Secure Health Information Sharing System (SHARE)

Min Wu

May 25, 2001

Secure Health Information Health System (SHARE)

by

Min Wu

Abstract

To date, sharing patient health information across multiple institutions while maintaining patient privacy remains a dilemma. I introduce a secure health information sharing system, simply referred to as SHARE, for generating multi-center health studies, capable of securely sharing patient information across multiple clinical institutions.

SHARE is a web-based computer system that automates most of the steps necessary to create a protected health information sharing system. It provides a secure database communication environment and enables users to manipulate multi-center health study through the Internet.

Thesis Supervisor: Peter Szolovits

Title: Professor of Computer Science and Engineering

Acknowledgements

This work would not have been possible without the advice and full support from Professor Peter Szolovits, my research advisor. I would like to thank Dr. Mojdeh Mohtashemi for her direction to SHARE and her encouragement and suggestions about my research work. I thank Lik Mui for his immense help in the SHARE system design and implementation. I also want to thank all members of MEDG for providing such wonderful research environment. I am grateful to Professor Martha Gray, my academic advisor, for her help and advice from early on. I thank family for their love and support. They are my hope, confidence and source of happiness. Finally, this research was supported (in part) by grant R01 LM06587 from the National Library of Medicine.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation and Problem Definition

A multi-center health study is a collaboration across multiple clinical and research institutions to share patient records as part of a comprehensive health study. Such collaboration can improve research quality by providing more patient data to be investigated. Moreover, innovative research topics and approaches could be raised with a larger amount of available data. However, since multiple institutions can share patient records, patient privacy protection is a main concern. Although researchers need patient data from different data sources (*e.g.* hospitals or medical labs), the patient's identity should not be disclosed to them.

Secure Health Information Sharing System (SHARE) is for multi-center health studies. We assume that when a study is approved and the SHARE tools are installed at each site from which data are to be collected and at

the central site used to hold study data, one person designated as the "study generator" will receive an authenticated certificate that will allow her to use a secure web site we have developed to create a patient data repository at the central site. The SHARE tools implement the authentication protocols to assure that only authorized researchers can access and manipulate the data, the encryption standards that create new, sharable identifiers from which the individual patient is de-identified, and the layered encryption scheme that allows some authorities from the central study site and the appropriate local source site to cooperate to re-identify a patient to researchers who can then collect follow-up data, when necessary and permitted by the study protocol.

SHARE is a web-based computer system that automates most of the steps necessary to create a secure information sharing system. It provides a secure data communication environment through the Internet. It enables studies that need to share patient information in a robust way that protects patient privacy. For each health study, SHARE implements the functionalities for

Study Creation: creating a central study database that stores study-related information;

Data Collection: collecting study-related patient information from different data sources to the central study database while hiding patient identity with encryption;

Patient Re-identification: re-identifying patient with decryption by authorities from the central study database back to the corresponding data source.

1.2 Organization of thesis

Chapter 2 introduces the background of patient privacy protection for multi-center health studies. Chapter 3 explains the system design in detail. Section 3.1 gives SHARE's glossary. Currently, we make simple assumptions about SHARE's policy issues. Section 3.2 states such assumptions. Section 3.3 discusses SHARE's goals. Section 3.4 analyzes how to guarantee SHARE's security. Section 3.5 gives two design choices to create the central study database. Chapter 4 elaborates how we implement SHARE. We use the Java language to build SHARE, Java servlet and Java database connectivity (JDBC) to support database-backed web sites and SSL to secure client-server and server-server communications. Section 4.1 is about multi-center study creation. Section 4.2 is about patient data collection. Section 4.3 is about patient re-identification. Section 4.4 introduces a two-server demo. Chapter 5 concludes the thesis with a brief summary, a discussion for the current system's security defects and some potential areas for future work.

Chapter 2

Background

The operation of any large health study requires some means of identifying the patients whose data are part of the study. For example, if data about the same individual are collected at different times, there must be a way to determine that these data are actually about the same individual and should properly be coordinated. Nevertheless, under most circumstances, those conducting the study have no need to know the actual identity of any particular patient in the study. Minimally, this means that patient data should not be identified by the patient's name, address, phone number, or other key that makes it very easy to go from the data back to the individual. Other possible identifiers, such as the Social Security Number (SSN), biometric measurements, medical record numbers, etc., make it relatively easy to determine the patient's identity but only in the presence of additional data, such as patient registries or SSN records. In the first three sections of this chapter, we review the use of SSN for patient identification, then briefly

discuss other unique patient identifier proposals, and touch on the problem of protecting privacy in widely-available records when vast amounts of data allow re-identification of data meant to be de-identified. We then describe the approach taken by this project and the overall effort of which it is a part.

2.1 SSN as patient identifier

To make sharing of patient records possible for a multi-center health study, each data item must be tagged with an identifier of each individual patient. A patient identifier is the index of a patient record. The Social Security Number (SSN) has been used to identify patients. Proponents have pointed out the cost-effectiveness and ease of adoption in current health institutions using this scheme. On the other hand, to the privacy advocates, SSN should not be used as a patient identifier [1] [2] because the use of the SSN increases the likelihood that medical information will be improperly disclosed to others and also invites many types of abuse of medical records.

2.2 ASTM's UPI study

American Standards for Testing and Materials (ASTM) has done an overall study about Unique Patient Identifier (UPI) options [3]. For example, Dr. Barry Hieb provided a sample Universal Healthcare Identifier (UHID), which consists of a sixteen (16) digit sequential identifier, a "." (period) that serves as a delimiter, a six (6) digit check-digit and a six (6) digit encryp-

tion scheme. Such UHID required a Central Trusted Authority to issue each patient a unique identifier. On the other hand, both Dr. Carpenter and Dr. Chute believed that the UPI should be based on immutable personal properties. They suggested a model consists of three universal immutable values plus a single check digit. The three values were a seven-digit date of birth field, a six-digit place of birth field and a five-digit sequence code. Although ASTM's study listed six UPI options, 3 non-UPI options and 5 alternatives to UPI, except the currently used SSN, all other options "require significant development since they do not already have all of the necessary operational characteristics, UPI components, administrative or technology infrastructure, implementation plan, policies and operating procedures" [3]. Moreover, by using UPI, there must be a nation-wide unanimous adoption of a particular judicious UPI design as well as the uniform federal and state legislation to prevent the UPI from misuse. Such a large-scale adoption is not an easy task, and has not occurred.

2.3 Datafly system

To protect personal privacy, Latanya Sweeney's Datafly system [4] uses computational disclosure techniques to maintain personal anonymity in publicly released data by automatically generalizing, substituting and removing entity-specific information as appropriate without losing many of the details found within the data. Each of its processed records can be made to map ambiguously to many possible people, providing a level of anonymity, while

still preserving its research value. However, such an approach can be thought of as a one-way function applied to each individual's data such that once the data is "scrubbed" of its identifiable attributes, tracing backward is close to impossible. Although the Datafly system prevents malicious patient re-identification, its unidirectional sharing scheme seems to be an obstacle to patient follow-up study and longitudinal care.

The Datafly approach also differs in goal from our study because it addresses the protection of patient privacy in data that are released for widespread public use, with no further legal or ethical control over that use. By contrast, researchers in a multi-center study have both formal and moral responsibility to protect patient privacy, and violations of these norms can lead to denial of access to the data. Therefore, the mechanisms used by SHARE are meant to reduce the risk of compromising patient privacy, but are not the only protections granted to the study data.

2.4 HIIDIT

Health Information Identification and De-Identification Toolkit (HIIDIT) [5] is a project to develop a set of tools that allow the creation of a broad range of patient identification systems, which would permit appropriate linking of multiple patient records but at the same time protect patient privacy. HIIDIT is not itself a patient identification system, but rather a generator of patient identification systems. HIIDIT gives the maximal freedom to the system designer to design appropriate system according to the tradeoff between

patient privacy and data's accessibility based on different social and security policies. SHARE can be treated as an implementation of one of the HIIDIT tools.

Chapter 3

System design

3.1 Glossary

SHARE enables multi-center health study by creating a central database that holds study data from multiple sources. We use some terms in the way that could be specific to SHARE:

GENERATION SITE: a secure web site where a generator can design a central study database and install it at an indicated study site server.

STUDY SITE: a secure web site in a study institution with one or more study databases, each of which independently collects patient data from its data sources and supports a multi-center health study.

SOURCE SITE: a secure web site in a clinical institution (*e.g.* hospital or medical lab) that agrees to take part in a multi-center health study

and is willing to provide study-related patient data to the central study database.

Each central study database is defined at a generation site, installed and operated at a study site and supplied with patient data by multiple source sites.

GENERATOR: a person who designs and installs a study database for a particular study purpose by using SHARE's generation site.

STUDY ADMINISTRATOR: a person who runs a study database at a study site. She collects data from multiple source sites.

STUDYIRB(study site institutional review board): a group of people at a study site who supervise whether the patient's privacy is compromised in a study database. They issue certificates to researchers who can then access the study database. In SHARE we treat studyIRB as a representative of the group.

STUDYOMB(study site ombudsman): a person at a study site who de-identifies patients in a study database. She is one of the two authorities involved in patient re-identification.

SOURCEIRB(source site institutional review board): a group of people at a source site who supervise whether the patient's privacy is compromised in the source site. They de-identify the patient data when the data are sent to a study database. They are the other authority

involved in patient re-identification. In SHARE we treat sourceIRB as a representative of the group.

RESEARCHER: a person at a study site who can access a study database with her certificate.

Each multi-center study has a central study database, which has a generator, a study administrator, a studyOMB, a studyIRB and a group of researchers.

SOURCE ID: a patient identifier that links the same patient information at a source site.

STUDY ID: a patient identifier that links the same patient information at a study site. Study ID hides the patient identity while allowing patient re-identification with studyOMB and sourceIRB's approval.

3.2 Assumptions

When we built the current SHARE, we made following assumptions to simplify (or even avoid) policy issues about sharing patient data for multi-center health study.

- SHARE allows generators to use a generation site to create a study database for a multi-center health study. SHARE allows researchers to use a study site to access a study database. However, both the generation site and the study site have the authentication protocols to

ensure that only authorized users can use SHARE. That is, generators and researchers need to have valid digital certificates before they enter SHARE sites. Under what condition the SHARE users (generator and researcher) can get such certificates is a policy question. Currently, we assume that these users already have certificates.

- Nowadays many clinical institutions refuse to share patient data with each other. Some of them worry about their patients' privacy; others are concerned about the research value of the data they own. Therefore, before institutions agree to share data, there should be some policy negotiations. For example, what types of the data are the source sites going to provide? What kinds of studies can use the shared data and what kinds of studies cannot? We assume that at the point when a study database collects data from its related source sites, the study site and the source sites agree to certain contracts to share data.

- The source sites in SHARE are the existing clinical institutions. It is reasonable to assume that each source site has its self-defined database structure to store patient data and its self-specified identification scheme to identify patients by their source-IDs. SHARE does not attempt to change the established data storage at the source sites. We also assume that the source sites only provide study-related patient data to the study site. SHARE only concerns how to securely collect these duly released patient data from source sites to a study site, how to securely store these data at a study site and how to securely provide these data

to researchers.

3.3 Goals

Based on the above assumptions, we clarify what SHARE should achieve:

3.3.1 Security

SHARE splits overall security into two pieces: patient privacy and system security.

Patient privacy

Patient privacy is a main concern of SHARE. For a multi-center study [5]:

1. Only data that are duly authorized for release from the source site are entered into the study site;
2. The study at a study site should operate without knowing the patient's identity;
3. It should be practically impossible for researchers to read the patient data in the study database without the correspondent studyIRB's approval;
4. It should be possible to reliably add new information obtained from the source sites to a patient's record in the study database without requiring that patient be identified to the study site;

5. If the studyOMB agrees to it, she will be able to decode the identity of the source site from which patient data came from, but not the source-ID for that patient. This will allow the source sites, with consent of sourceIRB, to identify the patient for more clinical questions. That is, it becomes possible, but only through collaboration between authorities enforcing privacy policy, to find the patient's identity in order to get more information.

System security

SHARE prevents any potential adversary from doing any operations. Only authenticated generators with valid certificates can enter the generation site. Only authenticated researchers with valid certificates can enter the study site. Username and password are additionally needed for the authenticated study administrator, studyIRB and studyOMB to log in to a study site and for sourceIRB to log in to a source site. Moreover, since the generation site, the study site and the source site can be accessed on line, all communications with these sites are encrypted and authenticated to prevent security attacks. The secure communication also guarantees the patient data's integrity, which is essential to get reliable and valuable study results. SHARE guarantees that a study site and its researchers get the correct patient data from the data sources.

3.3.2 Flexibility

In SHARE, the definition of a study site and a source site is very flexible. A hospital can be a source site to provide data for certain research; it may also act as a study site to get data from other source sites to do its own research. On the other hand, a clinical study institution can be a study site using the data from its related hospitals; it may also provide its study result to other health projects. Thus, it works as a source site. Although each multi-center study presents a tree structure (a study site and its related source sites), the whole SHARE system has an egalitarian "net" structure.

