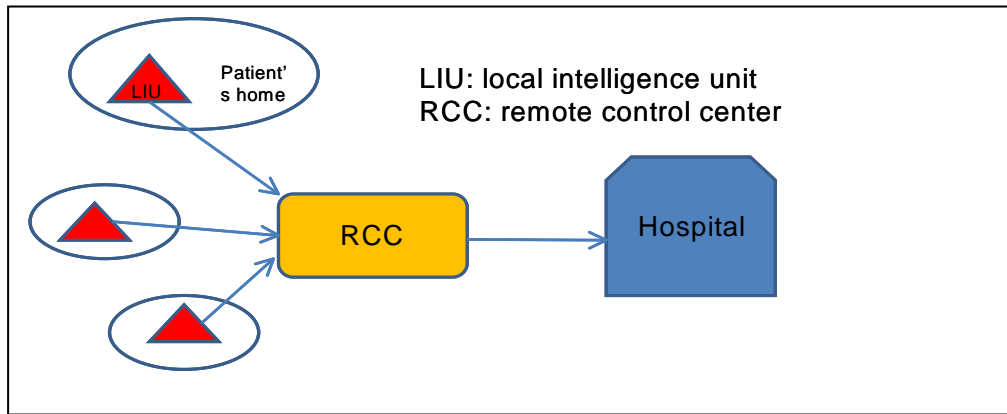


Figure 6. Health Smart Home

The Health Smart Home (HSH) is not new to many people. The Health Smart Home is shown in figure 6. It is a home equipped with special electronics to enable the remote control of automated devices specifically designed for remote health care [20]. The target HSH patients include disabled and elderly people living alone; elderly couples, one or both of whom are affected with disease; people who lose autonomy; and patients with chronic disease [20]. In the HSH system, automatic devices and various sensors are installed at patient's home, to make sure of the safety of the patient and to supervise their health status. They are linked to a Local Intelligence Unit (LIU). Then the LIU is connected with a Remote Control Center (RCC). The network includes wireless or wired users, services and devices. So there is an increased need for more automated security in the resulting pervasive environments [18].

Chapter 3 Security Rules for E-HealthCare

Privacy is important to e-healthcare systems. E-healthcare's corporations should protect the privacy and confidentiality of their customers and those individuals' health care information. In 1996, the Health Insurance Portability and Accountability Act (HIPAA) announced Public Law 104-191. The act "specifies the privacy, security and electronic transaction standards with regard to patient information for all health care providers" [1]. HIPAA also addresses the security rules and privacy rules of health data. Adopting these standards will improve the efficiency and effectiveness of the nation's healthcare systems through encouraging the widespread use of electronic data interchange in healthcare industry.

In this chapter we will discuss the security rule of the Health Insurance Portability and Accountability Act (HIPAA) [21] for Electronic Protected Health Information (EPHI) in e-healthcare.

HIPAA had two primary purposes for developing security standards. First, and foremost, the implementation of appropriate security safeguards protects certain electronic health care information that may be at risk. Second, protecting an individual's health information, while permitting the appropriate access and use of that information, ultimately promotes the use of electronic health information in the industry [21]. All HIPAA covered entities must comply with the Security Rule. The covered entities [21] include:

1. **Covered Health Care Providers** - Any provider of medical or other health care services or supplies who transmits any health information in electronic form in connection with a transaction for which HHS has adopted a standard.
2. **Health Plans** - Any individual or group plan that provides or pays the cost of health care
3. **Health Care Clearinghouses** - A public or private entity that processes another entity's health care transactions from a standard format to a non-standard format, or vice-versa.
4. **Medicare Prescription Drug Card Sponsors** –A nongovernmental entity that offers an endorsed discount drug program under the Medicare Modernization Act.

3.1 The Security Rule

The Final Rule on Security Standards was issued on February 20, 2003. It took effect on April 21, 2003 with a compliance date of April 21, 2005 for most covered entities and April 21, 2006 for “small plans.” The Security Rule complements the Privacy Rule. While the Privacy Rule pertains to all Protected Health Information (PHI) including paper and electronic, the Security Rule deals specifically with Electronic Protected Health Information (EPHI). It lays out three types of security safeguards required for compliance: administrative, physical, and technical. For each of these types, the Rule identifies various security standards, and for each standard, it names both required and addressable implementation specifications. Required specifications must be adopted and administered as dictated by the Rule. Addressable specifications are more flexible.

Individual covered entities can evaluate their own situation and determine the best way to implement addressable specifications. [25]

3.1.1 Administrative Safeguards

The Security Rule defines administrative safeguards as, “*administrative actions, and policies and procedures, to manage the selection, development, implementation, and maintenance of security measures to protect electronic protected health information and to manage the conduct of the covered entity’s workforce in relation to the protection of that information.*” [22]

The standards and the implementation specifications of Administrative Safeguards are describe in the following table 3. [22]

3.1.2 Physical Safeguards

The Security Rule defines physical safeguards as “*physical measures, policies, and procedures to protect a covered entity’s electronic information systems and related buildings and equipment, from natural and environmental hazards, and unauthorized intrusion.*” [23]

The standards and implementation specifications of physical safeguards are describe in the table 4. [23]

3.1.3 Technical Safeguards

The Security Rule defines technical safeguards in § 164.304 as “*the technology and the policy and procedures for its use that protect electronic protected health information and control access to it.*” [24]

The standards and implementation specifications of technical safeguards are described in table 5. [24]

Table 5. Specifications of Administrative Safeguards

ADMINISTRATIVE SAFEGUARDS			
Standards	Sections	Implementation Specifications (R)= Required, (A)=Addressable	
Security Management Process	§ 164.308(a)(1)	Risk Analysis	(R)
		Risk Management	(R)
		Sanction Policy	(R)
		Information System Activity Review	(R)
Assigned Security Responsibility	§ 164.308(a)(2)		
Workforce Security	§ 164.308(a)(3)	Authorization and/or Supervision	(A)
		Workforce Clearance Procedure	(A)
		Termination Procedures	(A)
Information Access Management	§ 164.308(a)(4)	Isolating Health Care Clearinghouse Functions	(R)
		Access Authorization	(A)
		Access Establishment and Modification	(A)
Security Awareness and Training	§ 164.308(a)(5)	Security Reminders	(A)
		Protection from Malicious Software	(A)
		Log-in Monitoring	(A)
		Password Management	(A)
Security Incident Procedures	§ 164.308(a)(6)	Response and Reporting	(R)
Contingency Plan	§ 164.308(a)(7)	Data Backup Plan	(R)
		Disaster Recovery Plan	(R)
		Emergency Mode Operation Plan	(R)
		Testing and Revision Procedures	(A)
		Applications and Data Criticality Analysis	(A)
Evaluation	§ 164.308(a)(8)		
Business Associate Contracts and Other Arrangements	§ 164.308(b)(1)	Written Contract or Other Arrangement	(R)

Table 6. Specifications of Physical Safeguards

PHYSICAL SAFEGUARDS			
Standards	Sections	Implementation Specifications (R)= Required, (A)=Addressable	
Facility Access Controls	§ 164.310(a)(1)	Contingency Operations	(A)
		Facility Security Plan	(A)
		Access Control and Validation Procedures	(A)
		Maintenance Records	(A)
Workstation Use	§ 164.310(b)		
Workstation Security	§ 164.310(c)		
Device and Media Controls	§ 164.310(d)(1)	Disposal	(R)
		Media Re-use	(R)
		Accountability	(A)
		Data Backup and Storage	(A)

Table 7. Specifications of Technical Safeguards

TECHNICAL SAFEGUARDS			
Standards	Sections	Implementation Specifications (R)= Required, (A)=Addressable	
Access Control	§ 164.312(a)(1)	Unique User Identification	(R)
		Emergency Access Procedure	(R)
		Automatic Logoff	(A)
		Encryption and Decryption	(A)
Audit Controls	§ 164.312(b)		
Integrity	§ 164.312(c)(1)	Mechanism to Authenticate Electronic Protected Health Information	(A)
Person or Entity Authentication	§ 164.312(d)		
Transmission Security	§ 164.312(e)(1)	Integrity Controls	(A)
		Encryption	(A)

In Tables 5-7, each set of safeguards consists of a number of standards, which are generally comprised of a number of implementation specifications that are either required or addressable. If an implementation specification is required, the covered entity must implement policies and/or procedures that meet what the implementation specification requires. If an implementation specification is addressable, then the covered entity must assess whether it is a reasonable and appropriate safeguard in the entity's environment.

This involves analyzing the specification in reference to the likelihood of protecting the entity's EPHI from reasonably anticipated threats and hazards. If the covered entity chooses not to implement an addressable specification based on its assessment, it must document the reason and, if reasonable and appropriate, implement an equivalent alternative measure. Addressable does not mean optional. [21] For example, an online bank must implement the Automation Logoff specification which is addressable in order to keep the security of customer's account and the time of Automation Logoff will be just few minutes after non-operation in the service. While for some online store, you may not implement the Automation Logoff specification if the store is pretty small. Or you may give a long time in an implementation of the Automation Logoff specification

According to the rules of HIPAA, EPHI is characterized in terms of confidentiality, integrity and availability by all covered entities. The EPHI files now may also include the medical digit image, such as CAT scans, X-rays, EKG tapes, blood and DNA analyses, and so on. A medical digit image often includes two parts, an image header and an image body. No matter what, the EPHI files should be protected by all security means during transmission and access by covered entities according to HIPAA rules.

3.2 The implementation of Security Rules in RBAC security model

The RBAC model's security rules that comply with HIPAA Security standard are shown below:

1) Access control: Only authorized customers, employees, agents or independent contractors are permitted to access Personal Information. All these authorized people

must agree in writing to abide by privacy policy. Example policies are: the patient can access his/her record and prescription; the doctor can write and read his/her patient's record and prescription; the nurse can write and read his/her patient's record and read prescription; the pharmacist can read the patient's prescription.

2) Database Security: The system database requires authorized client login and password verification. For example, the remote access should be from authorized login with password verification.

3) Data Security: Strict security measures are adopted to protect data stored on servers from unauthorized access.

4) Data Encryption: Data being stored to and information being transmitted back and forth between web-servers and your browser is protected with 128-bit SSL (secure socket layer) encryption technology. For example: a digital medical image, a patient's record, or prescriptions should be encrypted when transmitted.

5) Automatic Logoff: Users' sessions will automatically log off after 30 minutes inactivity on systems.

6) Data Backup & Archives: Data backups are done using DVD or optical disc backups. Disc backups will occur at regular intervals and be stored off-site.

7) Digital Signature: When a user invoices special applications, the user should sign his/her digit signature as the last step of using the application. For example: the pharmacist should sign his/her digital signature to finish picking up a shipment of the prescriptions.

Chapter 4 Web Services for E-Healthcare

Today's businesses can function more efficiently as the information technology develops. Each company already has software developed and customized to its specific needs. It is critical for companies to share information stored in isolated computer systems. The Internet solves this problem to some extent. However, the Internet opens many loopholes in security, making the owners of this information uneasy about the scope of their information's availability.

Hence, for better B2B (Business-to-Business) communication, the systems must have the ability to link up to each other, grant permissions through a system other than the Internet, and which would make all the systems network with each other like an Intranet. Web Services is one of the right solutions to such a problem. Web Services is based on the already existing and well-known HTTP protocol and uses XML as the base language, making it a very developer-friendly service system. Also, Web Services already has a set of standardized rules and specifications. That makes applications more portable.

4.1 Overview of Web Services

Web services [26] are a new breed of Web applications. Web services technology has reached a level of maturity sufficient to evolve from a promising technology to a reality; a reality on which IT departments are basing their operations in order to achieve a direct alignment with the business operations that they support [40]. The Stencil Group defines them as "loosely coupled, reusable software components that semantically

encapsulate discrete functionality and are distributed and programmatically accessible over standard Internet protocols”. [26] Some defining characteristics of web services are in table 6 [27].

Table 8. Characteristics of Web Services

Characteristics	Explanations and Advantages
Reusable software components	<i>The component-based model allows developers to reuse the building blocks of code created by others to assemble and extend them in new ways.</i>
Software components are loosely coupled	<i>Traditional application design depends upon a tight interconnection of all subsidiary elements. The complexity of these connections requires that developers thoroughly understand and have control over both ends of the connection; moreover, once established, it is exceedingly difficult to extract one element and replace it with another. Loosely coupled systems, on the other hand, require a much simpler level of coordination and are more flexible.</i>
web services semantically encapsulate discrete functionality	<i>The component describes its own inputs and outputs in a way that other software can determine what it does, how to invoke its functionality, and what result to expect in return.</i>
web services can be accessed programmatically	<i>Unlike web sites and desktop applications, web services are not designed for direct human interaction, and they do not have a graphical user interface. Rather, web services operate at the code level; they are called by and exchange data with other software.</i>
web services are distributed over the Internet	<i>Web services make use of existing, ubiquitous transport protocols like HTTP, leverage existing infrastructure and can comply with current corporate firewall policies.</i>

Web services have two distinct layers [27]: core layers and high-level layers. Core Layers of the Web Services Stack include common internet protocols, extensible markup language (XML) and simple object access protocol (SOAP). Higher-level players include: Web Services Description Language (WSDL), Universal Description,

Discovery, and Integration (UDDI), Web Services Flow Language (WSFL) and Other Business Rules.

Table 9. Layers of the Web Service Stack

Core Layers of the Web Services Stack	Common Internet Protocols	<i>Web services build on ubiquitous Internet connectivity and infrastructure to ensure nearly universal reach and support. In particular, web services take advantage of HTTP, the same connection protocol used by web servers and browsers.</i>
	Extensible Markup Language (XML)	<i>XML is a widely accepted format for exchanging data and its corresponding semantics. It is a fundamental building block for nearly every other layer in the web services stack.</i>
	Simple Object Access Protocol (SOAP)	<i>SOAP is a protocol for messaging and RPC-style communication between applications. It is based on XML and uses common Internet transport protocols like HTTP to carry its data.</i>
Higher- Level Layers of the Web Services Stack	Web Services Description Language (WSDL)	<i>WSDL is an XML-based description of how to connect to a particular web service. A WSDL description abstracts a particular service's various connection and messaging protocols into a high-level bundle and forms a key element of the UDDI directory.</i>
	Universal Description, Discovery, and Integration (UDDI)	<i>UDDI represents a set of protocols and a public directory for the registration and real-time lookup of web services and other business processes.</i>
	Web Services Flow Language (WSFL)	<i>WSFL is the least developed of the current web services layers. Sponsored by IBM, the WSFL team hopes to define a framework that implementers of web services can use to describe the business logic required to assemble various services into an end-to-end business process.</i>
	Other Business Rules	<i>Additional elements that support complex business rules must still be implemented before web services can automate truly critical business processes (example Security and authentication, contract management,</i>

		<i>quality of service</i>).
--	--	------------------------------

4.2 Web Services Security

The IT industry has been talking about Web services for almost seven years. The benefits of having a loosely-coupled, language-neutral, platform-independent way of linking applications within organizations, across enterprises, and across the Internet are becoming more evident as Web services are used on a wide-scale in production. [28] While security is still the mainstream in Web services since the customers, industry analysts, and the press identify a key area that needs to be addressed. Ensuring the integrity, confidentiality and security of Web services is critical, both for organizations and their customers, especially in the medical industry.