3.3.3 Usability

SHARE provides a user-friendly interface and automates most of the steps to facilitate the study database design at the generation site, the study database installation and operation at the study site and the patient re-identification process from the study site to the source sites. Moreover, SHARE sites can be accessed by any client machine by using a web browser through the Internet.

3.4 Security Design

3.4.1 Study ID scheme

As we mentioned earlier, patient de-identification at the study site is only part of the story. SHARE also provides patient re-identification functionality from the study site back to each source site in order to support follow-up study

or simply check the patient data's integrity. However, such re-identification is non-trivial and must be approved by some authorities. We use a multi-layer encrypted patient identifier to guarantee that patient information is communicated in a controlled manner.

In our notation, we denote a person's public key as $Person^{public}$ and the corresponding private key as $Person^{private}$. We denote encryption of a message using one of these keys as Key(message). We denote the hash function for a message as Hash(message). $Hash_{Base64}$ refers to the hash value encoded in the Base64 format for readability. SHARE assumes each patient already has a source-ID at the source site. Based on her source-ID, SHARE de-identifies a patient at the study site by creating her a study-id. That is[1]:

$$\begin{aligned} \text{study-ID} = \text{Hash}_{\text{Base64}}(\text{studyOMB}^{\text{public}} \\ (\text{sourceIRB}^{\text{public}}(\text{source-ID}), \text{source site name})) \end{aligned} \tag{3.1}$$

This scheme hides patient identity at the study site with encryption. On the other hand, since we use encryption, if the authorities (studyOMB and sourceIRB) agree to decrypt the study-ID, patient re-identification is possible: when an authenticated researcher wants to find more information about a patient from the source site, she will ask the studyOMB to use her

[1] A modified formula based on HIIDIT [5]

private key to decrypt the patient study-ID[2] and determine which was the source institution for that patient (from source site name). However, the studyOMB cannot determine what the source-ID is. To obtain the identity of the patient, the studyOMB would have to contact the source site and have the sourceIRB apply her private key to obtain the source-ID and find more information about the patient with that source-ID.

3.4.2 Patient privacy analysis

To figure out whether patient privacy is well protected at the study site, let us analyze the central study database's structure. Data stored in the study database can be divided into two parts: encrypted study ID and clear study-related data.

As we discussed earlier, encrypted study ID ensures the patient re-identification in a controlled manner. On the contrary, if we simple use a source ID combined with its source site name as a study ID, any researcher can contact the source site directly to find out patient identity, which increases the probability that patient privacy is improperly disclosed.

Since a study database only stores study-related information, although the patient data are in plaintext, the study database is de-identified. It

[2]SHARE uses the hash value to represent the study-ID for readability (30 byte-long hash value vs. 384 byte-long pre-hashed value). Since hash function is a one-way function, a hash table is needed for each study database. Thus, when a researcher presents the study-ID, the studyOMB can get the pre-hashed value from the hash table and decrypt that value.

does not have patient explicit identifiers[3] (*e.g.* name, email address, phone number) because they are of little research value. However, with enough patient data, patient identity in such a de-identified database can be disclosed by combining the study database with other publicly released information, such as federal census or voters list[4]. Nevertheless, de-identification of the study database makes it far more difficult and costly to look up details about a patient and therefore reduces the likelihood of accidental or non-malicious investigation.

To make patient data more secure, we restrict the study database's access. Each researcher should comply with certain agreements to use that database. Thus, for a multi-center study, patient data is protected in a "trusted environment". This differentiates our situation from the public release scenario.

In summary, patient privacy is well protected by using an encrypted study ID, de-identified patient data and database access control.

3.4.3 User authentication

We use SDSI certificates [6] to authenticate the generator at the generation site and the researcher, study administrator, studyOMB and studyIRB at the study site.[5]

[3]A set of attributes that can be used together to distinctly and reliably identify the individual.

[4]Latenya Sweeney has demonstrated it in detail in her work [4].

[5]Since source sites are already existed clinical institutions, they have their own authentication schemes. Current SHARE does not concern the source site authentication. We assume that the sourceIRB already has her authenticated username and password.

SDSI certificate

Simple Distributed Security Infrastructure (SDSI) describes a simple and flexible public key infrastructure. It makes extensive use of certificates that easily give names to public keys. Each principal (public key) is a certificate authority. It can create its own name space containing local names with which it can refer to other principals. The local names are arbitrary and flexible enough to fit into any organizations. SDSI certificate can also specify authorization given to public keys. Therefore, SDSI certificate comes in two categories: *name certificate* and *authorization certificate*. A name certificate binds a public key with a local name within a SDSI name space (mapping <name, key>); an authorization certificate passes empowerment to a public key (mapping <authorization, key>). Each SDSI certificate has a validity interval. SDSI defines a group as a set of principals with the same group name. The group's membership certificates are multiple name certificates with the same local name.

SDSI defines an ACL (Access Control List) mechanism that grants authorization to local names (mapping <name, authorization>). SDSI's ACL is held in the local memory and is issued by the owner of the computer or the computer itself to control access to its resources. SDSI also defines a timed CRL (Certificate Revocation List), which contains a list of revoked certificates and a validity interval[6]. CRL's availability is a key factor to form the reliable ACL, since a security hole can be formed when an adversary

[6]The validity interval makes CRL short and handleable.

simply prevents the CRL access from an ACL and keeps using the revoked certificate.

Generation site

The generation site creates its name space and issues the generator's SDSI certificates. A valid generator gets a pair of certificates, including a name certificate and an authorization certificate. The name certificate maps the generator's public key to a name "generator"; the authorization certificate maps the same public key to an authorization "create_study".[7] Each name certificate is a membership certificate for the "generator" group. The generation site defines its ACL to restrict the access to the server. By using a group name we make the generation site's ACL very simple. It only has one static entry <"generator", "create_ study">. With a local CRL, the ACL does not need to change at all. The generation site adds valid generators by issuing them a pair of certificates; it revokes any compromised certificates by adding them to the local CRL. Since the ACL and its CRL exist in one server or in a local network, the CRL is highly available. On the contrary, if we put all of the generators' local names into the ACL, every time when we add a generator, we need to update the ACL. If we assume the chance of certificates being compromised is low, updating the ACL is much more expensive than updating the CRL. Since generator is not a fixed set of individuals, we refer

[7] The authorization certificate is necessary to indicate that the generator's privilege is to use the generation site to create a study. A generator can do no other jobs (*e.g.* system maintenance and update).

to them by a group name to simplify the authentication procedure.

Study site

The study site authenticates researchers in the same way as the generation site authenticates generators. A researcher's authorization is to access a central study database. Therefore, the name certificate maps a researcher's public key to the group name "researcher"; the authorization certificate maps the same public key to a study database name. A researcher's SDSI certificate pair is issued by the studyIRB who supervises patient privacy in the database that the researcher wants to access.

The study site authenticates study administrator, studyOMB and study-IRB differently from study researchers. It is reasonable to assume that for a multi-center study at the study site, its study administrator, studyOMB and studyIRB are some real officials in the study institution. They have the authority to run the study site and they can easily be mentioned by their real names. Therefore, the study site issues these officials SDSI name certificate, binding their public keys to their identity (*e.g.* real names). These privileged individuals create their username and password by using their name certificates to enter the study site.

3.4.4 Secure communication

We use X.509 certificate [7] to authenticate SHARE web sites (generation site, study site and source site) and use Secure Sockets Layer (SSL) to secure

all client-server and server-server communications.

SSL is the most-widely deployed security protocol that provides secure communication over the Internet. SSL provides both server and client authentications based on SSL certificates. The most popular one is the X.509 certificate. An X.509 certificate has a standard format and is usually issued by some widely trusted third-party certificate authorities (CA), *e.g.* Verisign. To run a secure web server, the most common way is to purchase the web site a X.509 certificate from a trusted CA. When a browser connects to the server through SSL, the server sends back its certificate. Since all popular browsers know the well-known CAs' public keys, the browser can check the server's certificate. If the certificate is valid, the browser knows the server's public key. It then uses this key to set up a secure channel with the server.

To operate SHARE among multiple servers in a secure manner, the simplest way is to get X.509 certificate for each server and set up SSL connections.

3.5 Study site database design choices

For a multi-center health study, in order to collect patient data from multiple source sites and store them into a study database, one fundamental question is how to decide which part of the data in each source site database is related to the study and should be loaded to the study database. Two schemes are therefore proposed:

Scheme one is generator-design-administrator-match: When a generator

wants to create a health study, she knows exactly what kind of data the study needs and what the study database structure is. After she logs in a generation site, she designs the study database in detail and creates that database at a study site. Then she finishes her job. When a study administrator logs in a study site for a new specified study, she will see an empty study database created by a generator. The study administrator will collect data from the related source sites[8]. For each source site, before the study administrator actually loads data from it, she gets the source site database metadata[9]. The administrator views the generator-defined database metadata at the study site and decides which part of data in the source site is relevant to study and she is going to load into the study database. That is, she manually maps the two databases' metadata. Only then can data be collected.

Scheme two is administrator-design-administrator-match: In scheme one, the generator knows the study thoroughly. On the other hand, the generator can only name a study topic and leave all design work to the study administrator. In this scheme, the administrator receives the study topic at the study site and designs a database. She still needs to map the metadata with source sites.

We implement SHARE using scheme one. Implementing scheme two is

[8]There definitely will be a policy negotiation procedure to find out which source sites are willing to join the study and provide data and therefore become the related source sites.

[9]Database metadata is data that describe the database itself. It includes the table names in the database, the column names in each table, the data type for each column, the primary keys, the foreign keys, and so forth.

not hard because we only need to move the design procedure from the generation site to the study site.

Chapter 4

Implementation

SHARE's implementation can be divided into three parts: study creation, data collection and patient re-identification. Figures 4.1, 4.2, 4.3 are high-level UML use case diagrams that illustrate the SHARE users and main functionalities.

In the study creation part (Figure 4.1), a generator uses a generation site server to design a multi-center study. Then the generation site server contacts the setup daemon at a study site server, which in turn creates a central study database.

In the data collection part (Figure 4.2), a study administrator uses a study site server to specify multiple source sites. The study site server then contacts the provider daemon at each source site to collect study-related data from the source site database to the central study database.

In the patient re-identification part (Figure 4.3), a researcher (requestor) and a studyOMB use a study site server to send a re-identification request,

while a sourceIRB uses a source site server to send the answer back. The reply daemon at a source site receives a request and the query daemon at a study site gets the answer and presents it to the requestor.

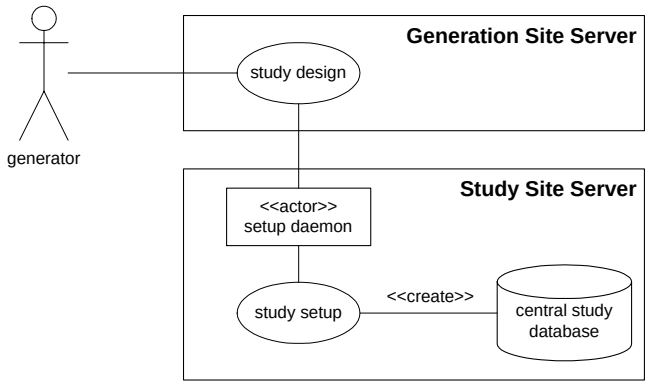

Figure 4.1: High-level UML Use Case Diagram for Study Creation

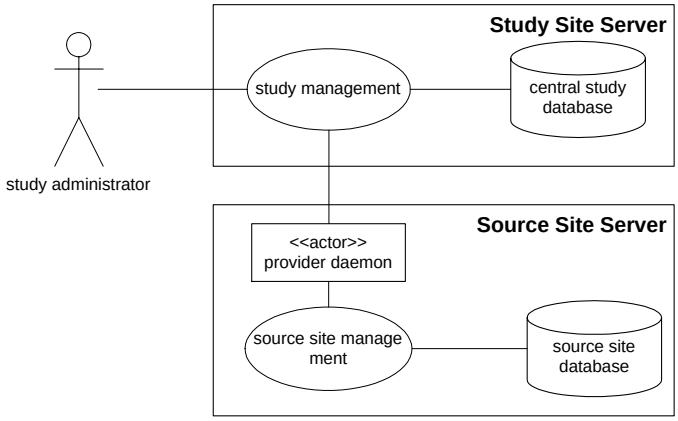

Figure 4.2: High-level UML Use Case Diagram for Data Collection

We use the Java language to implement SHARE. We use Java Servlet and Java Server Page (JSP) to implement server-side functionalities. We use

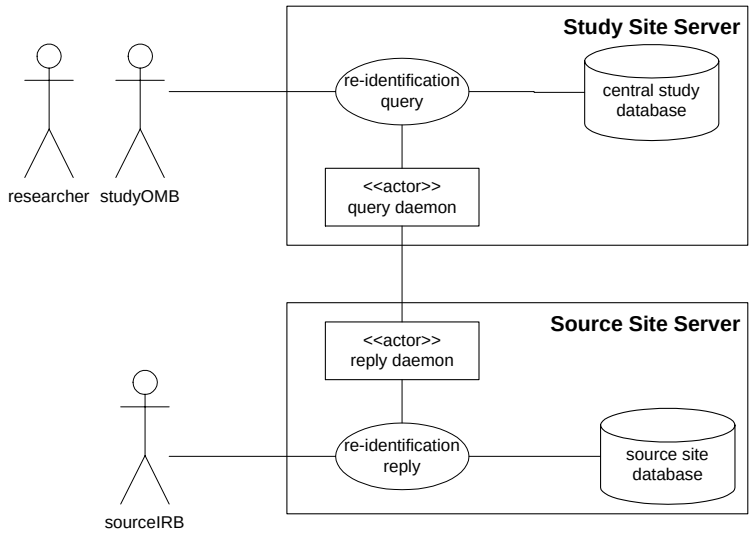

Figure 4.3: High-level UML Use Case Diagram for Patient Re-Identification

Java database connectivity (JDBC) to support database-backed web sites and to provide the on-line database manipulation.