A set of Web services security standards has come into being recently. OASIS (Organization for the Advancement of Structured Information Standards) has proposed Web Service-Security [29]. This OASIS specification proposes a standard set of SOAP extensions that can be used when building a secure Web service in order to implement message content integrity and confidentiality [29]. Message content integrity is provided by an XML Signature mechanism and message content confidentiality is provided by an XML Encryption mechanism. These two mechanisms can be integrated into the header and the body of a SOAP message and may be sent via any transport channel without disclosing security.

Beside transport level security extensions, a variety of standards provide means to manage and exchange security policies [30]. XACML [30] is a standard to define access control for resources in a system. The Security Assertion Markup Language (SAML) [31] is a standard for the exchange of security tokens. WS-Policy [31] defines a framework for

allowing web services to express their security constraints and requirements. WS-Security Policy [30] is a complementary standard to WS-Policy and specifies how actors can assert to potential partners their policies with respect to specific WS-Security mechanisms. WS-Trust [32] enables applications to construct trusted [SOAP] message exchanges through the exchange, the issuance and the validation of security tokens by a trusted third party.

4.3 E-Healthcare and Web Security

The critical part of e-Healthcare web services is to secure patients' personal medical records. As web security gets more and more mature, e-healthcare turns into reality sooner. However, there are some crucial major security issues which web services technologies must address, as described below:

Authorization: Web services should have mechanisms which control access to the services being provided. For example: a mechanism determine who and how can do what and how on the resources or special data.

Authentication: each web service that has an interaction with other parties may be required to provide authentication credentials.

Confidentiality: the information should be exchanged among web services nodes securely.

Integrity: the information received by a web service should remain the same as the information that was sent from the client.

Non-repudiation: a client can get the certification of utilization for a service and the service can provide the certification to a client. This security issue can be implemented by digital signatures.

Availability: availability has two important aspects: preventing Denial-of-Service attacks and arranging for redundancy in the system.

Chapter 5 RBAC model and Its Implementations

This chapter explains the RBAC model for e-healthcare enterprise case study, XACML and the RBAC profile of XACML v2.0 which implement RBAC: assigning role attributes, specifying permissions, and then authorizing access.

5.1 RBAC Model

Since the essence of Role-based Access Control [14] [RBAC] is that permissions are organizationally assigned to roles and users are administratively assigned to appropriate roles, it is quite suitable for the e-healthcare system. In the e-healthcare system, the object might be patient records, prescriptions, images, etc; the roles might be physician, nurse, patient, administrator, medical student, etc. These roles are fixed and group-oriented. This can reduce the complexity for the security administration in e-healthcare. Authorization-to-user is the most important thing in e-healthcare. RBAC provides a means of naming and describing many-to-many relationships between individuals and rights. So RBAC can well control the user's authorization. And control of protected resources is often based on employee functions (or roles) rather than data ownership in e-healthcare. Figure 7 demonstrates an example of the RBAC model in e-healthcare system.

In fact, roles are not only associated with permissions, but also associated with other roles and their respective permissions. To extend the application of RBAC, a role hierarchy is implemented. This hierarchy represents responsibility and specialization. It also limits redundant permission definition within roles and provides a more intuitive

view of permissions to ease the work of the role administrator. [33] Figure 8 presents an example of role hierarchy in e-healthcare system. In Figure 8, Surgeon and Radiologist are sub-roles of Physician role. A surgeon has the permission to operate. A radiologist has the permission to perform X-Ray.

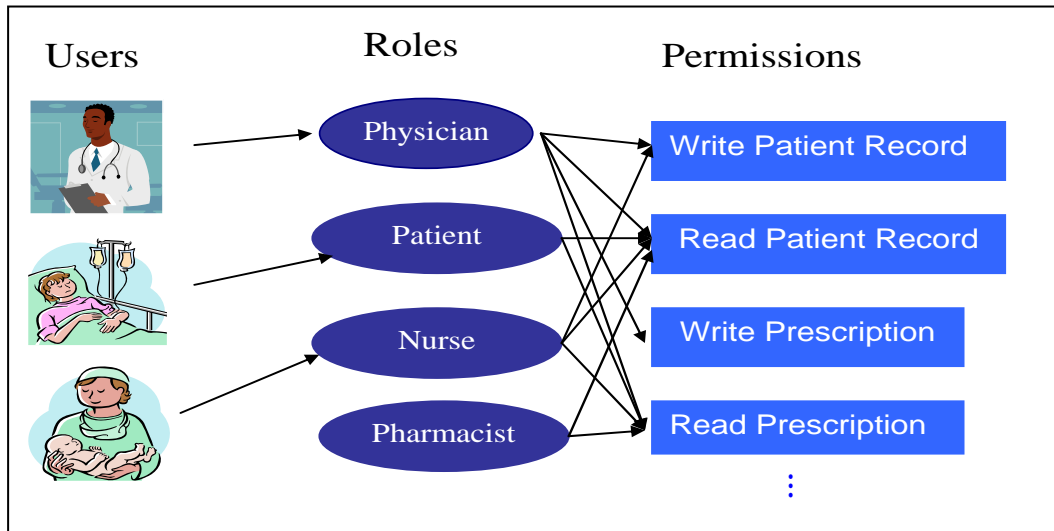


Figure 7. Role-based Access Control Model

Note:

Users: A user is a human being or an independent agent

Roles: A role is a kind of job function in an organization which has some associated permissions to secured resources and data.

Permissions (including objects and operations): Permission is an authorized right to access or perform operations on protected resources. Objects are the protected resource or tasks in the system. An operation is an executed action performed by a user.

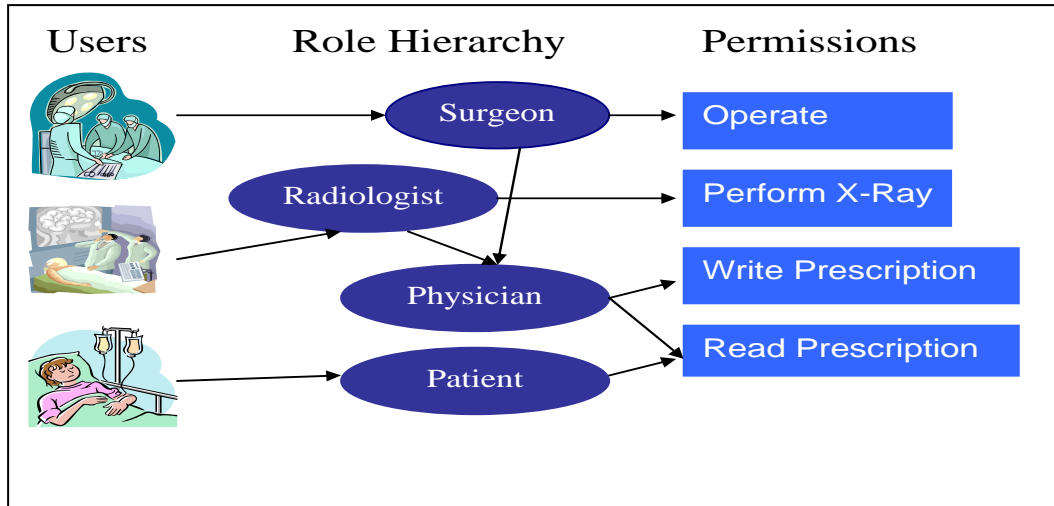


Figure 8. Role Hierarchy RBAC

5.2 eXtensible Access Control Markup Language (XACML)

XACML is an OASIS standard that defines a general policy language used to protect resource as well as an access decision language based on XML schema. The XACML policy language allows administrators to define the access control requirements for a protected application resource. The XACML access decision language is used to represent a query to ask whether a given action should be allowed for a resource. If a policy is found for the queried resource, attributes in the request are compared against attributes contained in the policy rules. Finally a response is determined including an answer about whether the request should be allowed using one of four values: Permit, Deny, Indeterminate (an error occurred or some required value was missing, so a decision cannot be made) or Not Applicable (the request cannot be answered by this service).

5.2.1 XACML Architecture

A Typical XACML Architecture [33] is shown in Figure 9.

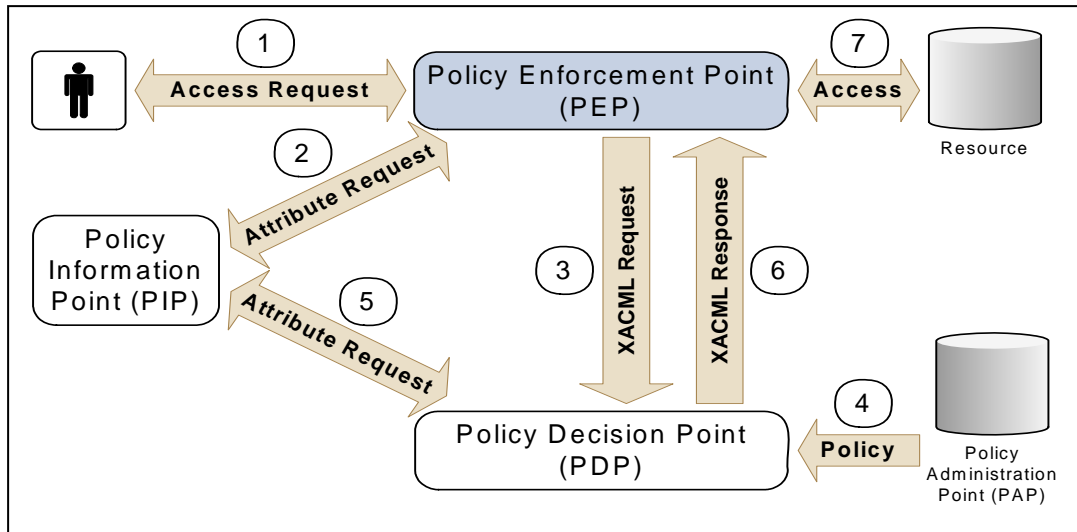


Figure 9. XACML Architecture

We provide a legend for figure 9:

1. A user sends an access request to a Policy Enforcement Point (PEP), the access requests are filtered by PEP.
2. The PEP retrieves attribute value from the Policy information Point (PIP) to form a formal XACML request.
3. A XACML request formed with the subject, resource and action attributes by PEP is forwarded to the Policy Decision Point (PDP).
4. The PDP gathers authorization policies that apply to this request.
5. The PDP gathers additional attributes needed to compare the request to the policies.
6. A comparison is made by the PDP and a decision is returned as an XACML response to the Policy Enforcement Point (PEP).
7. The PEP permits or denies access to the protected resources.

5.2.2 *XACML Policy Language*

XACML Policy Language is an XML extension language used to specify and enforce authorization policy.

5.2.2.1 Policy language components

The XACML policy language model [34] is shown in Figure 9. The main components of the XACML Policy Language Model are:

- **PolicySet** containing the following subcomponents:
 - a) A set of Policies, or PolicySets are as well as references to remote policies.
 - b) A Target to which this PolicySet is intended to apply
 - c) An Obligation, i.e. the actions that must be performed by the PEP in conjunction with the enforcement of an authorization decision.
 - d) A Policy combining algorithm, used to reconcile multiple results into a single decision.

Note: A PolicySet may contain multiple Policies that apply to the request and each policy may result in a different access decision, so the Policy Combining Algorithm is required to reconcile multiple results into a single decision.

- **Policy** contains the following subcomponents:
 - a) A set of rules
 - b) A target, which this policy is intended to apply
 - c) An obligation, which is the actions that must be performed by the PEP in conjunction with the enforcement of an authorization decision.

d) A rule-combining algorithm, which is used to reconcile multiple results into a single decision.

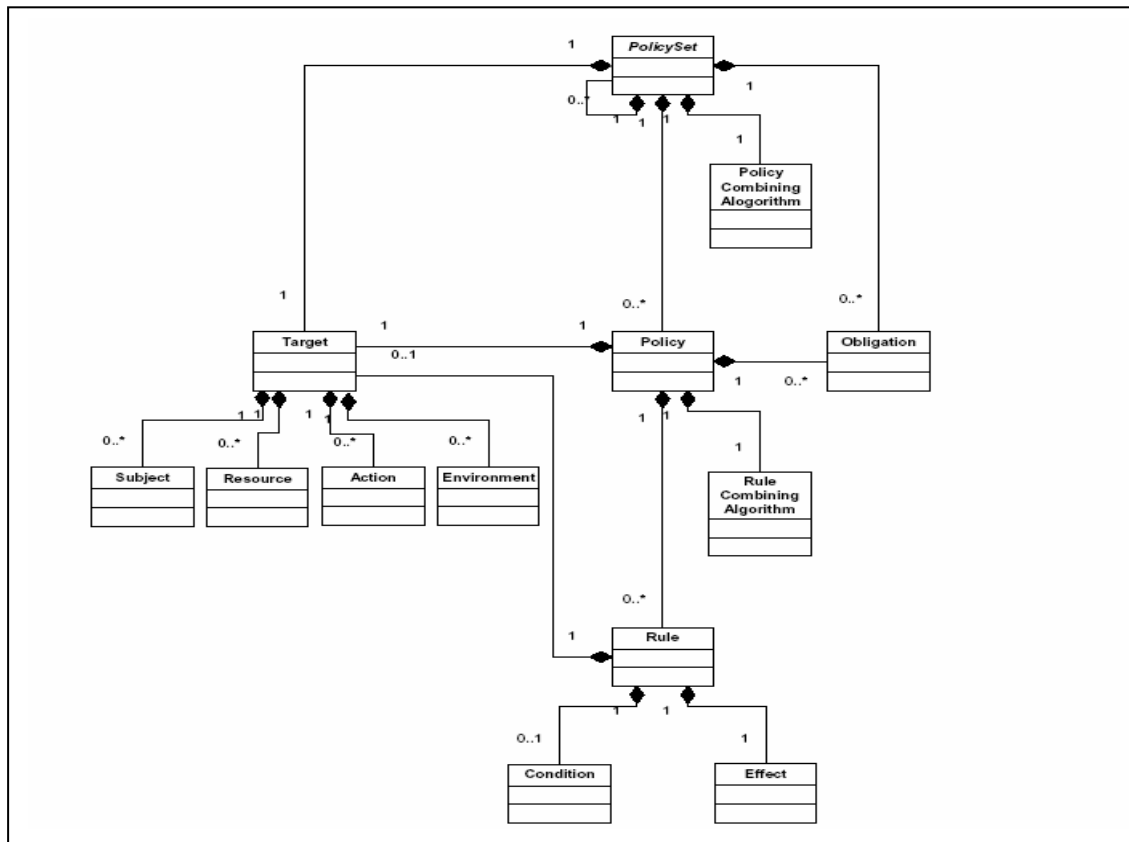


Figure 10. XACML Policy Language Model [34]

- **Rule** contains following subcomponents:

- a) A target, which this rule is intended to apply
- b) A condition, which is a Boolean expression used to evaluate attributes.
- c) An effect, which is an access decision defined by policy writer. Its value only can be “Permit” or “Deny”.