SSL3.0 protocol along with Java Secure Socket Extension (JSSE) provides the transport level security for client-server and server-server communications. Extensible Makeup Language (XML) specifies the server-server communication standard.

SHARE depends on email to notify people, such as to tell a study administrator to load data for a newly-created study database, to tell studyOMB and sourceIRB to decrypt a study ID for patient re-identification and to tell a researcher to get the re-identification answer. We use the JavaMail API to implement SHARE's notification functionality.

The remainder of this chapter will elaborate each part's implementation.

For the sake of consistency, a common template consisting of the following categories is used:

1. implementation overview
2. detailed UML use case diagram
3. servlet collaboration
4. inter-server communication

4.1 Multi-center study creation

4.1.1 Implementation overview

A SHARE generation site server provides functionality for creating a multi-center study at a study site. The user (a study generator) logs in to a generation site server from any client machine through SSL with her SDSI certificate pair in order to design a study. After the generator is authenticated, she can choose to either review her previously designed study or design a new study on a study site server. A study design includes two parts: specification of the study profile and design of the study database structure. For the study profile, the generator provides the study site Uniform Resource Locator (URL), study topic (*e.g.* breast cancer), and the study administrator's information (*e.g.* the name and the email address). For the database structure design, the generator specifies the database structure using a web-

based user-friendly interface.[1] The generator can create new tables or modify or drop existing tables. As prompted by the generator, the generation site server contacts a remote study site server through SSL using the study site URL specified by the generator and sends it the study design information. When the study site server has received and parsed the message, it processes the study profile and translates the database design information into its native database language and constructs a new study database. If the creation succeeds, an acknowledgement is returned. The study site server also sends a notification email to the study administrator defined by the generator. If the creation fails because of some improper definitions, mostly the definitions of the study database design, error messages are returned. The generator can then revise the database structure on-line and send the re-design information again.

4.1.2 Detailed UML use case diagram

Figure 4.4 is the detailed use case diagram for multi-center study creation.

4.1.3 Servlet collaboration

Servlet for login

LoginServlet at a generation site server authenticates a generator with her SDSI certificate pair . During the servlet initialization (**init()**) a **Referee**

[1] This is a convenience for relatively inexperienced database designers. Nothing prevents the study generator from using traditional database design tools instead.

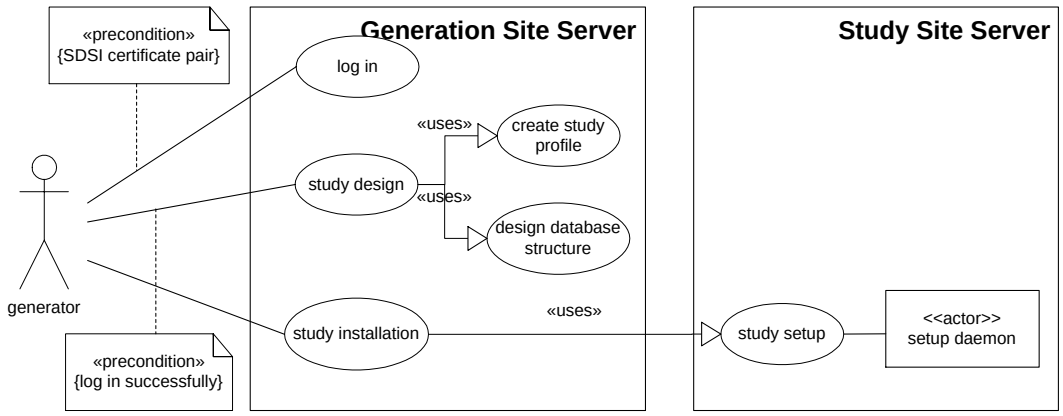

Figure 4.4: Detailed UML Use Case Diagram for Study Creation

object, **referee**, is created and it sets its **ACL** with a mapping <"generator", "create_ study">. Then the servlet waits for generator's login. When a generator connects to **LoginServlet** through SSL with an HTTP request, which includes the generator's username and a file containing her SDSI certificate pair (<"generator", public key> and <public key, "create_ study">), the servlet reads the certificate file and calls **referee.authorize()** to check

1. if the two certificates are signed correctly by the generation site (certificate issuer)

2. if the public keys in the two certificates are same

3. if the combined certificate pair (getting rid of the same public key) is the same as the ACL.

This function returns true if three checks are passed and then the authenticated generator can enter the generation site to design and create a

multi-center study. **LoginServlet** writes to a log file about every generator's login status, such as remote login URL, login timestamp, generator's username and login result (ok or fail).

Servlets for study design

At a generation site server, servlets in the **sites** and **design** packages are used for study design. The **sites** package deals with the study profile, while the **design** package deals with the study database structure. We use session tracking to share the study design information among servlets. **Javax.servlet.-http.HttpSession** class provides an elegant method for session tracking. When an authenticated generator enters a generation site, she creates an **HttpSession** object, **session**, which includes a **Generator** object (contains generator's information, such as username, public key hash[2] and email address), a **StudySite** object (contains study profile) and a **SHAREDBINFO** object (contains the study database structure). Each servlet in the **sites** and **design** packages can retrieve the study design information from a current **session**. When a generator finishes the design, she accesses **FinishSystemServlet** to save the design information into a *server database* at the generation site. (Each SHARE server has a database to store information about each multi-center study it involves. For example, the generation site server stores each study's generator username, her public key hash, study site server's URL and so on; the study site server stores the username and pass-

[2] **LoginServlet** gets the generator's public key hash from her SDSI certificate pair and stores it into current **session**.

word for each study's administrator, studyOMB and studyIRB, each study's database name, source site servers' URL and so on; the source site stores the information about where to send the patient data, that is the study site URL, study database name and so on. This database is different from the dynamically created study database and the source site database that holds patient data because it does not store patient information but the information to maintain a study. We call such a database the server database.)

Servlets for study installation

After a generator finishes a multi-center study design, she sends an HTTP request to the generation site's **OutputServlet** for the study installation. An **OutputRequest** object in **OutputServlet** gets the design information from the current **session**[3] or from the server database[4] using the generator's username and public key hash. Then it forms an XML message[5] containing the design information. **OutputServlet** opens a **java.net.URLConnection** to a study site server at the URL specified by the generator. By using JSSE, this connection is SSL-supported. This secure connection enables **OutputServlet** to contact the setup daemon at the study site server with an XML message. The setup daemon, that is **SystemInstallServlet**, parses the XML message, translates the information about the study database structure into SQL and creates a study database dynamically at the study site server through the

[3]If the generator designs and installs a study during the same HTTP session

[4]If the generator designs and installs a study through different HTTP sessions

[5]Detailed XML format will be given in the following inter-server communication part.

JDBC API. If the creation is successful, **SystemInstallServlet** sends an OK message back to **OutputServlet** at the generation site and sends the study administrator an email through the JavaMail API. If the creation fails, **SystemInstallServlet** returns the JDBC exceptions. **OutputServlet** at the generation site presents these exceptions to the generator to indicate what is wrong with her database design.

4.1.4 Inter-server communication

Inter-server communication is standardized by XML. A generation site sends to a study site an XML message containing study design information. To analyze this message, let us look at its Document Type Definition (DTD).

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE GENERATIONOUTPUT
[
<!ELEMENT GENERATIONOUTPUT (TOPIC, ADMINNAME, ADMINEMAIL, DBINFO)>
<!ELEMENT TOPIC (#PCDATA)>
<!ELEMENT ADMINNAME (#PCDATA)>
<!ELEMENT ADMINEMAIL (#PCDATA)>
<!ELEMENT DBINFO (TABLENO, TABLE+)>
<!ELEMENT TABLENO (#PCDATA)>
<!ELEMENT TABLE (TABLENAME, COLUMNNO, COLUMN+)>
<!ELEMENT TABLENAME (#PCDATA)>
<!ELEMENT COLUMNNO (#PCDATA)>
```

```
<!ELEMENT COLUMN (COLUMNNAME, MAXLENGTH)>
<!ELEMENT COLUMNNAME (#PCDATA)>
<!ELEMENT MAXLENGTH (#PCDATA)>
<!ATTLIST COLUMN ISPRIMARY (yes|no) #REQUIRED>
<!ATTLIST COLUMN ALLOWNULL (yes|no) #REQUIRED>
<!ATTLIST COLUMN ISTEXT (yes|no) #REQUIRED>
]>
```

TOPIC defines the multi-center study topic. ADMINNAME and ADMINEMAIL specify the study administrator's username and email address. DBINFO wraps the generator-designed database structure, which may contain one or more tables. TABLENO represents the total number of tables for each type of information. TABLE defines each table structure, which includes a table name, column structures and the total number of columns. COLUMN contains each column's information, that is, if the column represents the primary key for the table, if the column allows a null value, if the column data type is text and so on. MAXLENGTH indicates the maximal length (by character) for column data. Currently, we store everything at the study database in characters.

4.2 Patient data collection

4.2.1 Implementation overview

When the study administrator gets a notification email from the study site server after a study is successfully created, she can click the link contained in the email to quickly access the study site through SSL. The administrator indicates who will be the studyOMB and the studyIRB. She also specifies multiple source sites. She fetches the database metadata from each source site and views the generator-defined study database metadata. Based on these two databases' metadata she decides which part of the study-related data in each source site database she is going to load into the study database. She then sends a registration message to the source site. The source site server will store the study site's registration information, which includes the study topic, the study database name, the data query pattern and the studyOMB and studyIRB's name and email address. The source site will send an acknowledgement back at the end of registration. The study site administrator gets the acknowledgement message and finishes the source site's specification. Then she can collect data from the source site. Once a source site determines that the data loading is permitted, it de-identifies the patients with encryption and sends the duly released data to the study site. All the communications are secured by SSL.

4.2.2 Detailed UML use case diagram

Figure 4.5 is the detailed use case diagram for patient data collection.

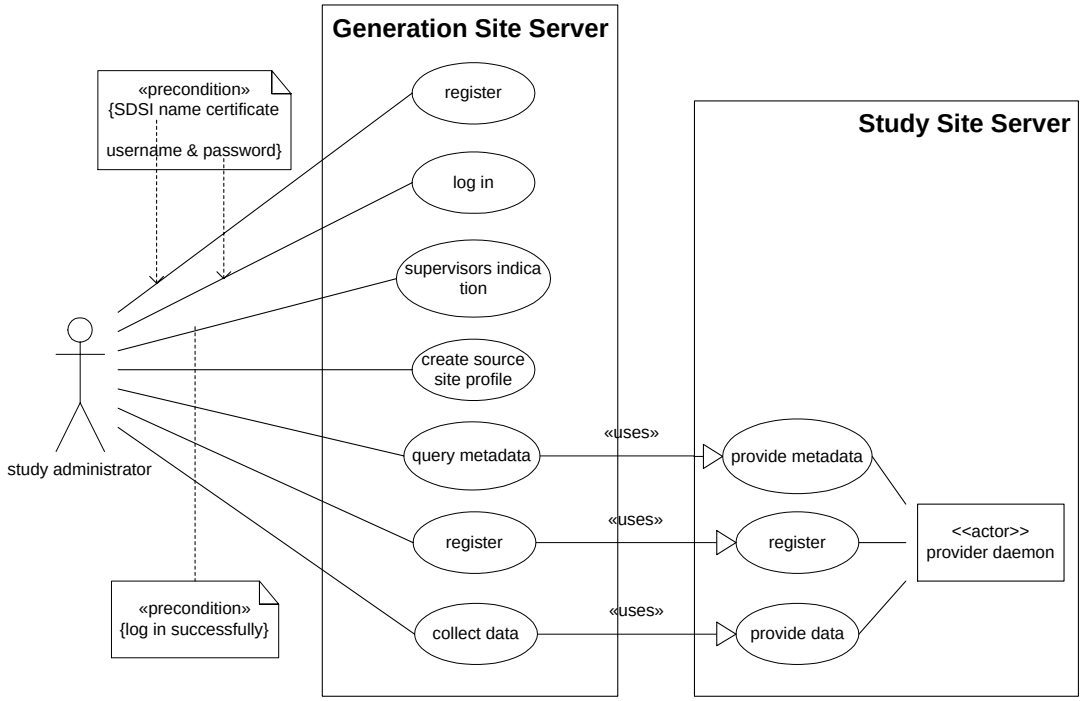

Figure 4.5: Detailed UML Use Case Diagram for Data Collection

4.2.3 Servlet collaboration

Servlet for register and login

RegisterServlet at the study site server authenticates the study administrator with her SDSI name certificate. It checks the certificate's validity (*e.g.* contained administrator's real name, valid signature from the study in-

stitution and the validity peroid) and allows the authenticated adminsitrator to create her username and password. **LoginServlet** checks the study administrator's username and password with the server database and allows the valid administrator to enter the site. It creates a **HttpSession** object with an **Administrator** object to store the valid study administrator's information, such as username. Based on the administrator's username in current **session**, **LoadStudyServlet** presents her the newly-created study database to which she is going to collect data.

Servlet for supervisor indication

Before an administrator collects data to a study, she indicates who will be the studyOMB and studyIRB. **LoadPrincipalServlet** receives the usernames and email addresses of studyOMB and studyIRB from the administrator and stores them into the server database for later contact.

Servlets for data collection

Three procedures are for data collection. Each procedure is performed by a pair of servlets, one at a study site and one at a source site, and their communications. Study site servlets initialize all three procedures and the corresponding inter-server communications.

Get metadata by GetSrcMetadataServlet and MetadataServlet :

After getting a source site server's URL from an administrator, **GetSrcMetadataServlet** at the study site opens an SSL-support **java.net.-**

URLConnection to the source site's **MetadataServlet** to get the metadata through the JDBC API of a particular database at the source site whose name matches the multi-center study topic. That is, if the study topic is breast cancer, **GetSrcMetadataServlet** asks for and **MetadataServlet** sends back the database metadata with the database named "breast cancer". When **GetSrcMetadataServlet** gets the answer, it presents both the source site database metadata and the generator-designed database metadata to the administrator using an HTML form so that she can map them on line and trigger the registration procedure.

Register study by TestSourceServlet and RegisterServlet :

TestSourceServlet at the study site sends an XML-formatted registration message to **RegisterServlet** at the source site through a SSL-support **java.net.URLConnection**. The registration message[6] tells the source site 1) who is the studyOMB (in order to use her public key to encrypt source-ID) and 2) which part of data in the source site database are study-related and needed by the study database. **RegisterServlet** parses the registration message and stores the registration information to the server database at the source site. The source site also saves the study site URL and study database name for data loading procedure.

Load data by QueryDataServlet and LoadDataServlet :

QueryDataServlet at the study site sends a loading request, includ-

[6]Detailed XML format will be given in the following inter-server communication part.

ing the study topic and the study database name, through SSL to **LoadDataServlet** at the source site. **LoadDataServlet** gets the remote study site URL, study topic and study database name and retrieves the corresponding data-loading pattern from the registration information. That pattern tells **LoadDataServlet** how to load study-related patient data. Patient source ID is encrypted by formula 3.1. **LoadDataServlet** forms an XML package containing patient data, encrypted IDs and their pre-hash values and sends it back to **QueryDataServlet**, which parses the package and saves the data to the study database through JDBC API.

4.2.4 Inter-server communication

XML message

XML-formatted registration message is used to register a multi-center study at a source site. Part of its DTD is:

```
<!ELEMENT TOPIC (#PCDATA)>
<!ELEMENT STUDYDBNAME (#PCDATA)>
<!ELEMENT STUDYOMBNAME (#PCDATA)>
<!ELEMENT STUDYOMBEMAIL (#PCDATA)>
<!ELEMENT QUERYTABLESINFO (TABLE+)>
<!ELEMENT TABLE (STUDYTABLENAME, SOURCETABLENAME, COLUMN+)>
<!ELEMENT STUDYTABLENAME (#PCDATA)>
<!ELEMENT SOURCETABLENAME (#PCDATA)>
```

```
<!ELEMENT COLUMN (#PCDATA)>
```

TOPIC defines the study topic. STUDYDBNAME indicates the study database name. STUDYOMBNAME and STUDYOMBEMAIL indicate the studyOMB whose public key will be used to encrypt patient source-ID. We assume that the studyOMB's public key is available. QUERYTABLESINFO specifies the metadata mapping. Each TABLE in QUERYTABLESINFO has STUDYTABLENAME and SOURCETABLENAME to map table names. COLUMNs in TABLE indicate that in each source table named as SOURCETABLENAME the data in which columns should be loaded to the study database.

Sequence diagram

Figure 4.6 displays the three types of study-source communications and their time-ordering. Since server-server communication is secured by SSL, the servlet engine has to authenticate every message from other servers and then passes the request to the corresponding servlet to handle. We will give the servlet engine configuration to support SSL in section 4.4.

4.3 Patient re-identification

4.3.1 Implementation overview

Since patients are de-identified with encryption at the study database, re-identification is possible only with studyOMB and sourceIRB's approvals.

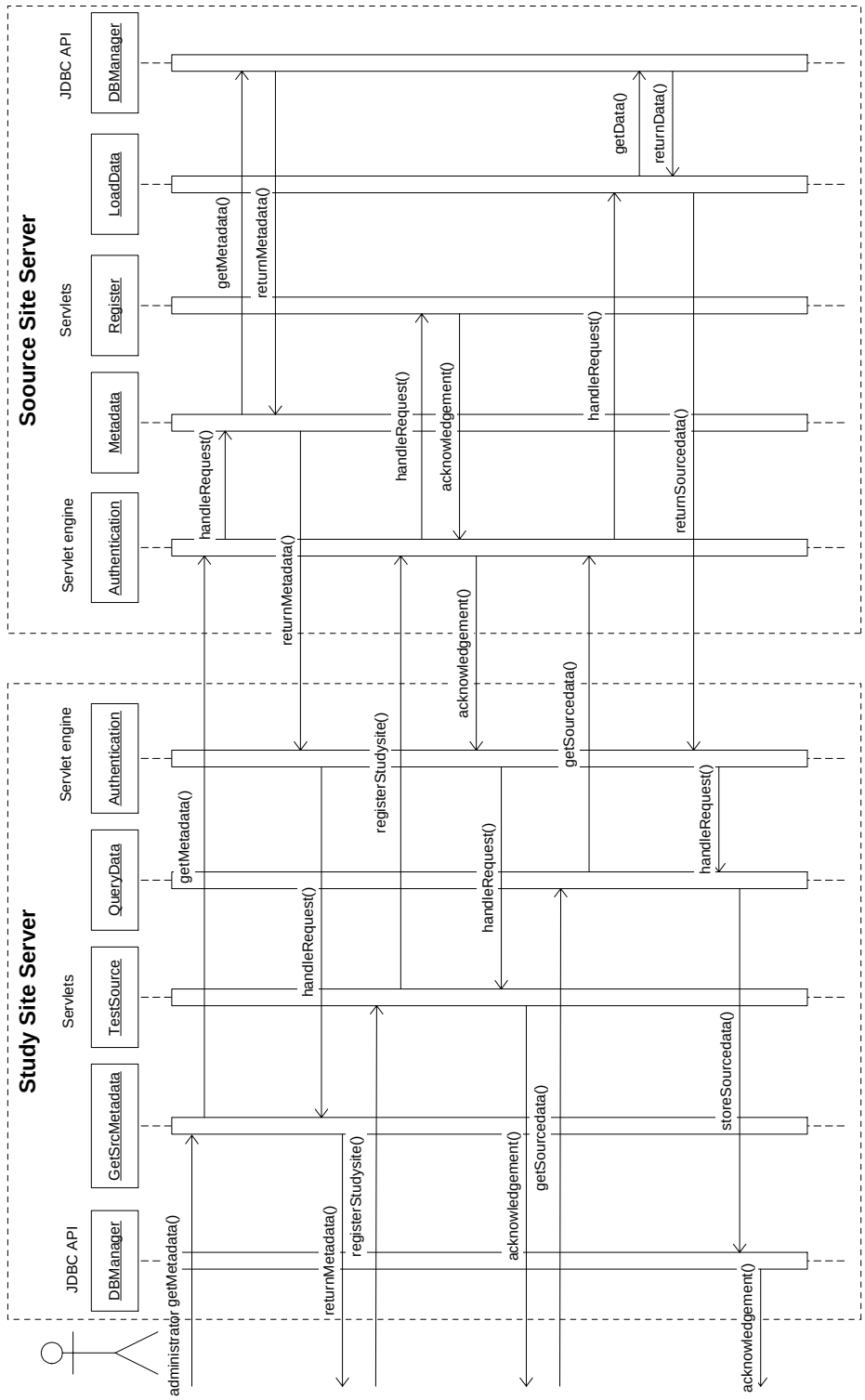

Figure 4.6: UML Sequence Diagram for Patient Data Collection

When a researcher wants to find more information about some patients based on a study database (let us denote this database as *SDB*), she logs in to the study site server using her SDSI certificate pair issued by the studyIRB who supervises *SDB*. She submits a request which contains several <study-ID, query field> entries. The researcher also provides the email address in order to be notified when the answer is ready. The study site stores the request to a request table, assigns it a unique tracing-ID and sends a notification email with the tracing-ID to the studyOMB who encrypts patient identity at *SDB*. The study OMB gets the email, clicks the quick-access link and finds the researcher's request using the tracing-ID. She may disapprove some unsuitable entries[7] and decrypt the other entries' study-ID to get the corresponding source site names and the encrypted source-ID. She sends a re-identification request (with the same tracing-ID), which contains the encrypted source-IDs and the corresponding query fields to the source site server according to the decrypted source site names. Each source site receives and stores the request to its request table and sends a notification email with the tracing-ID to the sourceIRB. The sourceIRB gets the email, accesses the source site server, decrypts the source-IDs, gets the data and returns the answers back to the study site server (SourceIRB can also disapprove some request entries and send the disapproval result back). The study site gets the answer, stores it to the request table and sends the researcher an email to notify her to get the answer. Then the researcher logs in with her SDSI certificate and gets the answer using the same tracing-ID. With encryption

[7]Entries that ask for sensitive information which can easily disclose patient's identity

patient re-identification occurs in a controlled way. A unique tracing-ID ensures the correct mapping between the request and the response data flow.

4.3.2 Detailed UML use case diagram

Figure 4.7 is the detailed use case diagram for patient re-identification.

4.3.3 Servlet collaboration

As discussed earlier, a researcher (requestor) initializes a patient re-identification. However, after re-identification is approved by the studyOMB and sourceIRB, the requestor can only get the information she wants but not know the patient identity.

We divide re-identification into the four procedures and will elaborate them one by one.

- A requestor posts a request at a study site.
- The studyOMB decrypts the study ID and sends a request with a encrypted source ID to the corresponding source site.
- The sourceIRB decrypts the source ID and replies with the requested patient information.
- The requestor gets the answer.

For clarity, we analyze the request with only one study ID. The mechanism is the same to deal with request with several study IDs.

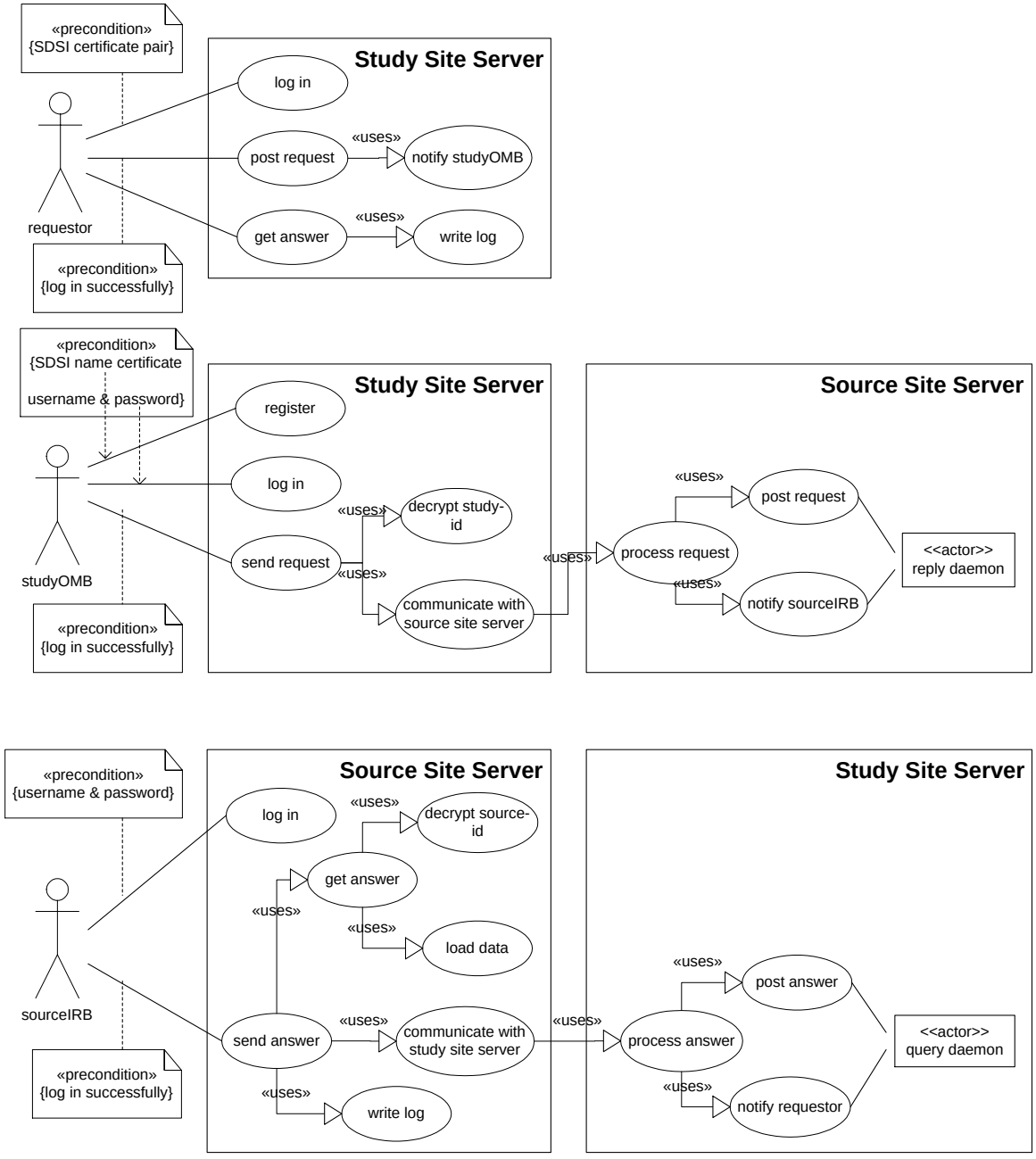

Figure 4.7: Detailed UML Use Case Diagram for Patient Re-Identification

Servlets to post request