5.2.2.2 An example of XACML Policy

Here we give an easy XACML Policy example to better understand the XACML Policy Language. Table 10 demonstrates an access control policy that states anybody is allowed to perform a read action on the protected file “PatientList”.

In Table 10, Line 1 and Line 2 define the name of the policy and specify the algorithm that is used to resolve the result of the rule. In this instance, “permit-overrides” algorithm is specified so if any rule evaluates to “permit”, then the policy returns “Permit”.

Lines 3 through 13 define which subjects, resources, and actions the policy applies. In this instance, the target section says that the policy is applicable to any decision request.

Line 14 defines a rule in this example and its effect is “Permit” if the rule is evaluated to “True”.

Lines 15 through 40 define the decision requests to which subject, resource, and action that the rule applies.

Lines 15 through 18 describe the rule that applies the decision requests to any subject.

Lines 19 through 29 describe the rule that applies the decision requests to a specific resource value, which must match the <ResourceMatch> element. The <ResourceMatch> element specifies a matching function in the MatchId attribute, a string value “PatientList”, and a pointer to a specific “resource attribute” in the request context, by means of the <ResourceAttributeDesignator> element. This match function is used to compare the string value with the value of the “resource attribute” in the request context. This rule applies to a decision request, only if the match returns “True”.

Table 10. XACML Policy Example

1	<Policy PolicyId="policy1" RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-
2	combining-Algorithm:permit-overrides">
3	<Target>
4	<Subjects>
5	<AnySubject/>
6	</Subjects>
7	<Resources>
8	<AnyResource/>
9	</Resources>
10	<Actions>
11	<AnyAction/>
12	</Actions>
13	</Target>
14	<Rule RuleId="Rule1" Effect="Permit">
15	<Target>
16	<Subjects>
17	<AnySubject/>
18	</Subjects>
19	<Resources>
20	<Resource>
21	<ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
22	<AttributeValue
23	DataType="http://www.w3.org/2001/XMLSchema#string">PatientList</AttributeValue>
24	<ResourceAttributeDesignator
25	AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
26	DataType="http://www.w3.org/2001/XMLSchema#string"/>
27	</ResourceMatch>
28	</Resource>
29	</Resources>
30	<Actions>
31	<Action>
32	<ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
33	<AttributeValue
34	DataType="http://www.w3.org/2001/XMLSchema#string">read</AttributeValue>
35	<ActionAttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
36	DataType="http://www.w3.org/2001/XMLSchema#string"/>
37	</ActionMatch>
38	</Action>
39	</Actions>
40	</Target>
41	</Rule>
42	</Policy>

Lines 30 through 39 define the action match in a similar format to a resource match. It defines the match of “action attribute” in the request context to string value “read”.

Line 41 closes the Rule element and line 42 closes the Policy element.

5.2.3 RBAC Profile of XACML

This OASIS standard “defines a profile for the use of the OASIS eXtensible Access Control Markup Language (XACML) to meet the requirements for “core” and “hierarchical” role based access control (RBAC) as specified in [ANSI-RBAC Standard].” [35]

This profile specifies four types of policies to implement RBAC.

1. Permission <PolicySet> or PPS: contains all actual permissions associated with a given role. It contains <Policy> elements and <Rules> that describe the resources and actions that subjects are permitted to access. [35] A given Permission <PolicySet> may also contain references to Permission <PolicySet>s associated with other roles that are junior to the given role, thereby allowing the Permission <PolicySet>. The <Target> element of a Permission <PolicySet>, if present, must not limit the subjects to which the <PolicySet> is applicable [35]. Figure 10 shows the Permission PolicySet.)

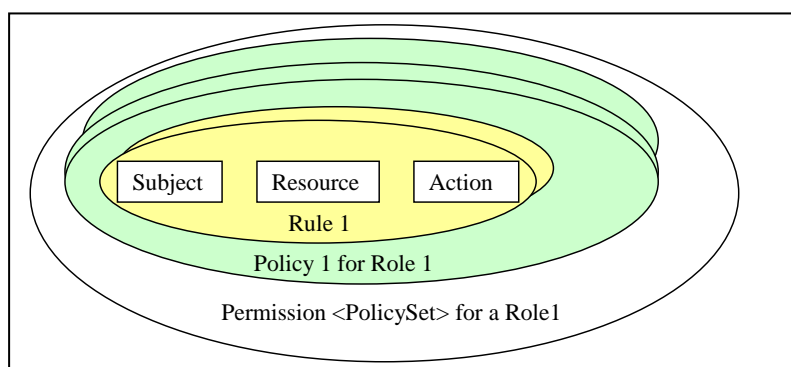


Figure 11. Permission PolicySet

(Note: The < > represents the tag which is created by the author of the XML document. XML allows the author to define his own tags and his own document structure.)

2. Role <PolicySet> or RPS: associates a role with a PPS that contains the actual permissions associated with the given role. Each RPS contains a <Target> element indicating the applicable role and a reference to the single corresponding PPS. Figure 11 shows the Role PolicySet.
3. Role Assignment <Policy> or <PolicySet>: defines which roles are enabled or assigned to which subjects. This type of policy is optional.
4. HasPrivilegeOfRole <Policy>: supports requests asking whether a subject has the privileges associated with a given role. This type of policy is optional.

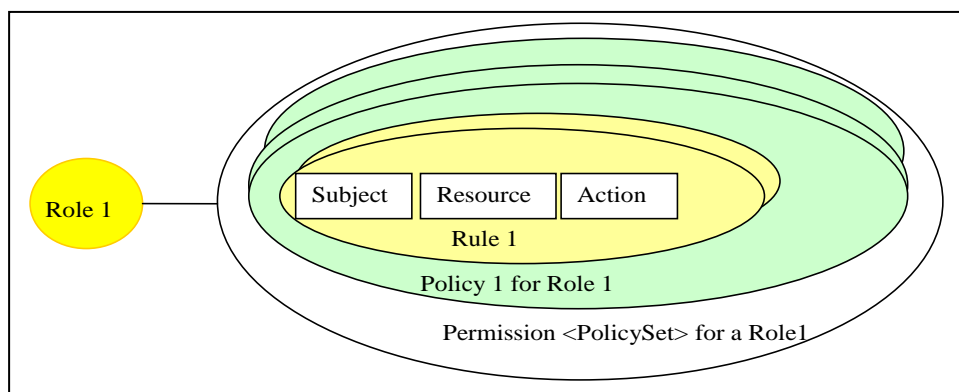


Figure 12. Role PolicySet [37]

5.2.4 Current State of the Art: choosing an XACML Implementation

Currently, there are very few tools available for working with XACML, and most of these are in the early experimental stage. Sun's XACML Implementation was originally created in Sun Microsystems Research Laboratories by members of the Internet Security Research Group. It is an open source implementation of the OASIS XACML standard 1.x, written in the JavaTM programming language. Sun's XACML Implementation provides complete support for all the mandatory features of XACML and a number of optional features. It also provides support for parsing both policy and

request/response documents, determining applicability of policies, and evaluating requests against policies. There are APIs for adding new functionality as needed and writing new retrieval mechanisms for finding things like policies and attributes. All of the standard attribute's types, functions, and combining algorithms are supported [36].

Chapter 6 A Case Study of E-healthcare System

In this chapter, we will focus on the implementation of the RBAC (Role-based access control) for an e-healthcare system, which is not a trivial task. Healthcare involves many participants and organizations. The required security authorizations are therefore very complicated and difficult to administer. The actors can be divided into different roles in an e-healthcare system. There are several essential roles, such as system administrator, physician, patient, office staff of Clinic, etc. The roles are defined by policies of an organization. In e-healthcare, policies define the identification of a role and the authentication for each role. In this thesis, an administrator can control the whole e-healthcare system and implement authentication: assign a role to a user and assign a patient to a physician. A physician can read and write his patients' medical records. Office staff of a clinic can manage active/inactive patients and discharge patients for physicians. The patient can read his medical records and update his personal information online.

We choose Microsoft Visual Studio 2008 Express as the development environment and the C# language as the programming language to implement the RBAC security model in our case study for this thesis. Microsoft SQL Server Management Studio Express (SSMSE) provides a graphical management tool for SQL Server 2005 Express.

We proposed 7 security rules in our RBAC security model in Chapter 3.2. The prototype for e-healthcare system we implement achieves two security rules: access control based on RBAC and automatic log-off based on a time interval that

administer can set up. We also provide a delegation function for physicians.

The chapter proceeds as follows: Firstly, we will introduce the structure of our e-healthcare system, a 3-Tier Architecture. Secondly, we will illustrate the design of its database. Thirdly, we will illustrate the roles and their windows applications of the e-healthcare system. We will discuss their sub-system functions by Data Flow Diagram and demonstrate their interfaces. Web-based applications were also designed for non-technical users such as patients and physicians who are out of office. They will be discussed within these roles. And finally, we will examine the access control in our prototype.

6.1 Three-Tier Architecture

Microsoft Visual Studio 2008 provides a way to create a 3-tier architecture approach that consists of three layers, and uses data-transfer objects to pass data back and forth. As the name implies, three-tier architecture, which is client-server architecture, has three layers: presentation tier or GUI tier, business logic tier and data access tier. The three layers are developed and maintained as independent modules and most often on separate platforms [38]. Figure 13 illustrates the 3-Tier Architecture.

In Visual Studio 2008, the ASP.NET website or the windows application can be the presentation layer. This layer is what everyone sees and uses. The business logic tier validates the input conditions before calling a method from the data layer and ensures the correct output. Any calculations or other actions can take place in the business layer. In Visual Studio 2008, you can use the DataSet, DataTable, DataRow and SqlHelper classes to build the data access layer. The three tiers can be developed separately. Because of the

separation of business logic from the user interface and database access, the business logic component can be reusable. Changing the data access layer does not affect the business logic module significantly. So 3-tier applications are easy to maintain and provide reusability and scalability.

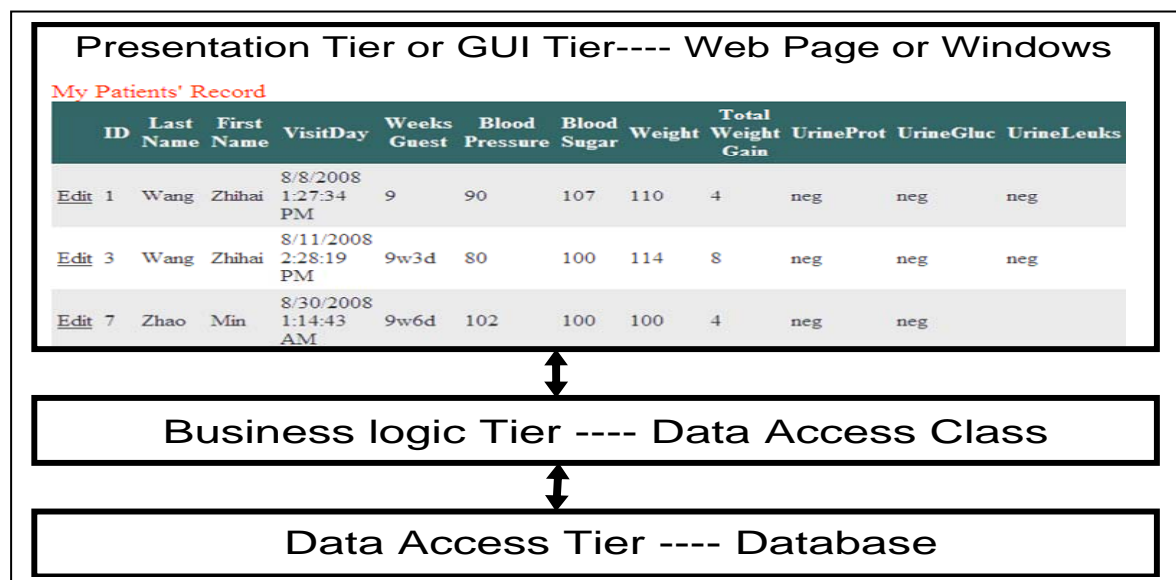


Figure 13. A typical three-Tier Architecture

6.2 Database Design of an e-healthcare prototype

In e-healthcare, all the medical information and patient information is critical and personal. The data must be stored safely and only accessed by authorized people. This information is stored in a secure database, the design of which is a very crucial part of system development. It is the first step before coding even begins.

A good database design must reduce data redundancy, make the queries simple, and provide an easy way to do maintenance. Figure 14 demonstrates our database design for the prototype e-healthcare system. In Figure 14, there are Users Table, Roles Table, UserRole Table, PatientVisitRecord Table, PhysicianPatient Table, PhysicianDelegation

Table and SecureQuestion Table. Every user has a unique UserID and UserName in the Users Table. Every role has a unique RoleID and RoleName in the Roles Table.

The UserRole Table manages the relationship between a role and a user. In this system, a role may include many users. However, one user, with a single UserName can only be assigned to one role. If the user wants to have two different roles, he/she should have two different Username's.

Every medical record includes the user's PatUserID which stands for patient userID in PatientVisitRecord. A patient or a physician should be able to read the medical records based on the users' PatUserID. The PhysicianPatient Table controls the relationship between physicians and patients. A physician can access his/her patients' records. A patient can be assigned to multiple physicians. For this purpose we created the PhysicianPatient table. It provides an m-n relationship with a low degree of coupling among related components. Changes in the PhysicianPatient table will not affect other tables.

The PhysicianDelegation Table is in charge of the delegation of a temporary physician during the primary physician's absences. A physician's access can only be delegated to one other physician. It may only be delegated directly. If a physician already has delegation from another physician, he/she can not re-delegate to other physicians.

The SecureQuestion Table manages the secure questions which are set by each user. The UserID in this table is unique. Therefore, for each UserID, there is only one record which directs the latest secure questions set by the user. The purpose of SecureQuestion Table is to provide higher security during any password retrieval process.

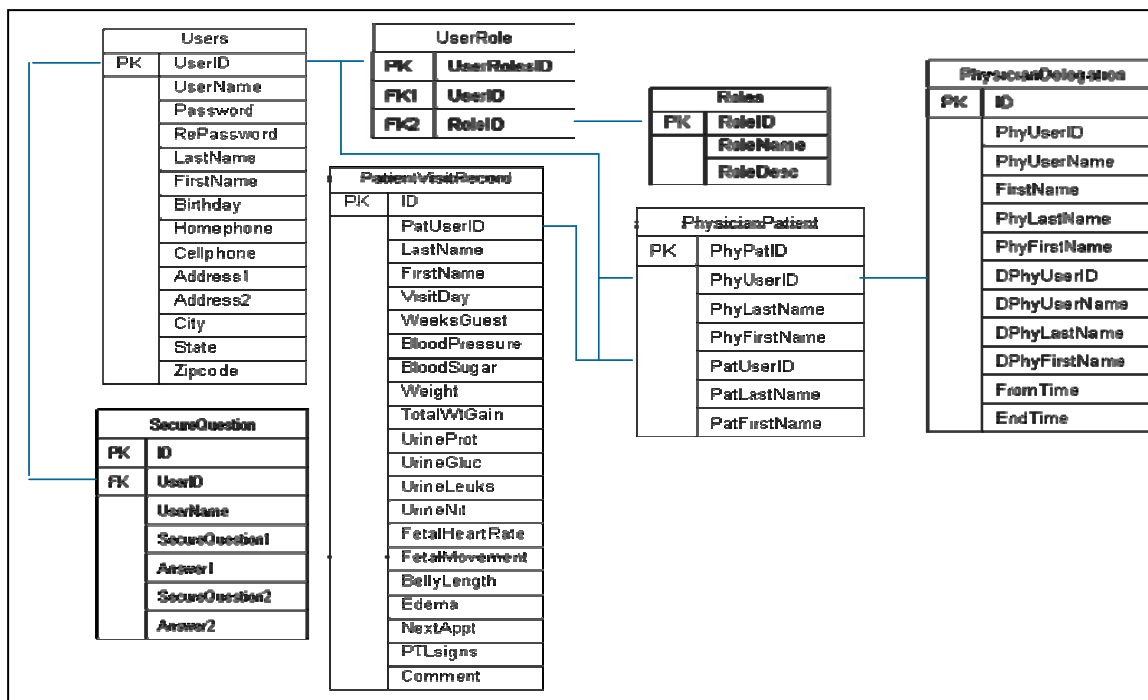


Figure 14. RBAC Database

6.3 Roles and their Applications

In this prototype, we employed 4 roles: system administrator, physician, patient and staff. Because of the different roles, the windows application tier contains several windows applications: administrator control panel, clinic windows application and physician windows application. This tier also contains two website applications: one for patients and another for physicians.



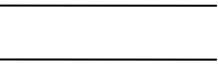
The overall system will check a user's role, username and password when the user tries to log in the system. There are two same functions for all users in their windows application: change password and set secure questions. All users can change their password and set their secure questions after they log in the prototype. If a user forgets password, the user can retrieve password quickly by answering the secure questions he/she set before. According to the role of the user, the system then selects the proper

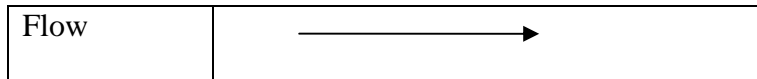
windows application if the username and password are correct. Each windows application provides different functionality. But there are two same functions for all users in their windows application: change password and set secure questions. All users can change their password and set their secure questions after they log in the prototype. If a user forgets password, the user can retrieve password quickly by answering the secure questions he/she set before.

In order to increase the security of the prototype, the windows applications and the web applications automatically log-off if there is no action on the website within a certain period of time (default 30 minutes). This time frame can be set and adjusted at any time by the administrator. This function prevents the leakage of private information of patients and the user's account information. For example, one user forgot to log-off the e-healthcare system and left the office. Other people can see his/her patients' information in the computer. The security of private information is very important in e-healthcare. The automatically logging-off well controls this scenario.

In this chapter, we deploy data-flow diagram to illustrate each role's functions which belong to the business logic of the prototype. Table 11 shows the notation of the Data Flow Diagram.

Table 11. Notations for Data Flow Diagram

Function	
Input/Output	
Database/File	



The functions which are represented by circles (processes) are implemented in the business logic tier. Input/Output operations, which are represented by square, belong in the GUI tier. Arrow indicates the data flow. The same color arrows represent the data-flow for an entire operation. Counting the number of the different color, we can know how many functions are provided in this application. Following the arrows, we can see how the data are transferred and passed when a function is carried out.

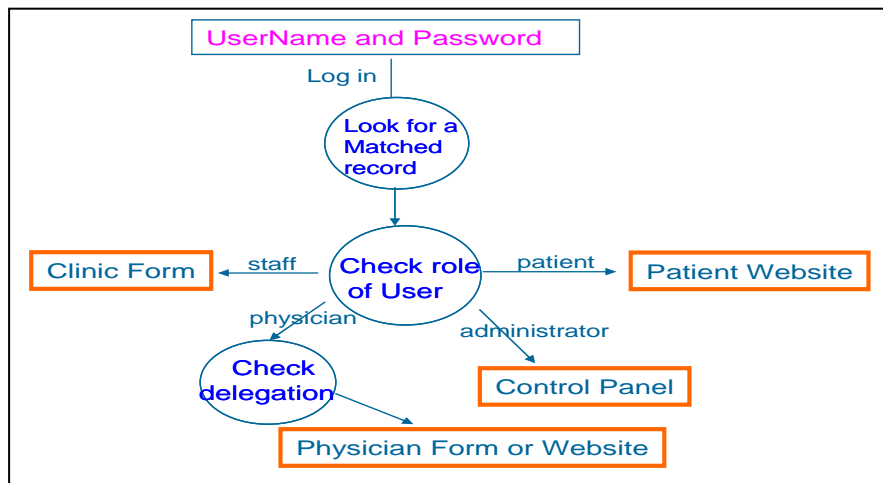


Figure 15. Data Flow Diagram for Log-in Function

Firstly, we will introduce the log-in function of the prototype. After a user inputs username and password, he/she invokes the log-in function, which queries the database to find a record to match the inputs. And if there is a matched record, then the prototype goes to check the user's role and invoke related windows form or web page. Figure 15 shows the data-flow diagram for log-in function.

There are two functions for all users: set secure password and change password. Figure 16 illustrates the data-flow diagram for these two functions.

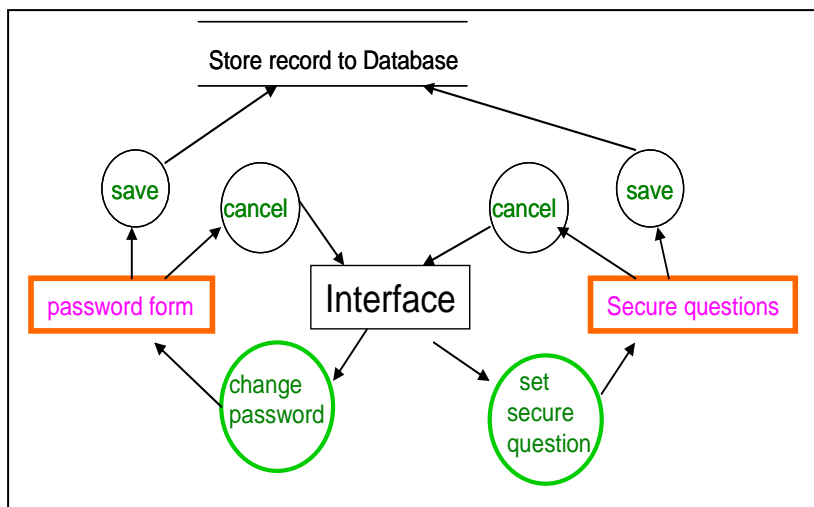


Figure 16. Data Flow Diagram for Change Password and Set Secure Questions

Changing password or setting secure questions can be operated by any user. In Figure 16, “interface” is a windows form or a web page which accessed by any user.

In the prototype, all “save” function will store the data into the database and the data shown in the GUI will be updated. All “cancel” function will go back to the last interface.

In data access tier, the SqlHelper class helps encapsulate high performance, scalable best practices for common uses of SqlClient package in Microsoft Visual Studio 2008. SqlHelper.ExecuteDataSet method execute a stored procedure via a SqlCommand to query a database and return a DataSet containing the resultset generated by the command. DataSet is an offline disconnected Data Store. DataSet is a collection of DataTables. DataRow represents a row of data in a DataTable. A logical business entity (object) retrieves value from DataRow. SqlHelper.ExecuteNonQuery method executes a stored procedure to insert, delete or update data in a database. Two parameters should be provided in these two methods: Sql command and connection string of a database.

6.3.1 Administrator Role and its windows applications

The administrator role is the critical role in e-healthcare prototype. An administrator has the highest right in an e-healthcare system and control the whole e-healthcare system. Figure 17 shows the use-case diagram for an administrator of the prototype.

6.3.1.1 Functions for Administrator Role

The administrator goes to the control panel after successful logging in. In control panel, there are 4 sections: User Management, Role Management, Delegation Management, and Assign Patient to Physician. Each section has many functions which belong to business logic layer. we will illustrate these functions by data-flow diagrams.

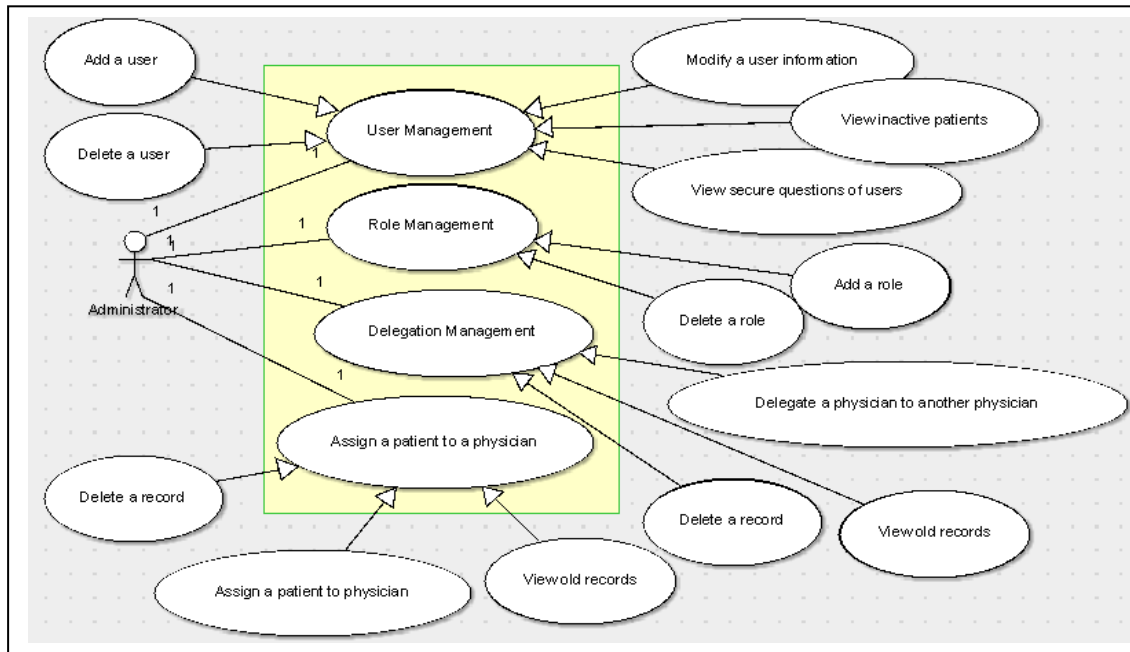


Figure 17. Use Case Diagram for Administrator Role

Figure 18 illustrates data-flow diagram for User Management in the administrator control panel. From figure 15, we know that there are five functions. An administrator

can add a new user, delete an old user, modify user information, view users' secure questions and view inactive patients. The administrator can add a new user by calling "add" function, filling an "A new user form" and doing a "save" action. And he can delete a user by calling the "delete" function. The "modify" function is a very important function. Because the administrator calls "modify" function to assign a user to a role. In this prototype, only the user who had been assigned to a role can log into the system. Also he can modify the user's information. The administrator has the right to view all inactive patients. In this function, he also can view inactive patient's personal information or medical records. And the administrator can view all users' secure questions. He sends the password to a user who forgot password and answered right secure questions.

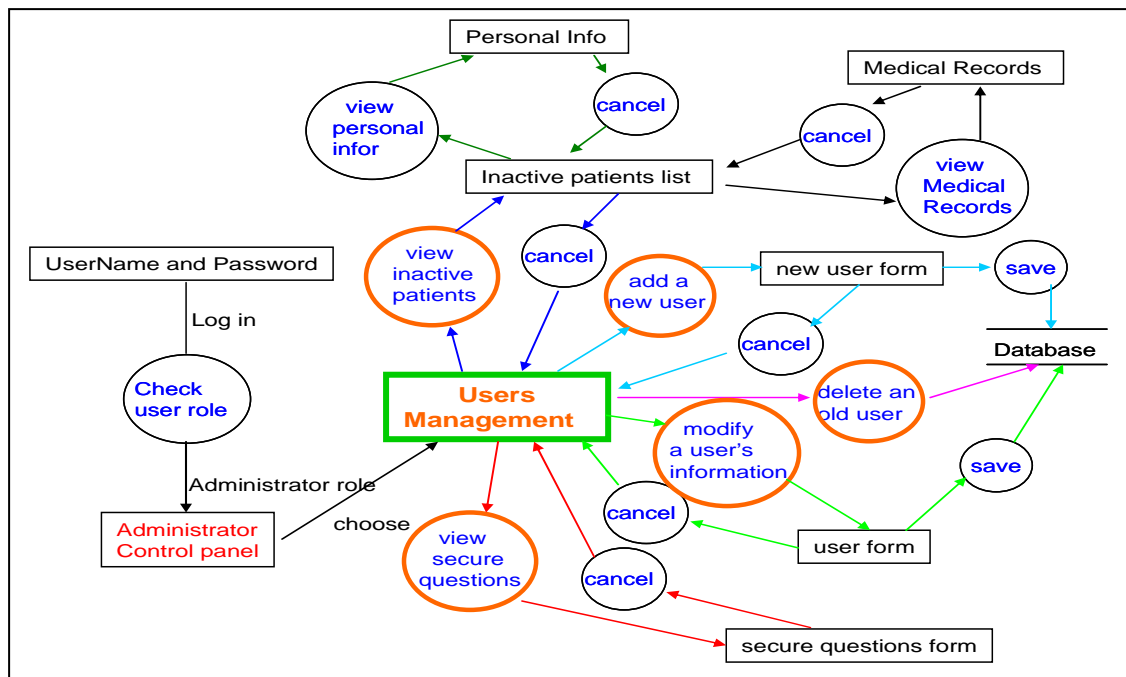


Figure 18. Data Flow Diagram for Users Management

Figure 19 illustrates the data-flow diagram for roles management of administrator control panel. There are two functions. The administrator can add a new role or delete an old role. Comparing to the number of users, the number of the role is much smaller. This

reduces the complication of the security authorization management of e-healthcare. Also, the role can be predefined without any influence on the system. This provides an efficient way to administer access control in the e-healthcare system.