A requestor logs in to the study site with her SDSI certificate pair. The authentication function is mostly the same as we mentioned for the generator's authentication in section 4.1.3. The requestor then accesses the web page where she can input her request through an HTML form and sends it to **PostqueryServlet**. This servlet stores the request into a request table, creates a random string to serve as a tracing ID for this request and sends an email through the JavaMail API with the tracing ID to the studyOMB to notify her that there is a request that she needs to take care of. **PostqueryServlet** also link the requestor's email address to the request[8] in order to inform her when the answer is returned. When the storage finishes, **PostqueryServlet** sends an acknowledgement to the requestor, who can then log off and wait for the answer.

Servlets to decrypt study ID

When the studyOMB gets a notification email from the study site, she logs in to the study site with her username and password[9] and the tracing ID. **LoginServlet** authenticates the studyIRB, finds the request using the tracing ID and presents her the request. If she approves it, she contacts **OmbprocessServlet** at the study site, which will retrieve the pre-hash

[8] A requestor needs to input his email address while logging in

[9] As the study administrator, the studyOMB also needs to register first to create her username and password.

study ID value from the study database's hash table, decrypt[10] it and get the encrypted source ID with the source site name. Then the servlet opens an SSL-supported **java.net.URLConnection**[11] to the source site's **QuerylistenerServlet** and sends a request with the same tracing ID. The request contains the encrypted source ID and the original query field. The source site **QuerylistenerServlet** gets the request, stores it into a request table and sends an email through the JavaMail API with the tracing ID to the sourceIRB to notify her that there is a request that she needs to take care of. After that **QuerylistenerServlet** sends an acknowledgement to **OmbprocessServlet**, which presents the acknowledgement to the studyOMB. At this time, the studyOMB finishes her job.

Servlets to send answer

When the sourceIRB gets a notification email from the source site, she logs in to the source site with her username and password and the tracing ID. **LoginServlet** authenticates the sourceIRB, finds the request using the tracing ID, gets the source ID by decryption[12] and presents her the request with clear source ID and query field. If the sourceIRB approves the request, she

[10] Currently the studyOMB's private key is stored in the study site server and can be retrieved only be its **OmbprocessServlet**. If we assume that the study site server is safe, it is a simple and reasonable strategy.

[11] **OmbprocessServlet** gets the source site URL from the server database.

[12] Currently the sourceIRB's private key is store in the source site server and can be retrieved only by its **LoginServlet**. If we assume that the source site server is safe, it is a simple and reasonable strategy.

indicates through an HTML form **IrbprocessServlet** where in the source site patient database it can find the answer. **IrbprocessServlet** then gets the answer through the JDBC API, forms an XML message with the same tracing ID, opens an SSL-supported **java.net.URLConnection** to the study site's **AnswerlistenerServlet**, which will parse the message, write the answer to the request table for a request with the same tracing ID and send an email to the requestor through the JavaMail API. After that **AnswerlistenerServlet** sends an acknowledgement to **IrbprocessServlet**, which writes the request to a log file, delete the request from the request table at the source site and presents the acknowledgement to the sourceIRB. At this time, the sourceIRB finishes her job.

Servlets to get answer

When the researcher gets an email from the study site, she logs in to the study site with her SDSI certificate pair and the tracing ID. Then she gets the answer from the request table. After that she contacts **FinishqueryServlet**, which writes the request to a log file and delete the request from the request table at the study site.

4.3.4 Inter-server communication

Figure 4.8 gives the context and overall organization of the interactions of client-server and server-server to re-identify a patient and retrieve the requested information.

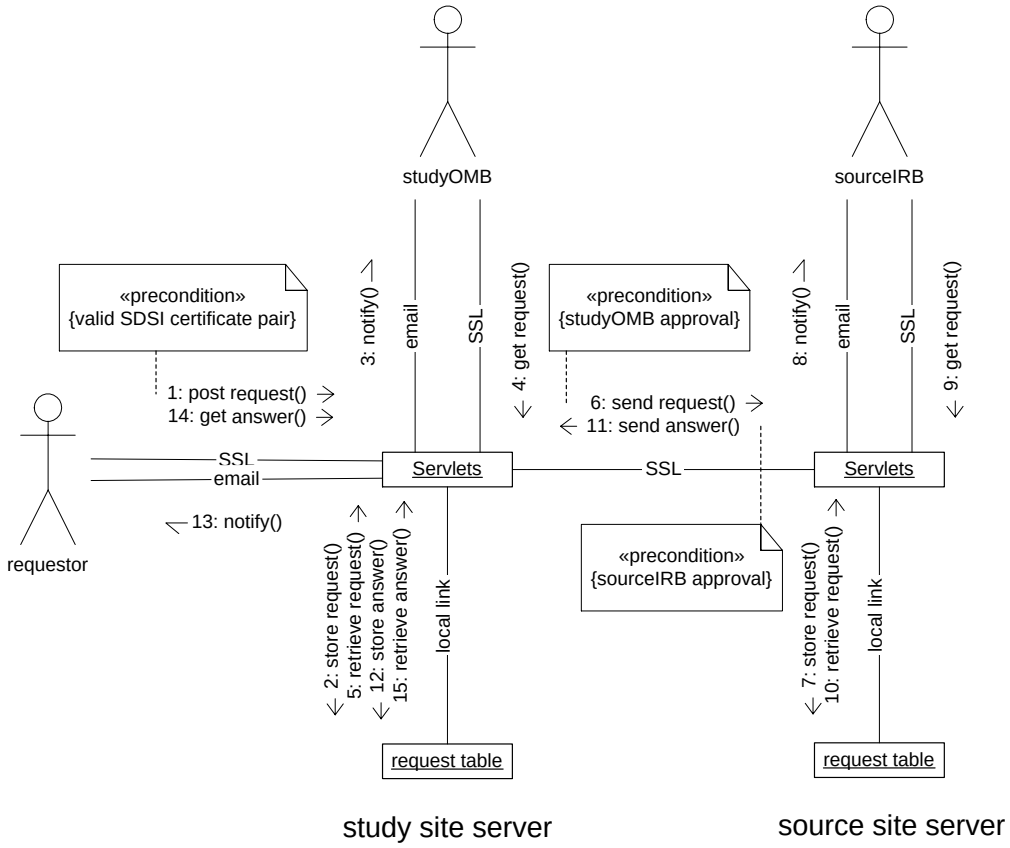

Figure 4.8: UML Collaboration Diagram for Patient Re-identification

4.4 Demo

We have built a two-server demo for SHARE. One server acts as a generation site, a study site and a source site, while the other server is another source site. We use Resin as a stand-alone server and a servlet engine. Configuring Resin to support SSL is simple. We register the JSSE provider (com.sun.net.ssl.internal.ssl.Provider) and add the following lines to Resin's configuration file.

```
<http port='443'>
  <ssl>true</ssl>
  <key-store-file>java keystore file</key-store-file>
  <key-store-password>password</key-store-password>
</http>
```

We open another port to support SSL with client authentication, which is used for server-server communications.

```
<http port='8443'>
  <ssl>true</ssl>
  <authenticate-client>true</authenticate-client>
  <key-store-file>Java keystore file</key-store-file>
  <key-store-password>password</key-store-password>
</http>
```

As discussed earlier, we need X.509 certificates to support SSL. For the demo, we use the Java keytool to generate a self-certificate for each server.

We use Microsoft SQL Server 7.0 as SHARE's database servers.

The demonstration works well to dynamically create the study database at the study site, to load sample data from two source sites and to re-identify a patient at two source sites with the appropriate approvals.

Chapter 5

Conclusions

5.1 Summary

We introduced SHARE, a web-based computer system for generating multi-center health studies, capable of sharing patient information across multiple institutions in a secure manner. We have demonstrated how patient information could be communicated in a controlled manner between a study site and a source site through a multi-layered encrypted patient identifier at the study site. Upon information request, the study-ID can then be decrypted only to identify the corresponding medical institution and the authorized principal capable of identifying the patient and extracting the requested information.

We build SHARE in Java. We widely use Java Servlet to make SHARE an on-line system. SHARE enables user to manipulate a multi-center health study through Internet, from the study's design, installation to operation.

SHARE is de-centralized and flexible. We do not attempt to propose a

Unique Patient Identifier for the healthcare industry. Because of the nation-wide UPI is not currently available, we argue that if a group of institutions want to do a multi-center study, they can create their own version secure and sharable patient identifier with encryption.

5.2 Current Defects

The current SHARE has some security holes that might be exploited by insiders, which will be eliminated in the next version. Although the communication between the study site server and the source site servers is secured and both servers are authenticated by SSL, an insider could maliciously use the real study site server to load patient data to his own database instead of the study database. To protect this, for each data collection, the source site should authenticate not only the study site server but also the study administrator, who actually loads the data. The same problem happens during the patient re-identification procedures. One way to eliminate this security hole is to let each source site issue a certificate to the study site, which can delegate such a certificate to its study administrator and studyOMB. When they trigger communication from the study site to the source site, they need to provide the corresponding certificate issued by that source site and delegated by the study site. A similar scheme is also necessary during the patient re-identification when the sourceIRB triggers communication from the source site to the study site to send the additional patient information. Another security problem is that although the current SHARE supports audit trails,