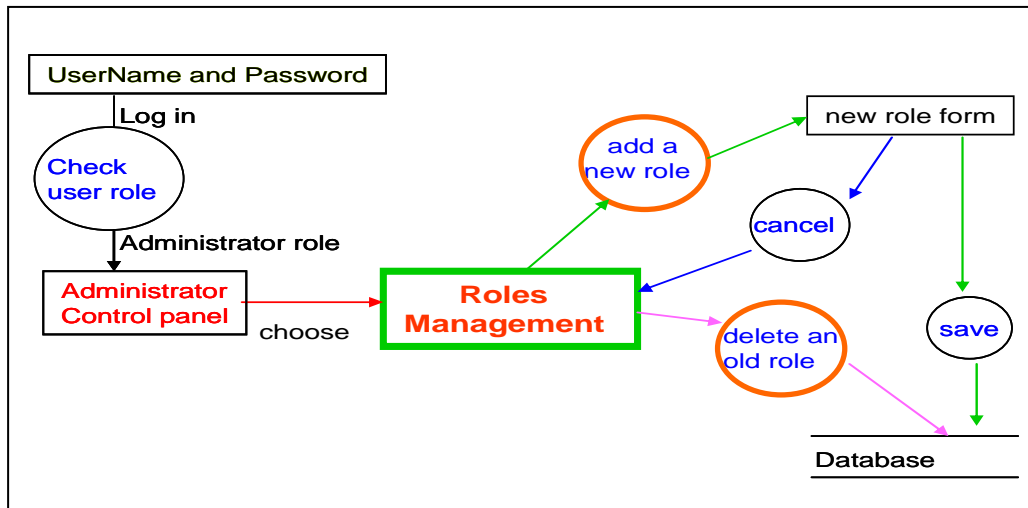


Figure 19. Data Flow Diagram for Roles Management

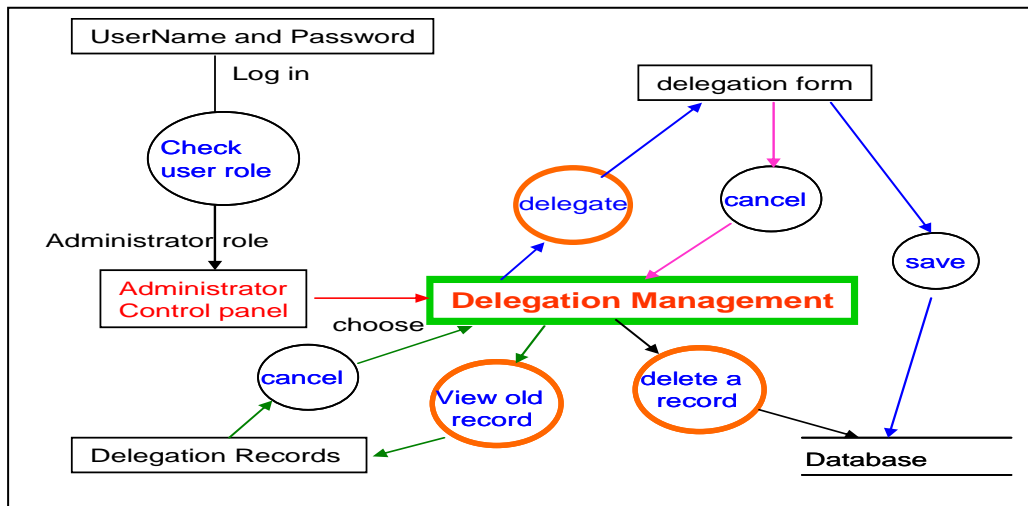


Figure 20. Data Flow Diagram for Delegation Management

Figure 20 illustrates the data-flow diagram for delegation within the administrator control panel. This section includes three functions. The foremost function is delegation function. In normal life, the physician may delegate their patients to other physicians during periods when they are unavailable. The delegation function can satisfy this

scenario. After the regular physician returns, the delegated physician will no longer be able to access the delegated patient's records. An administrator will do delegation for a physician if the administrator receives agreement from the physicians. In the agreement, there is a period showing the start time and the ending time for the delegation. After the ending time, the administrator can call the "delete" function which will remove the delegation record from the delegation list and move it to the backup database. This keeps full records for a physician's activity. Also, the administrator can "view old records" to view previous delegation record.

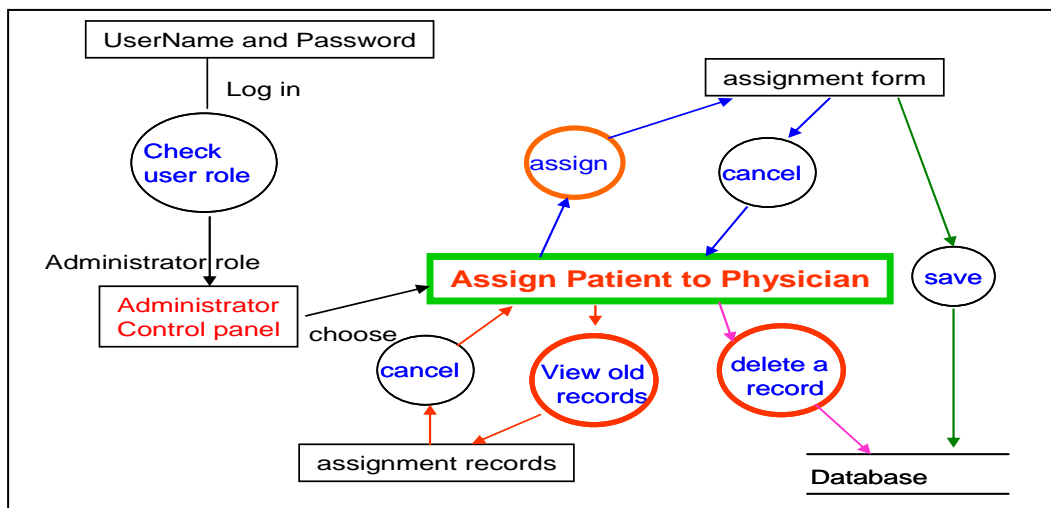


Figure 21. Data Flow Diagram for Assign Patient to Physician

Figure 21 illustrates the data-flow diagram for assigning a patient to a physician from the administrator control panel. After getting the agreement from a patient by, for example, signing some paper work when a patient come to a clinic at the first time, the administrator can assign the patient to the physician. The physician can see this patient in his/her patient list only after this assignment. If the patient changes to another physician, the administrator deletes the assignment record (which also will be moved to the backup

database) and adds a new assignment record to another physician for this patient. The administrator can “view old records” to get previous records for the patient.

6.3.1.2 Data Access for the Administrator Role

In the administrator control panel, there are four tables which separately display the users, roles, delegations of physicians and assignments of patient to physician. Since the codes of methods for retrieving data from a dataset are pretty similar, we will use one table as an example to illustrate it.

```

1 public void RetrieveUsers()
2 {
3     ArrayList mList = new ArrayList();
4
5     string sql = "select * from GetUsers ";
6     string conn = @"Data Source=MSEIP-GRAD\SQLEXPRESS;Initial Catalog=HealthCare;Integrated Security=True";
7     DataSet ds = SqlHelper.ExecuteDataset(conn, CommandType.Text, sql);
8     if (ds != null && ds.Tables.Count > 0)
9     {
10         DataTable dt = ds.Tables[0];
11         if (dt.Rows.Count > 0)
12         {
13             for (int i = 0; i < dt.Rows.Count; i++)
14             {
15                 DataRow dr = dt.Rows[i];
16                 Users mUsers = new Users();
17
18                 mUsers.UserID = dr["UserID"].ToString();
19                 mUsers.RoleName = dr["RoleName"].ToString();
20                 mUsers.UserName = dr["UserName"].ToString();
21                 mUsers.Password = dr["Password"].ToString();
22                 mUsers.LastName = dr["LastName"].ToString();
23                 mUsers.FirstName = dr["FirstName"].ToString();
24                 mList.Add(mUsers);
25             }
26             this.dataGridView1.AutoGenerateColumns = true;
27             this.dataGridView1.DataSource = mList;
28         }
29     }

```

Figure 22. Source Code for RetrieveUsers in User Management

In the User Management, the table shows users' personal information and their assigned roles. Figure 22 illustrates the source code for retrieving user data from the database and binding to a DataGridView control. Firstly, we create a DataSet by calling SqlHelper.ExecuteDataSet method. Secondly, the object (Users) retrieves values from DataRow which represents a row of a DataTable created for the DataSet and then is added to an ArrayList. Thirdly, the DataGridView control retrieves data from the ArrayList and provides a way to display data in a tabular format.

6.3.1.3 GUIs for the Administrator Role

We have introduced the administrator's functions by dataflow diagrams. In this section we briefly discuss the GUI panels used to access those functions. We will show several important interfaces for administrator role.

Role	User Name	Password	Last Name	First Name
Physician	lang1128@hotmail.com	134156	zhao	lang
Patient	silangjia@yahoo.com	123123	zhao	Lisa
Admin	wave1128@chinaren.com	123123	zhang	Lily
Physician	xzhang@magnet.fsu.edu	12345	zhang	xiaojun
Patient	langlang@yahoo.com	zl1234	zhang	zhao
Physician	xzhang@hotmail.com	123345	lang	lisa
Patient	silangjia@hotmail.com	12345	Zhang	Lang
Patient	xihuan@hotmail.com	zh123	Wang	Zhihai
Patient	lisazhang2007@hotmail.com	lz710	Lisa	Zhang
Physician	sindyzhao@hotmail.com	134134	Zhao	Shudan
Staff	xiaozhao@hotmail.com	123123	Zhao	Xiao
Patient	zhaozhao@hotmail.com	aa123	Zhao	Min
Staff	xzhang22@magnet.fsu.edu	111111	jian	xu
Staff	Merlina@hotmail.com	100100	Gui	Merlina
Patient	Fanfan@hotmail.com	111999	Fan	liang
Patient	yuanyuan@yahoo.com	190190	Fang	Yuan

Figure 23. GUI for User Management in Control Panel

Figure 23 shows the GUI tab for user management in control panel. There is a user list which includes the username, user's role, user's password, user's last name and user's first name. At the bottom, there are five function buttons: modify or assign role to user, add new, view secure questions, delete and view inactive patient.

Figure 24 shows the tab for role management in control panel. The diagram clearly shows role names and role description. There are two function buttons: add a new role and delete a role record.

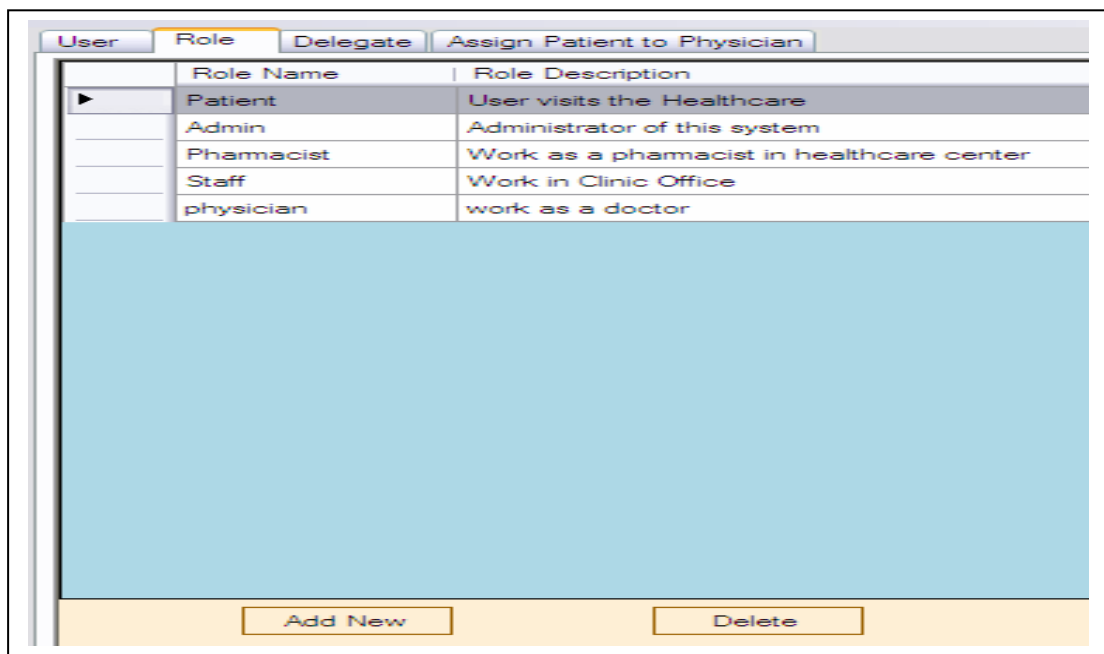


Figure 24. GUI for Role Management in Control Panel

Figure 25 shows the tab for delegation management in control panel. This screen shows the current delegations of physicians. You can clearly figure out a physician, related delegated physician and a time period. There are three function buttons. The “delegate” button is invoked to do the delegation. The “view old delegation records” button is invoked to view the old delegation records for physicians.