all the log files are open to the insiders to look at, modify or delete. We need to restrict log file access.

5.3 Future Work

SHARE is a starting point to specify, design and develop flexible and secure health information sharing systems for multi-center study with cryptographic-based patient identification schemes. There are many interesting areas for future research.

Patients may visit different health care institutions over their life times. In the current implementation of SHARE, we cannot link the same patient information from different source site. We assume that data about a particular patient will come from only a single source, and accept the loss of information and occasional duplication of data when in fact one of their patients deals with two or more sources each of which contributes to a study. However, we plan to develop the technology to automate the capability for the study site to integrate same patient's health data from different source sites. One way to achieve this functionality is to devise different naming mechanisms for the study-ID.

SHARE uses SDSI certificate to authenticate users. To make SHARE a fully-fledged system, a SDSI-version public-key infrastructure is indispensable. Furthermore, currently mapping the generator-designed data requirements to data available is a manual process. We are working on automating this mapping using information retrieval and ontology merging techniques.

Another interesting topic to investigate is the policy negotiation to create a multi-center health study and to operate it in a "trusted environment".

Finally, our goal is to make SHARE a working system used in the real world. Portability is therefore an important concern. Since different source site servers may install different types of database, with their own scheme for data manipulation, storage and transformation, SHARE needs to provide the corresponding functionality for each scheme. Fortunately, our system is built upon portable tools and widely adopted standards such as Java and XML, providing favorable conditions for the design of the final product. For exchange of health information, we also plan to support HL7 in our system.

Appendix A

Demonstration Scenario

We give part of the SHARE's demonstration scenario.

For multi-center study creation at the generation site server, figure A.1 shows the generator's login page. Figure A.2 is the web page for the generator to design the study database. Figure A.3 is the generator-designed database structure. Figure A.4 is the web page after the study database has been created at the study site server.

For patient data collection at the study site server, figure A.5 displays the newly created study that the study administrator needs to take care of. Figure A.6 and figure A.7 show the table name and column name mapping between the generator-designed database metadata and the source site database metadata. Figure A.8 is the web page after two source sites' data have been loaded.

For the patient re-identification between the study site and the source site, figure A.9 shows the researcher's login page. Figure A.10 is the web

page for the researcher to post her request. Figure A.11 is the studyOMB's login page. Figure A.12 shows that the studyOMB approves part of the request. Figure A.13 is the web page for the sourceIRB to get the request at the source site. Figure A.14 is the login page for the researcher to get the answer. Figure A.15 shows that the researcher finally gets the answer.

Figure A.1: Generation Site : Generator's Login

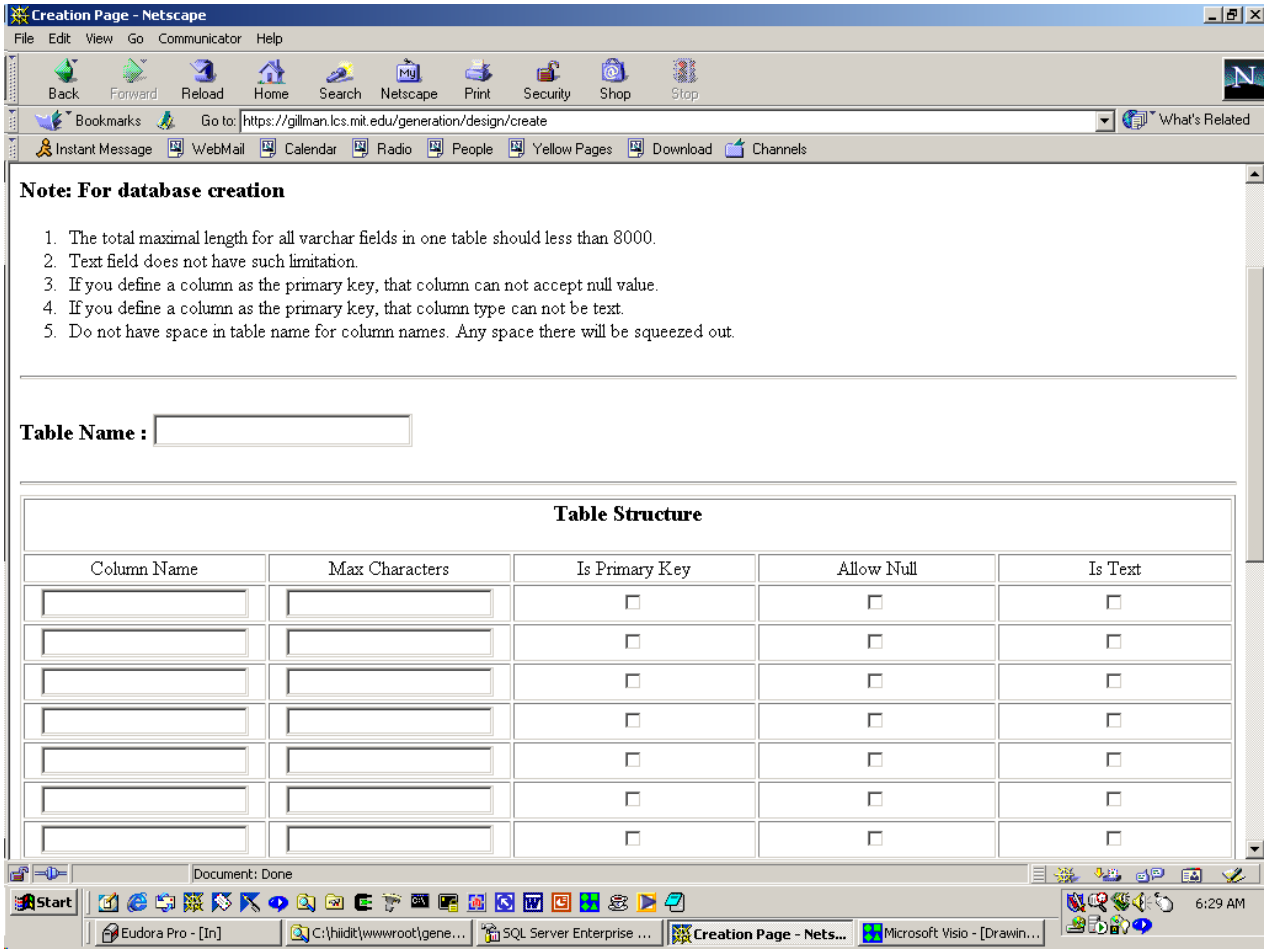

Figure A.2: Generation Site : Study Database Design

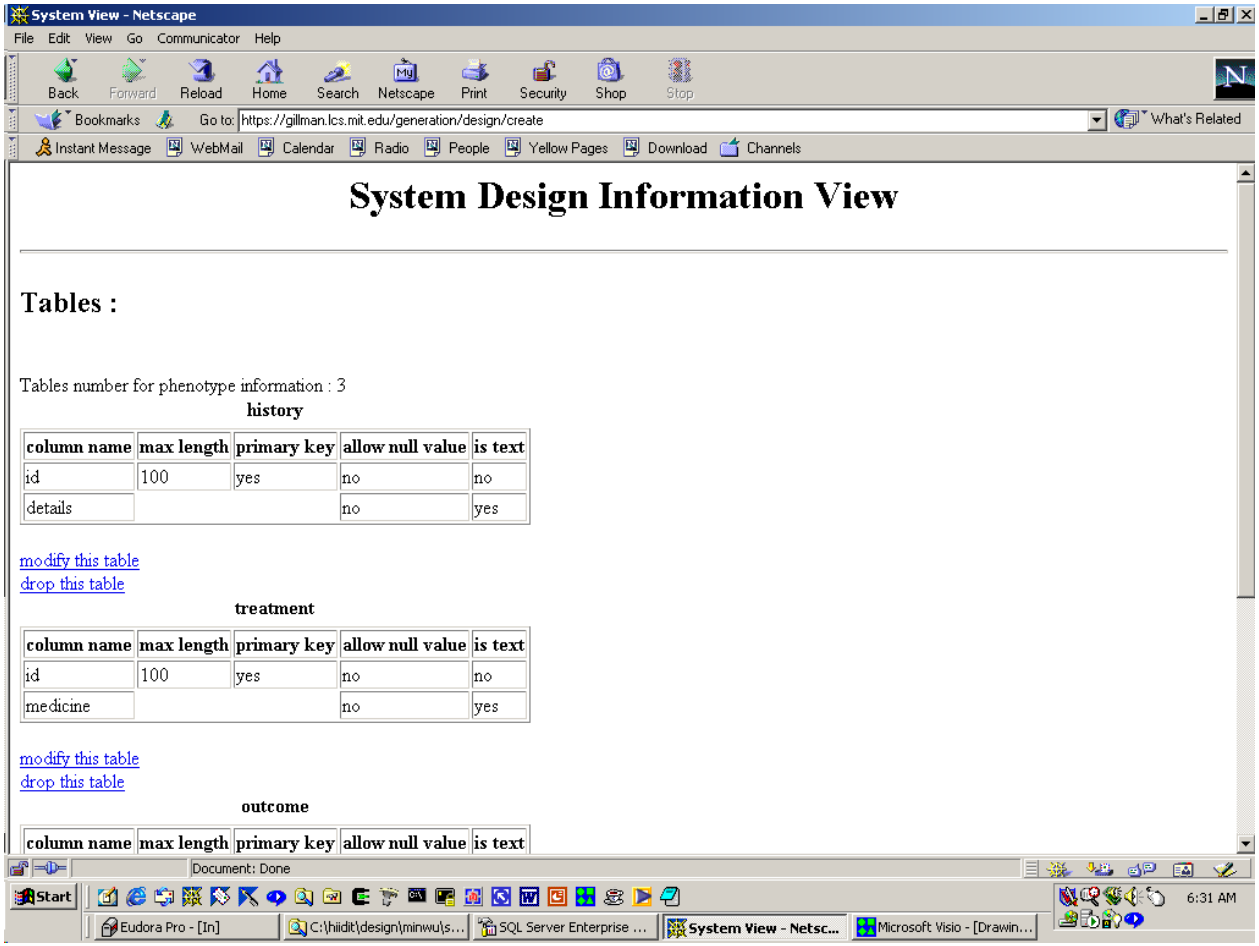

Figure A.3: Generation Site : Study Database Structure

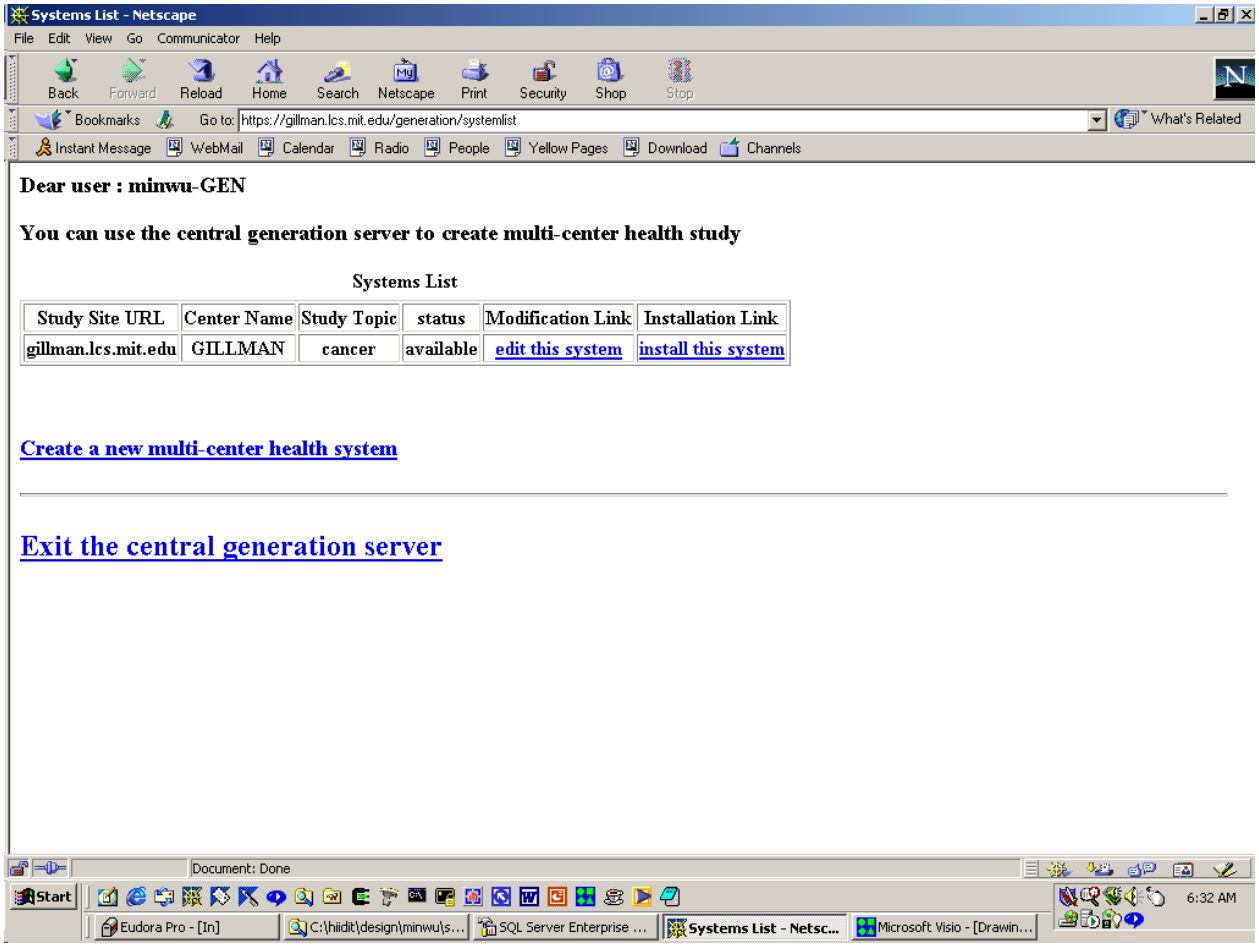

Figure A.4: Generation Site : Study Database Installation

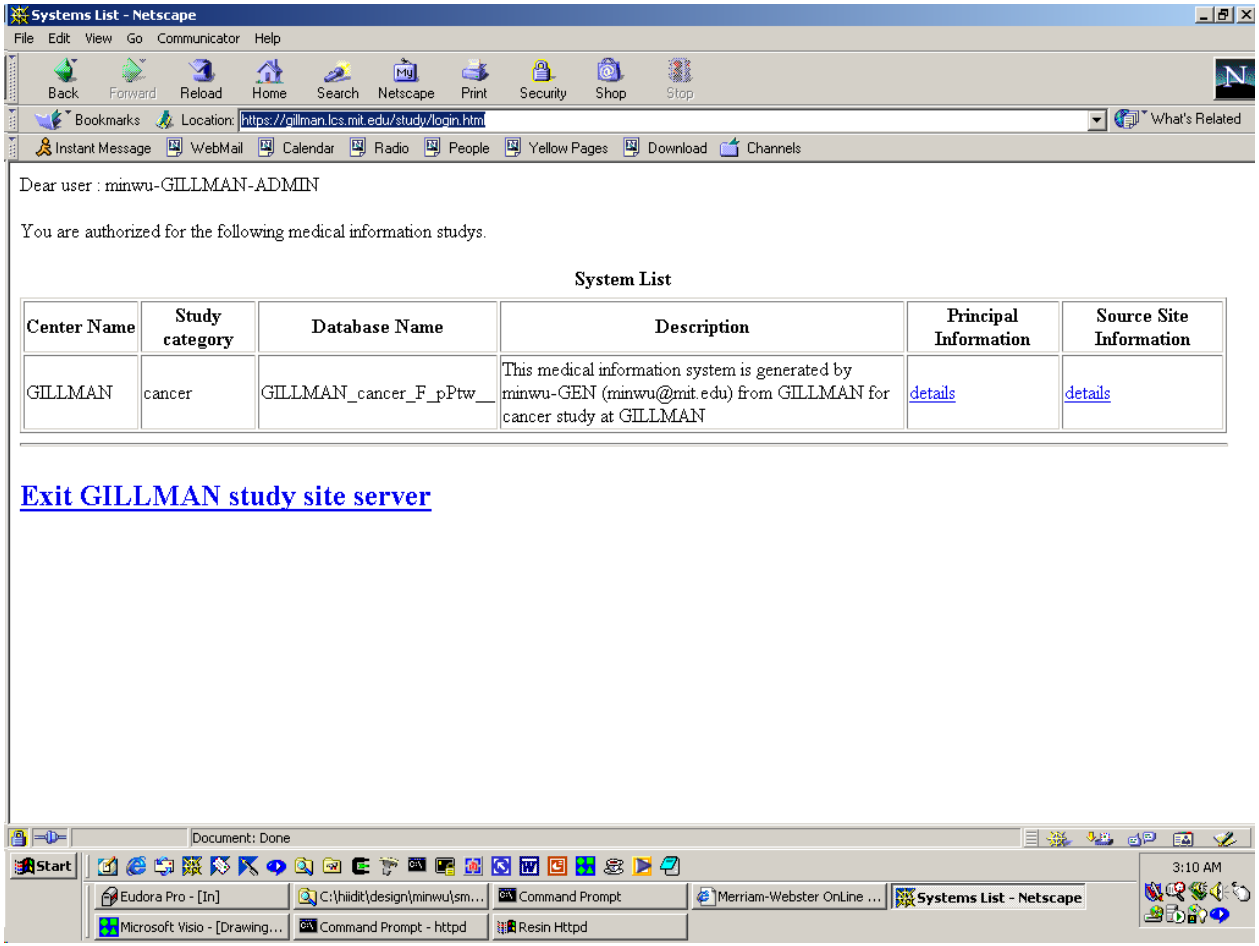

Figure A.5: Study Site : Study Information

Figure A.6: Study Site : Metadata's Table Matching

Figure A.7: Study Site : Metadata's Column Matching

Figure A.8: Study Site : Source Data Collection

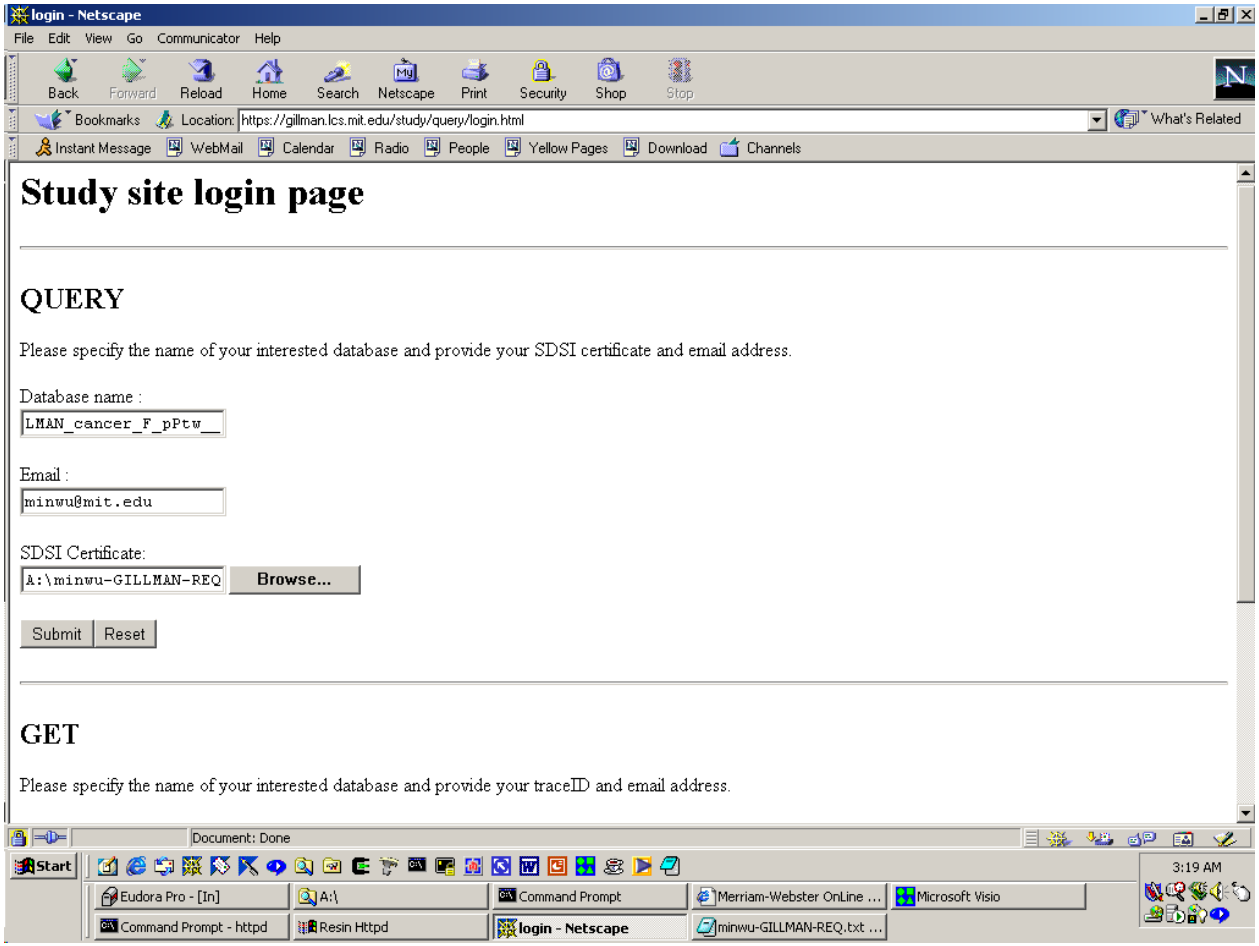

Figure A.9: Study Site : Researcher's Login For Request

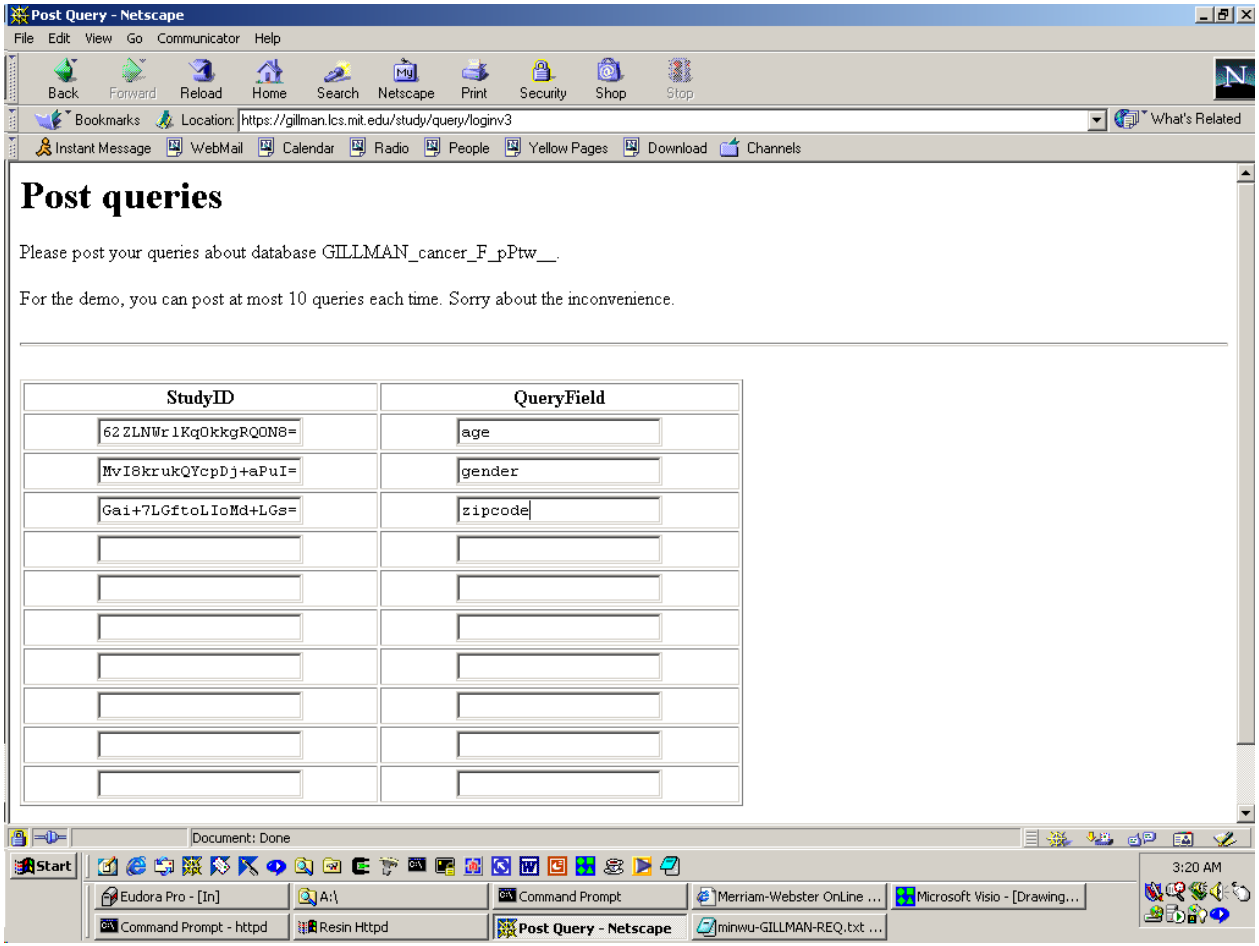

Figure A.10: Study Site : Researcher Posts Request

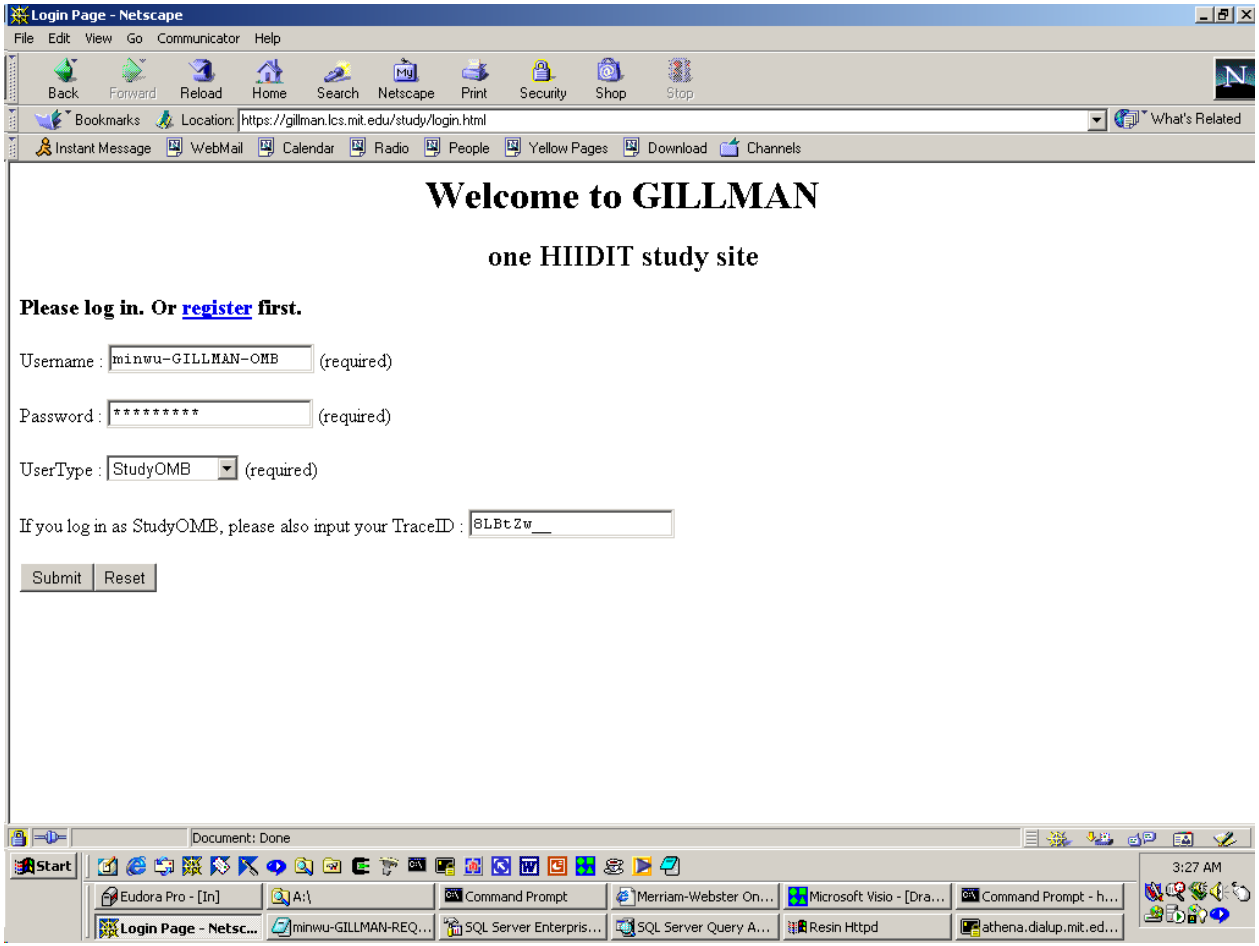

Figure A.11: Study Site : StudyOMB's Login

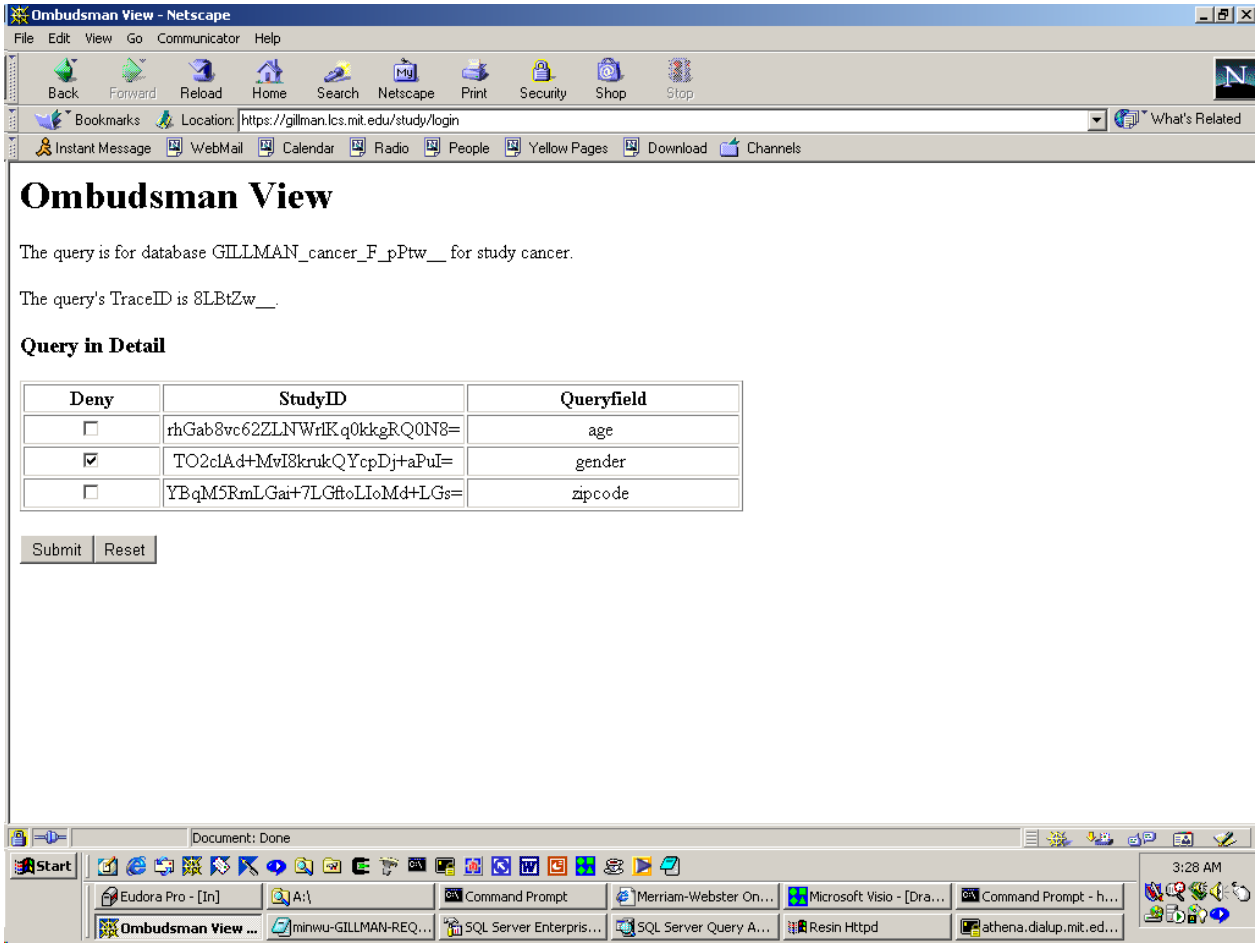

Figure A.12: Study Site : StudyOMB Approves Request

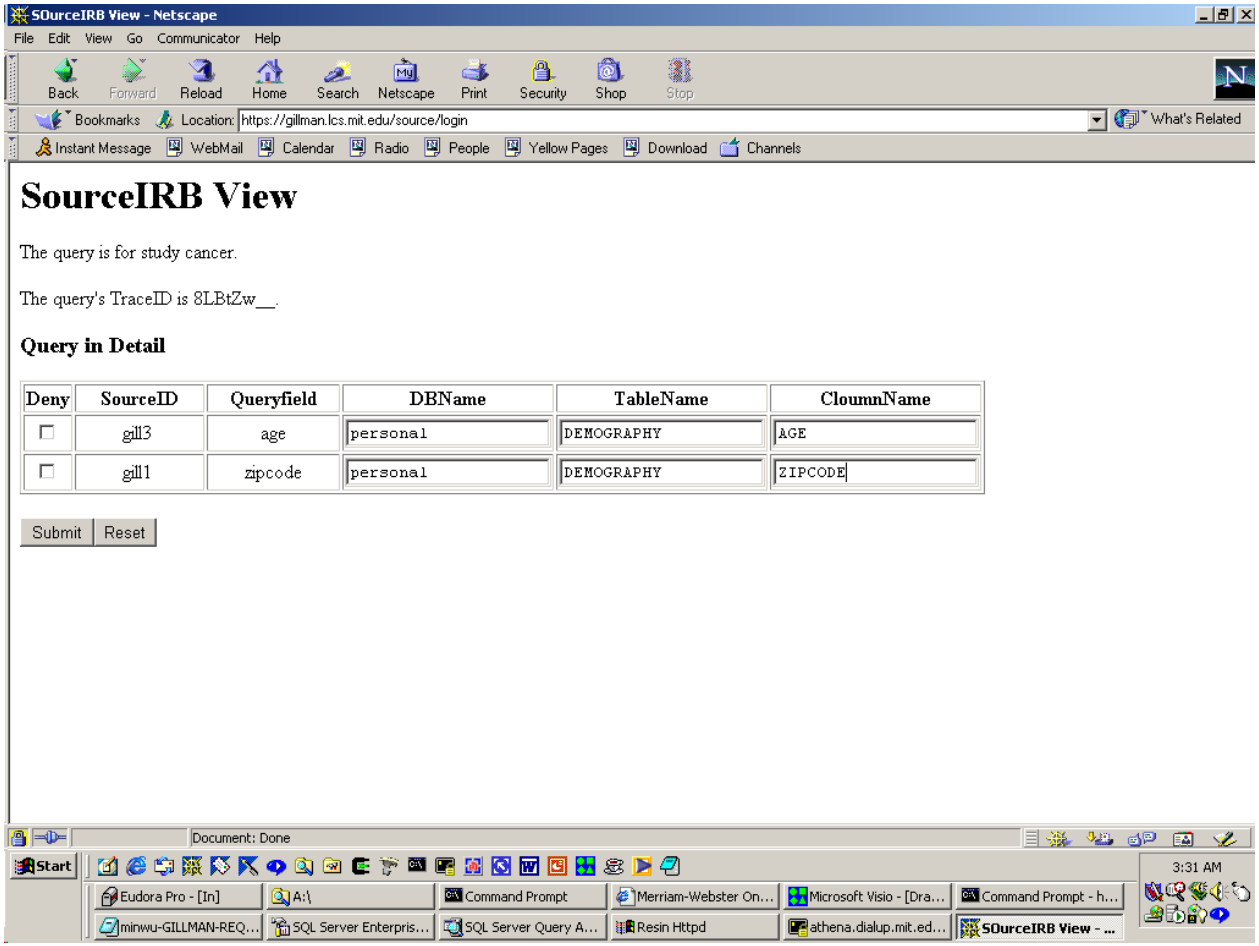

Figure A.13: Source Site : SourceIRB Approves Request

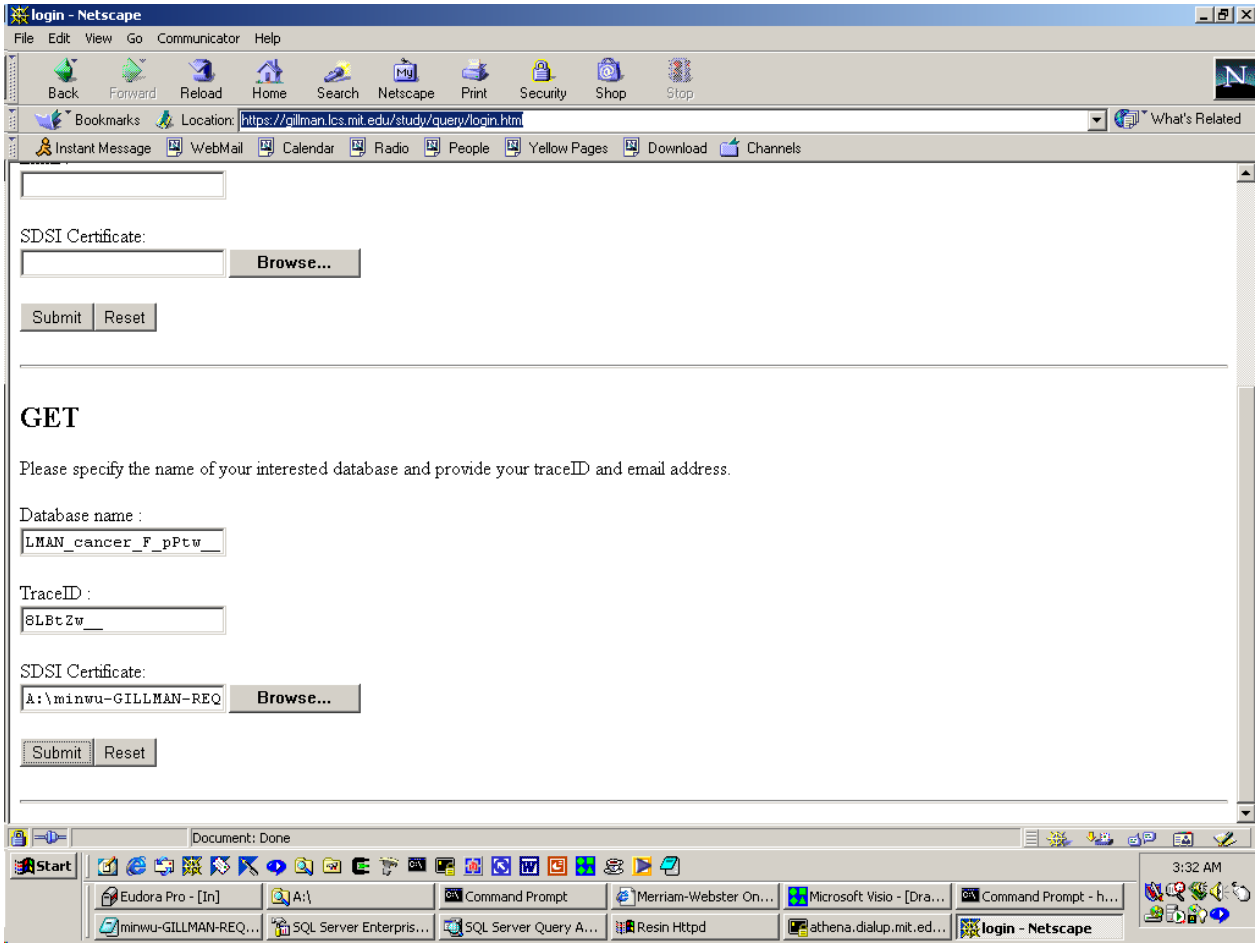

Figure A.14: Study Site : Researcher's Login for answer

Figure A.15: Study Site : Researcher Gets Answer

Bibliography

[1] Peter Szolovits, Issac S. Kohane (1994) *Against Simple Universal Health Identifiers.* Journal of the American Medical Informatics Association, pp. 316-319.

[2] Marc Rotenberg et al. (1993) *Letter From Privacy Advocates To Hillary Clinton Urging That The Social Security Number Not Be Used As The Health Identification Number.* http://www.epic.org/privacy/medical/ssn_letter.txt

[3] Soloman I. Appavu (1997) *Analysis of Unique Patient Identifier Options.* THE DEPARTMENT OF HEALTH AND HUMAN SERVICES.

[4] Latanya Sweeney (2001) *Computational Disclosure Control, A Primer on Data Privacy Protection.*

[5] Issac S. Kohane, Hongmei Dong, Peter Szolovits (1998) *Health Information Identification and De-identification Toolkit.* In Proc AMIA Symposium, pp. 356-360.

[6] MIT CIS Group (2001) *A Simple Distributed Security Infrastructure (SDSI).* http://theory.lcs.mit.edu/~cis/sdsi.html

[7] R. Housley, W. Ford, W. Polk, D. Solo (1999) *Internet X.509 Public Key Infrastructure.* RFC 2459