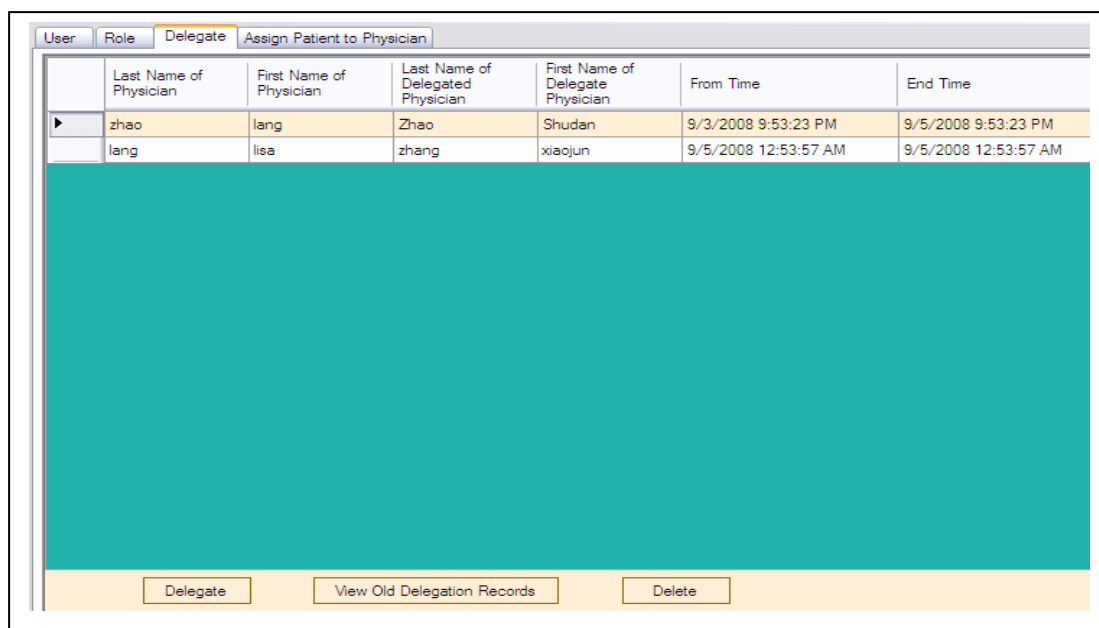


Figure 25. GUI for Delegation Management in Control Panel

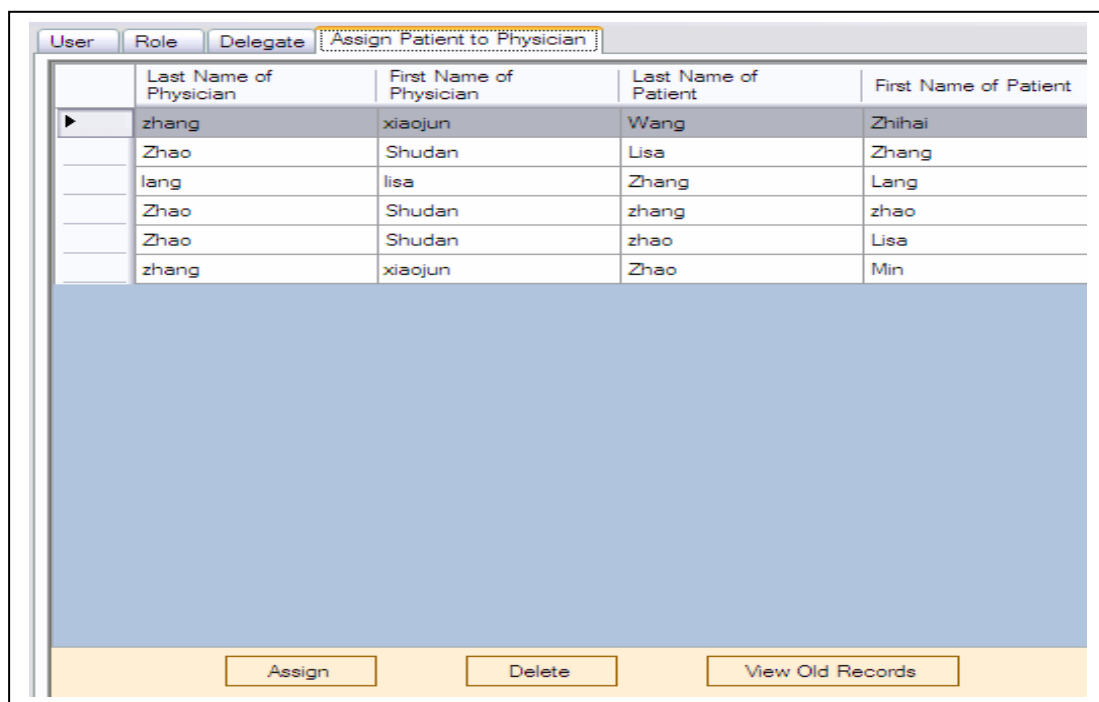


Figure 26. GUI for Assign Patient to Physician in Control Panel

Figure 26 shows a GUI tab for assignment of patient to physician in our control panel. The diagram shows records physicians and their patients. The “assign” button is

invoked to assign a patient to a physician. The “view old record” button is invoked to view a physician’s previous patients.

6.3.2 *The Physician Role and its windows Application*

The physician role is another important role in any e-healthcare system. As the name of e-healthcare shown, the objective of e-healthcare is to fulfill the operations of healthcare electronically by physicians. With the development of the information technology, the healthcare industry also enters the Information Age. Nowadays every doctor use computers instead of handwriting. Figure 27 shows the use-case diagram for the physician role. In this prototype, we implement two cases: Patient Management Form and Medical Record Form.

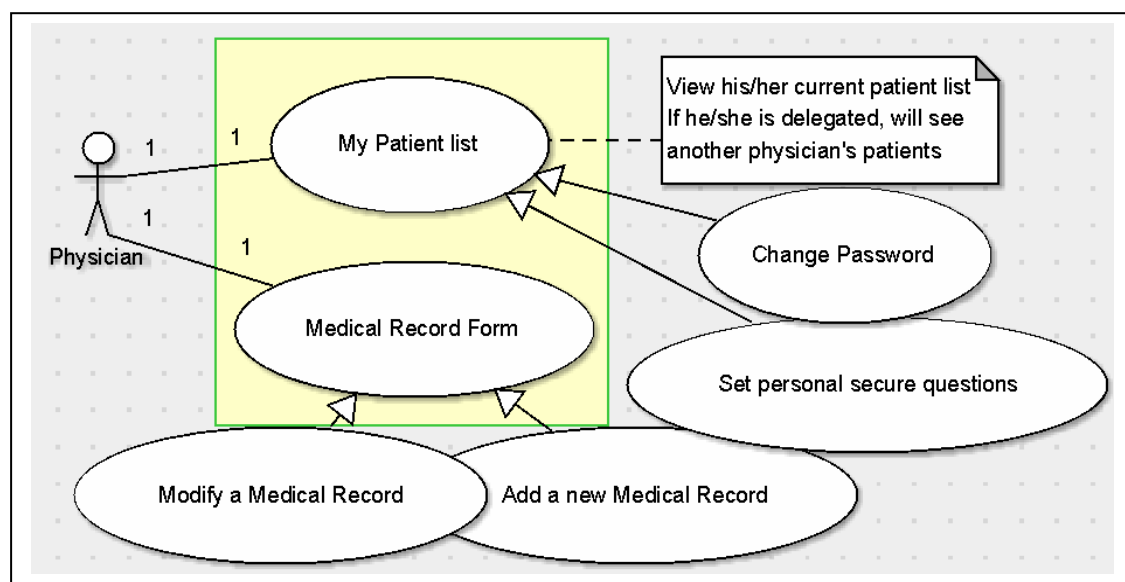


Figure 27. Use Case Diagram for Physician Role

6.3.2.1 **Functions for Physician Role**

Physicians have to record the medical records for every patient after treatment. Figure 28 illustrates the data-flow diagram for medical record form. There are two

important functions: adding a new medical record and modify a patient's medical record. When a physician tries to log in, the e-healthcare system will check the role and delegation state. If the physician successfully logs in, he will see a current-patients list which also includes the delegated patients. The physician chooses a patient in the list and then selects "View Record" function. The medical record form will be open. The physician may add a new medical record by calling the "add" function, fill the checked result and then "save" record. The physician also may choose one record from the medical record list and then picks the "Modify" function which opens a "medical record modification form". In this prototype, this is allowed only within two days after the patient's visit. Physicians cannot change the patients' medical record informally, which helps ensure the integrity of the information.

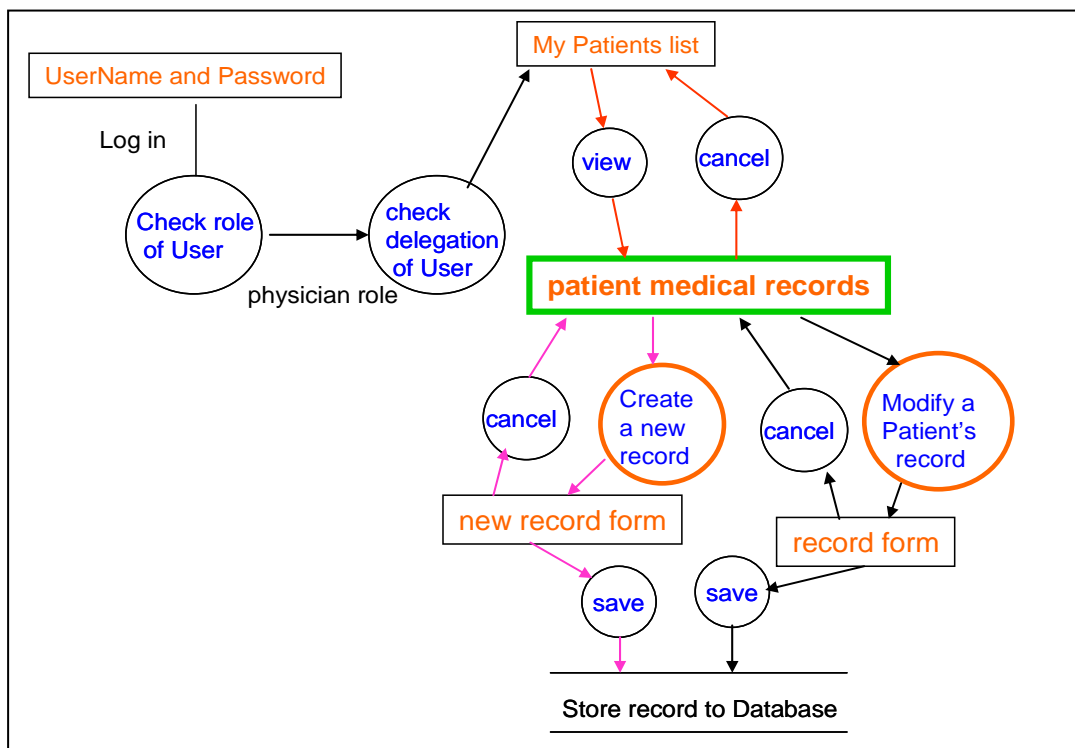


Figure 28. Data Flow Diagram for Medical Record Form

6.3.2.2 Data Access for Physician Role

There are two tables which separately display patient list of a physician and record list of a patient. Figure 29 shows source code for retrieving a Patient's Record for a physician.

```

1  public void RetrieveMyPatientRecord()
2  {
3      ArrayList mList = new ArrayList();
4      if (userid != 0)
5      {
6          string sql = "select * from PatientVisitRecord where PatUserID= ' " + userid + " ";
7          string conn = @"Data Source=MSEIP-GRAD\SQLEXPRESS;Initial Catalog=HealthCare;Integrated Security=True";
8          DataSet ds = SqlHelper.ExecuteDataset(conn, CommandType.Text, sql);
9          if (ds != null && ds.Tables.Count > 0)
10         {
11             DataTable dt = ds.Tables[0];
12             if (dt.Rows.Count > 0)
13             {
14                 for (int i = 0; i < dt.Rows.Count; i++)
15                 {
16                     DataRow dr = dt.Rows[i];
17                     VisitRecord mVisitRecord = new VisitRecord();
18                     mVisitRecord.ID = dr["ID"].ToString();
19                     mVisitRecord.PatUserID = dr["PatUserID"].ToString();
20                     mVisitRecord.VisitDay = dr["VisitDay"].ToString();
21                     mVisitRecord.LastName = dr["LastName"].ToString();
22                     mVisitRecord.FirstName = dr["FirstName"].ToString();
23                     mVisitRecord.WeeksGuest = dr["WeeksGuest"].ToString();
24                     mVisitRecord.BloodPressure = dr["BloodPressure"].ToString();
25                     mVisitRecord.BloodSugar = dr["BloodSugar"].ToString();
26                     mVisitRecord.Weight = dr["Weight"].ToString();
27                     mVisitRecord.TotalWtGain = dr["TotalWtGain"].ToString();
28                     mVisitRecord.UrineProt = dr["UrineProt"].ToString();
29                     mVisitRecord.UrineGluc = dr["UrineGluc"].ToString();
30                     mVisitRecord.UrineLeuks = dr["UrineLeuks"].ToString();
31                     mVisitRecord.UrineNit = dr["UrineNit"].ToString();
32                     mVisitRecord.FetalHeartRate = dr["FetalMovement"].ToString();
33                     mVisitRecord.BellyLength = dr["BellyLength"].ToString();
34                     mVisitRecord.Edema = dr["Edema"].ToString();
35                     mVisitRecord.NextAppt = dr["NextAppt"].ToString();
36                     mVisitRecord.PTLsigns = dr["PTLsigns"].ToString();
37                     mVisitRecord.Comment = dr["Comment"].ToString();
38                     mList.Add(mVisitRecord);
39                 }
40                 this.dataGridView1.AutoGenerateColumns = true;
41                 this.dataGridView1.DataSource = mList;
42             }
43         }
44     }
45 }

```

Figure 29. Source code for Retrieve a Patient Record

6.3.2.3 GUI for the Physician Role

Here we will show some important GUIs for the physician role. Figure 30 shows the GUI after a physician logs in successfully. The physician will see his patients in a list.

In this thesis, we implemented “write medical record” function. Figure 31 shows the GUI for all medical records of a patient. In the screen, the table shows all records for a patient. A physician can hit the “add new” button to add a new medical record or the “modify” button to change a medical record.

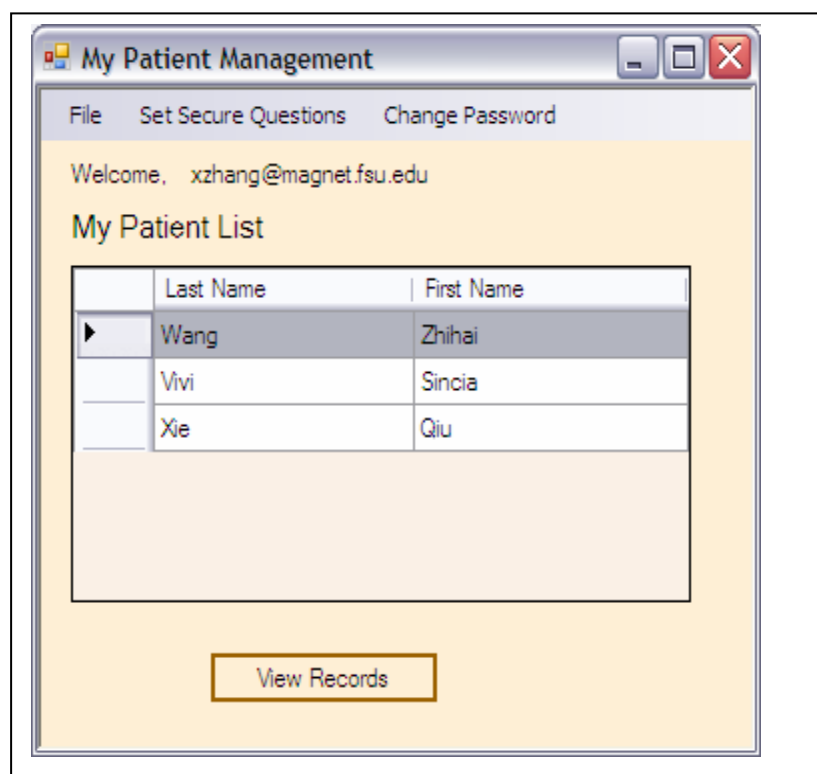


Figure 30. GUI for a Physician

Windows applications are normally used in the office. In this thesis we also created website applications for physicians who are out of office. Figure 32 shows the website after a physician logs in. It is convenient for physicians when they are out of office for any reason, because they can still pay close attention to his/her patients. It is

Figure 31. Medical Records for a Patient

Figure 32. Physician Website

6.3.3 The Clinic Staff Role and its windows Applications

The clinic staff role is an indispensable role in e-healthcare system. The office staff helps the physician arrange many things, such as collecting patient personal information, discharging a patient, preparing the copy of medical record for a patient, etc. Figure 33 illustrates the use-case diagram for clinic staff role. It has one main case: Clinic Management Form, which includes seven sub-cases: view patient medical records, view patient personal information, add a new patient, view inactive patients, discharge a patient, change password and set personal secure questions. In the “view inactive patient” case, there are two sub-cases: view inactive patient personal information and view inactive medical records. In “view patient personal information” case, there is one sub-case: modify patient personal information.

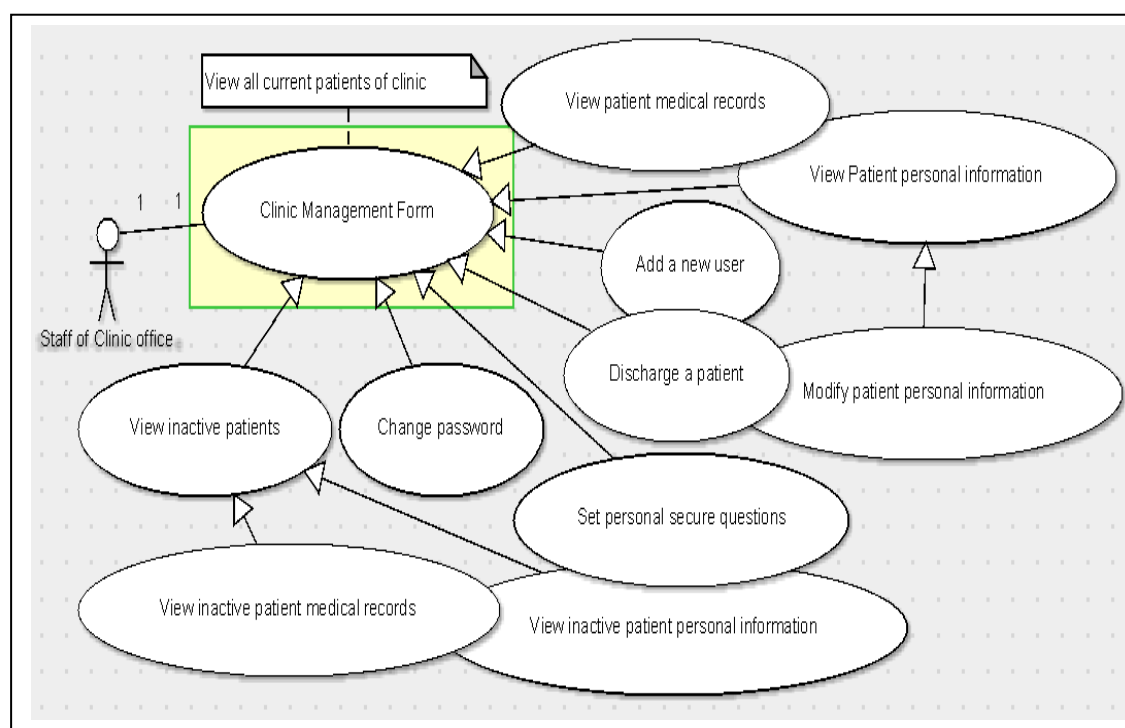


Figure 33. Use Case Diagram for Clinic Staff Role

6.3.3.1 Functions for Clinic Staff Role

After the log-in by office staff, he/she will see a list of current patients in this clinic. These current patients are called “active patients”. In this thesis, there are five functions for clinic staff.

Firstly, a clinic staff can view the patient’s medical record through calling the “View Patient’s Medical Record” function. But he/she can not modify the medical records. The data-flow diagram for this function is shown in Figure 34.

Secondly, a clinic staff can view a patient’s personal information by invoking the “view patient personal information” function. In the pop-up “patient information form”, the staff can modify a patient’s information. The data-flow diagram for this function is shown in Figure 34.

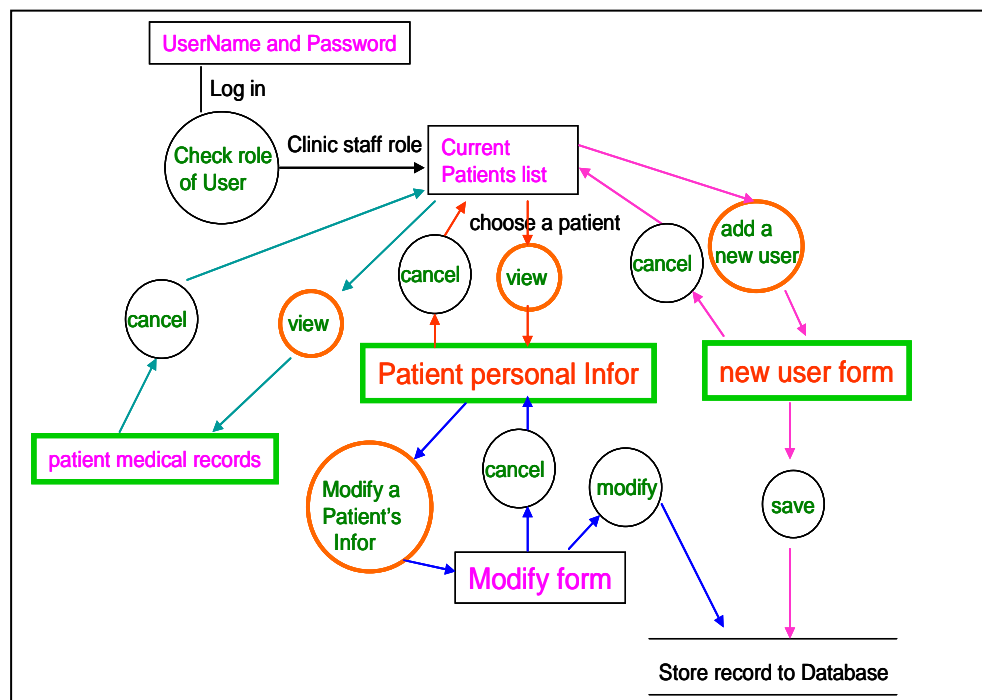


Figure 34. Data Flow Diagram for View Medical Record

Thirdly, a clinic staff can add a new patient user. Normally, the staff helps physicians accept patient, schedule appointment, etc. And it is convenient for a clinic to add a new patient user by the staff instead of the administrator. But only the administrator can assign “patient” role to the new user, which is in order to well control the accessing for the system. In another word, only the administrator has the right to administrate the access control. The staff can view the patient’s personal information by calling the “View Patient’s Personal Information” function.

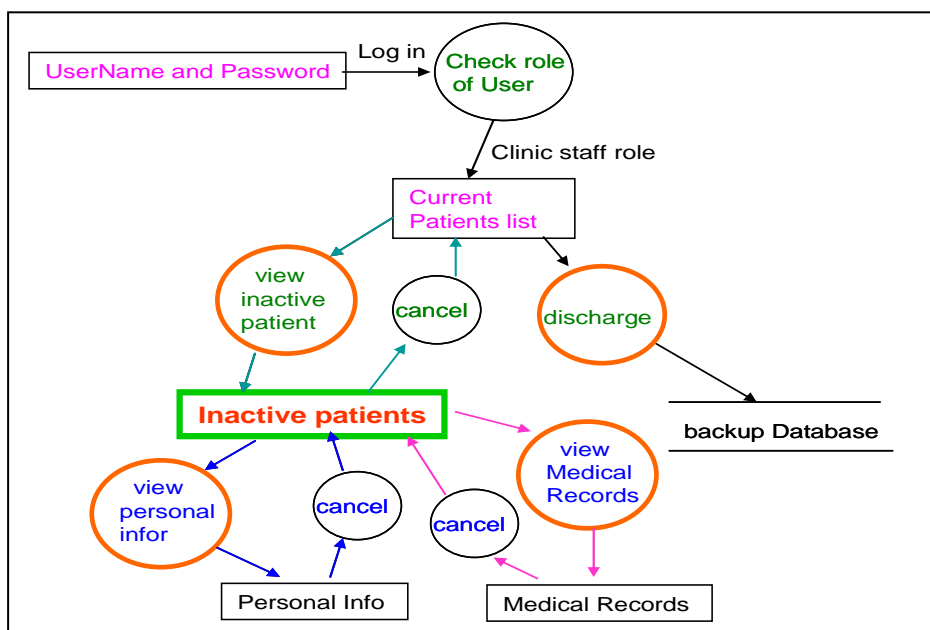


Figure 35. Data Flow Diagram for Discharge and View Inactive Patients

When the patient leaves care, the staff will help the physician discharge a patient. A discharged patient is called an ‘inactive’ patient. Discharge is the fourth function. The discharged patient will not be shown in the current patient list any longer and all his/her record will be moved into the backup database. Sometimes the inactive patient may need his/her old medical records from the clinic. The staff can get his/her personal information and medical records by calling the “view inactive patient” function. This is the fifth

function. The separation of the active patients and inactive patients makes it easy to administrate active patients and provide better service to current patients. Figure 35 illustrate these two functions.

6.3.3.2 GUIs for Clinic Staff Role

There is an important GUI for clinic staff: clinic office form. Figure 36 illustrates GUI for clinic office. A diagram displays all current patients of the clinic. Four functions we mentioned before can be achieved by hitting four buttons.

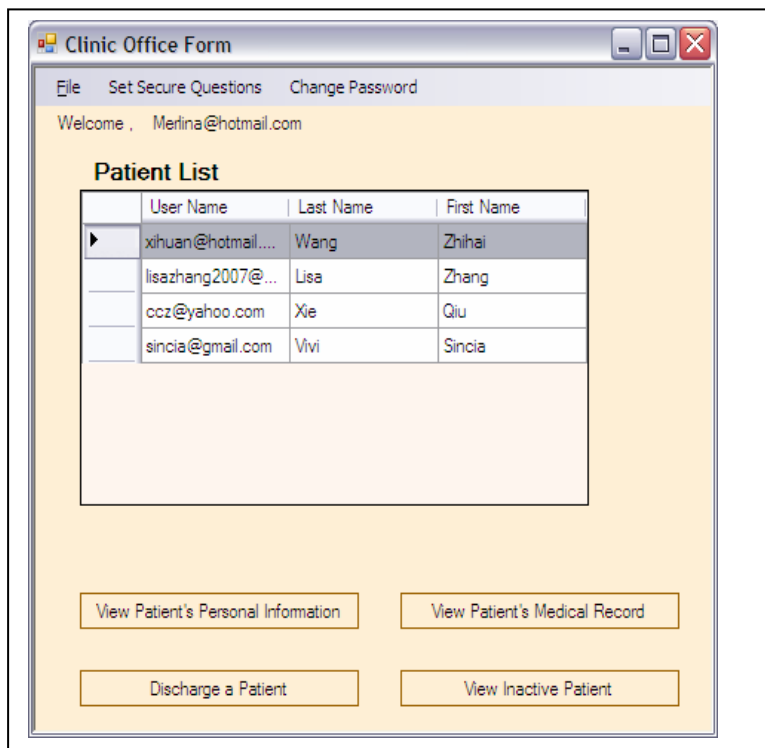


Figure 36. GUI for Clinic Office

6.3.3.3 Data access for Clinic Staff Role

There are two tables which separately display the clinic patient list and inactive patient list. Figure 37 illustrates the source code for retrieving patient list of a clinic.

```

1 private void RetrievePatient()
2 {
3     ArrayList mList = new ArrayList();
4     string sql = "select UserID, UserName, LastName, FirstName from GetUsers where RoleName = 'Patient'";
5     string conn = @"Data Source=MSEIF-GRAD\SQLEXPRESS;Initial Catalog=HealthCare;Integrated Security=True";
6     DataSet ds = SqlHelper.ExecuteDataset(conn, CommandType.Text, sql);
7     if (ds != null && ds.Tables.Count > 0)
8     {
9         DataTable dt = ds.Tables[0];
10
11         if (dt.Rows.Count > 0)
12         {
13             for (int i = 0; i < dt.Rows.Count; i++)
14             {
15                 DataRow dr = dt.Rows[i];
16
17                 User mUser = new User();
18                 mUser.UserID = dr["UserID"].ToString();
19                 mUser.UserName = dr["UserName"].ToString();
20                 mUser.LastName = dr["LastName"].ToString();
21                 mUser.FirstName = dr["FirstName"].ToString();
22                 mList.Add(mUser);
23             }
24
25             this.dataGridView1.AutoGenerateColumns = true;
26             this.dataGridView1.DataSource = mList;
27         }
28     }
29 }

```

Figure 37. Source code for retrieving patient list of a clinic

6.3.4 The Patient Role and its Web Application

Usually, a patient has to go to clinic to get paper records for his/her treatment. In this thesis website applications were created for patients who want to view their medical records online. Any patient can sign up on the Sign-up website. They can log in after getting the confirmation letters from the administrator.

6.3.4.1 Functions for Patient Role

There are two functions in the patient website: change password and modify personal information. Figure 38 illustrates these two functions by data flow diagram.

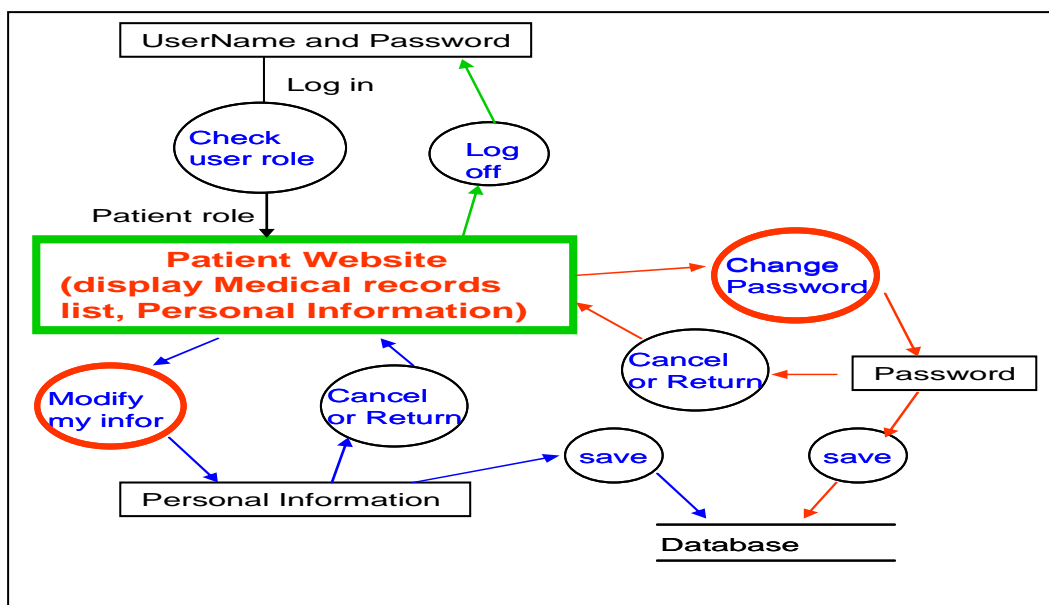


Figure 38. Data Flow Diagram for Patient Website

6.3.4.2 GUIs for Patient Role

Figure 39 shows website after a patient log in the system. The patient will see his/her medical records and his personal information. He/she can modify his personal information by clicking “edit” and change his/her password on line by hitting “Change my password” button.

Welcome, xihuan@hotmail.com												Change my password	Sign Out
My Medical Record													
Last Name	First Name	VisitDay	Weeks Guest	Blood Pressure	Blood Sugar	Weight	Total Weight Gain	UrineProt	UrineGluc	UrineLenks	UrineNit	Fetal Heart Rate	
Wang	Zhihai	8/8/2008 1:27:34 PM	9	90	107	110	4	neg	neg	neg	neg	146	
Wang	Zhihai	8/11/2008 2:28:19 PM	9w3d	80	100	114	8	neg	neg	neg	neg	173	
Wang	Zhihai	9/4/2008 10:04:18 PM	12w6d	100	97	118	12	neg	neg	neg	neg		
My Personal Information													
Last Name	First Name	Birthday	Home Phone	Cell Phone	Address 1	Address 2	City	State	Zip Code				
Edit	Wang	Zhihai	05/13/1975	850-575-1123	850-980-1678	121 Bliss Dr. Apt 11	Tallahassee	FL	32310				

Figure 39. Patient Website

6.3.4.3 Data Access for Patient Role

There are two tables which separately display the patient's personal information and patient's medical records. Figure 40 illustrates the source code for retrieving personal information and medical records.

```

1  this.Label1.Text = "Welcome, " + Session["UserName"];
2
3  string conn1 = @"Data Source=MSEIP-GRAD\SQLEXPRESS;Initial Catalog=HealthCare;Integrated Security=True";
4  string sql1 = "Select LastName,FirstName,VisitDay,WeeksGuest,BloodPressure,BloodSugar,Weight,TotalWtGain,UrineProt,UrineGluc,UrineLeuks,UrineNit,FetalHeartRate,FetalMovement,
5  DataSet ds1 = SqlHelper.ExecuteDataset(conn1, CommandType.Text, sql1);
6  // set the data source
7  this.GridView1.DataSource = ds1.Tables[0];
8  // bind the data source to the datagrid
9  this.GridView1.DataBind();
10
11 string conn2 = @"Data Source=MSEIP-GRAD\SQLEXPRESS;Initial Catalog=HealthCare;Integrated Security=True";
12 string sql2 = "select UserID,LastName, FirstName, Birthday, Homephone,Cellphone,Address1,Address2, City,State,Zipcode from Users where UserID = '" + Session["UserID"] + "'";
13 DataSet ds2 = SqlHelper.ExecuteDataset(conn2, CommandType.Text, sql2);
14 // set the data source
15 this.GridView2.DataSource = ds2.Tables[0];
16 // bind the data source to the datagrid
17 this.GridView2.DataBind();
18 // close the connection to the database since we are done using it

```

Figure 40. Source code for retrieving personal information and medical records

6.4 Authentication ----- Access Control Mechanism

The establishment of Access Control Mechanism begins with identification and authentication. Identification is an assertion of who someone is or what something is. Authentication is the act of verifying a claim of identity. There are three different types of information that can be used for authentication: something you know, something you

have, or something you are. In this prototype, the UserName is the form of identification. The password is the form of authentication: something you know.

Access to certain protected data must be restricted to people who are authorized to access the data. Authorization determines what informational resources are permitted to access and what actions are allowed to perform (read, create, update, delete, assign, or delegate) by a user who has successfully been identified and authenticated. Authorization to access data begins with administrative policies. The policies define what data can be accessed, by whom, and under what conditions. In this thesis, the authorization of a role is defined by the healthcare organization. Administrator who is in administrator role does assign a user to a role, assign a patient to a physician, and delegate a physician to another physician. A physician who is in physician role does view patients, create a medical record, view medical records, and modify a medical record within two day. Office staffs that are in staff role do add a new patient, modify a patient's personal information, view a patient's medical records, discharge a patient, and view a inactive patient's personal information and medical records. A patient who is in patient role does view own medical records and update personal information. Also, the policies define that the prototype should automatically log off if there is no action on the windows application within a certain period.

6.5 Summary

In this chapter, we illustrated how we implement the required roles for an e-healthcare system. For each we discussed the function, database design, and GUI's, which were either or both windows applications and web applications. In this role-based

e-healthcare system, a user can not log in until the administrator assigns a role to the user. A physician can view or write record for a patient after the administrator assigns the patient to him/her. A physician can delegate another physician to take his/her work when he/she goes out of office. All users can change their password and set their secure questions to get back forgotten password. The prototype will automatically log out or exit if there is no action on the windows application in 30 minutes. In addition, we built active/inactive database for past records of patients. The access of system is based on RBAC model and well controlled in our implementation.

Chapter 7 Conclusions and Future Work

Until now, we have made investigation and implementation a prototype of e-health system via workflow management system, security models, web services and eXtensible Access Control Markup Language. We also examined the eXtensible Access Control Markup Language (XACML) and RBAC XACML Profile. We implemented a small and simple e-healthcare system using the role-based security model suitable for our real-world case study. This system is developed in the Microsoft Visual Studio 2008 using C# language. The system includes two parts: windows application and web application.

The e-healthcare system can be logged in by authorized users. Users can log into different application according to their roles. In different applications, the user can do different operations according to the policy.

For better protection of patient privacy, a logged-in user will log out automatically if there is no action in the system within a set time (default 30 minutes). The time frame can be adjusted according to individual policy. This greatly increases the security of the system. When a user signs up, the user can set a few secure questions. If the user forgets the password, he/she can retrieve the password from the administrator by answering the secure questions correctly. Also, the system has a delegation function. If a physician goes out for business, he/she can delegate another physician to do his/her work.

This e-healthcare system is very straightforward, and is also an integral and complicated system. It controls the access of the system quite well. Because of the 3-tier

architecture of the system, it is easy to maintain and expand. In real life, there may be more roles that will be involved in the e-healthcare system. We plan to add roles to the system according to different business or policies, and add new functions and applications for special purpose. For example, we plan to add a function to encrypt the password to make the system more secured. We have completed a web service which provides a method to find users who are in the distributed healthcare organizations. We will add this web service to the current system, so that the users in the distributed healthcare organization can log in the same website to view their medical records. Also, we plan to finish the “Help” section for password retrieval to make the system more user-friendly.

A new design and implementation of e-healthcare is presented in this thesis. The advantage of this scheme is that the system is scalable and is easy to expand to mobile computing environment. Our implementation is still within a relatively small number of organizations. We will carry out a big, yet feasible, implementation that will provide the scenario required for a real e-healthcare system. More complex implementation, including insurance and billing information, need to be done for further validation of this system.

REFERENCES

- [1] E. Weippl, A. Holzinger, A. M. Tjoa, “Security aspects of ubiquitous computing in health care”, *e & i Elektrotechnik und Informationstechnik*, Volume 123, Number 4 / April, 2006, 156-161
- [2] Dickson K.W. Chiu, S.C. Cheung and Sven Till, Kamalakar Karlapalem, Qing Li Eleanna Kafeza, “Workflow View Driven Cross-Organizational Interoperability in a Web Service Environment”, *Information Technology and Management* 5, 2004, 221–250
- [3] F. Malamateniou, G. Vassilacopoulos, P. Tsanakas , “Workflow-based Approach to Virtual Patient Record Security”, *IEEE TRANSACTIONS ON INFORMATION TECHNOLOGY IN BIOMEDICINE*, VOL. 2, NO. 3, SEPTEMBER 1998,139-145
- [4] “What is Workflow?”, January 03, 2007, <http://www.e-workflow.org/>
- [5] “What is e-healthcare?”, <http://en.wikipedia.org/wiki/EHealth#Definitions>
- [6] Pierangela Samarati, Sabrina de Capitani di Vimercati, “Access Control: Policies, Models, and Mechanisms”, lecture Notes in Computer Science, Vol.2171, 2001, Pages 137
- [7] Edward A. Stohr, J. Leon Zhao, “Workflow Automation: Overview and Research Issues”, *Information Systems Frontiers* 3:3, 2001, Pages 281–296
- [8] Yet Another Workflow Language, <http://yawlfoundation.org/lexicon/index.php>
- [9] Rodney Gedda, Qld uni, “YAWLs in open source BPM”, 06/07/2006, <http://www.computerworld.com.au/index.php/id;396496377;fp;16;fpid;0;pf;1>
- [10] William Tolone, Gail-Joon Ahn, Tanusree Pai, Seng-Phil Hong, “Access Control in Collaborative Systems”, *ACM Computing Surveys*, Vol. 37, No. 1, March 2005, pp. 29–41.
- [11] Dr. Brewer, Dr. M. Nash, “The Chinese Wall Policy”, *Proc. In IEEE Symposium on Research in Security and Privacy*, May 1989, Oakland, California
- [12] Elisa Bertino, “Access Control Models”, CERIAS and CS &ECE Departments, Purdue University
- [13] Myong H. Kang, Joon S. Park, Judith N. Froscher, “Access Control Mechanisms for Inter-Organizational Workflow”, *SACMAT’01*, May 3-4, 2001, Chantilly, Virginia, USA pp 66-74. ACM 1-58113-350-2/01/0005.

- [14] John A. Miller, Mei Fan, Shengli Wu, Ismailcem B. Arpinar, Amit P. Sheth, Krys J. Kochut, "Security for the METEOR Workflow Management System", Large Scale Distributed Information Systems Lab (LSDIS), Department of Computer Science, the University of Georgia, <http://LSDIS.cs.uga.edu>
- [15] R. K. Thomas, R. S. Sandhu, "Task-based Authorization Controls (TBAC): A Family of Models for Active and Enterprise-oriented Authorization Management", *Proceedings of the IFIP WG11.3 Workshop on Database Security*, Lake Tahoe, California, August 11-13, 1997
- [16] Patrick Brézillon¹ and Ghita Kouadri Mostéfaoui, "Context-Based Security Policies: A New Modeling Approach", *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops (PERCOMW'04)*, IEEE, 2004, pages 154 *Conference, 2004. COMPSAC 2004. Proceedings of the 28th Annual International*, vol. 1, 2004, 72-77.
- [17] Rebecca Montanari, Alessandra Toninelli, Jeffrey M. Bradshaw, "Context-based Security Management for Multi-Agent Systems", *Multi-Agent Security and Survivability*, 2005 IEEE 2nd Symposium, 30-31 Aug. 2005, pages 75- 84
- [18] Lalana Kagal, Tim Finin and Anupam Joshi, "A Policy Language for a Pervasive Computing Environment", *Proceedings of the 4th International Workshop on Policies for Distributed Systems and Networks, (POLICY'03)*, 2003 IEEE
- [19] P. Filipe, M. Barata, N. Mamede, P. Araújo, "Enhancing A Pervasive Computing Environment with Lexical Knowledge", <http://www.inesc-id.pt/ficheiros/publicacoes/3161.pdf>
- [20] Vincent Rialle, Norbert Noury, Jacques Demongeot, "Smart Home: Information Technology for Patients at Home", *TELEMEDICINE JOURNAL AND e-HEALTH*, Volume 8, Number 4, 2002, pages 395-409
- [21] "HIPAA Security Series 1 Security 101 for Covered Entities"
<http://www.cms.hhs.gov/EducationMaterials/Downloads/Security101forCoveredEntities.pdf>
- [22] "HIPAA Security Series 4 Security Standards: administrative Safeguards"
<http://www.cms.hhs.gov/EducationMaterials/Downloads/SecurityStandardsAdministrativeSafeguards.pdf>
- [23] "HIPAA Security Series 4 Security Standards: Physical Safeguards"
<http://www.cms.hhs.gov/EducationMaterials/Downloads/SecurityStandardsPhysicalSafeguards.pdf>

- [24] “HIPAA Security Series 4 Security Standards: Technical Safeguards”
<http://www.cms.hhs.gov/EducationMaterials/Downloads/SecurityStandardsTechnicalSafeguards.pdf>
- [25] “What is HIPAA?”
http://en.wikipedia.org/wiki/Health_Insurance_Portability_and_Accountability_Act
- [26] Steffen Staab, “Web Services: Been There, Done That?”, *the IEEE Computer Society*, IEEE INTELLIGENT SYSTEMS, JANUARY/FEBRUARY 2003, p.72-85.
- [27] Vivek Devarajan, “Introduction to Web Services”, IBM SYSTEMS JOURNAL, VOL 41, NO 2, 2002
- [28] “Security in a Web Services World: A Proposed Architecture and Roadmap”, A joint security whitepaper from IBM Corporation and Microsoft Corporation. April 7, 2002, Version 1.0
- [29] WSS: SOAP Message Security (WS-Security 2004), 1 February 2006.
<http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>
- [30] Michael Hafner, Ruth Breu, “Realizing Model Driven Security for Inter-organizational Workflows with WS-CDL and UML2.0”, MoDELS 2005, LNCS 3713, pp.39-53, 2005.
- [31] “ws-securitypolicy-1.2”, Committee Draft 02, 7 March 2007,
<http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702/ws-securitypolicy-1.2-spec-cd-02.pdf>
- [32] “OASIS Committee Draft”, “WS-Trust 1.3”, September 2006,
<http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.doc>
- [33] Brian J. Garback, Alfred C. Weaver, “XACML for RBAC and CaDABRA: Constrained Delegation and Attribute-based Role Assignment” Computer Science Department, University of Virginia, Charlottesville, VA,
<http://www.cs.virginia.edu/~bjg5x/>
- [34] Tim Moses, “eXtensible Access Control Markup Language Version 2.0”, OASIS Standard, 1 Feb 2005, http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf

- [35] “Core and Hierarchical role based access control (RBAC) profile of XACML v2.0”, OASIS Standard, 1 February 2005, http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-rbac-profile1-spec-os.pdf
- [36] “Sun’s XACML Implementation”, <http://sunxacml.sourceforge.net/>
- [37] Xin Jin, “Applying Model Driven Architecture approach to Model Role Based Access Control System”, University of Ottawa, 2006
- [38] “What is 3-tier architecture?”, http://en.wikipedia.org/wiki/Multitier_architecture
- [39] “Figure of 3-Tier Application”, <http://microsoft.apress.com/index.php?id=50>
- [40] “What is Information assurance? ”, http://en.wikipedia.org/wiki/Information_Assurance
- [41] “ISO/IEC 27002:2005 Information technology – Security techniques – Code of Practice for Information Security Management”, <http://www.iso27001security.com/html/27002.html#ScopeOfISO17799>