Archival and communication of DICOM images on a hospital network

M.Tech Project – Third Stage

Submitted in partial fulfillment of the requirement for the degree of
Master of Technology

by

Dr. Sheikh Mahmood H.M
99314008

Under the guidance of
Prof. Sridhar Iyer
KR School of Information Technology

Co-Guide
Prof. Puniyani
School of Biomedical Engineering

School of Biomedical Engineering
Indian Institute of Technology
Bombay

Acknowledgement

I would like to thank my guide, **Prof. Sridhar Iyer** for his invaluable guidance and encouragement during the project. My gratitude to **Prof. Puniyani**, my co-guide, **Mr. Joerg Riesmeier** and **Mr. Marco Eichelberg**, Healthcare Information and Communication Systems, OFFIS, Germany, and **Zainul**, my colleague, without whom this report would have been very difficult to complete.

Sheikh Mahmood H.M
Dept of Biomedical Engineering
IIT, Bombay
12th March 2001.

Abstract

The introduction of digital imaging devices in the medical world has made the management of images easier and more efficient. The Digital Imaging and Communications in Medicine (DICOM) standard promotes the communication of digital image information regardless of device manufacturer. Digital Images in a hospital are managed with Picture Archiving and Communication System (PACS), which receives, stores and makes these images available to Radiologists and Physicians for diagnosis and review. The interface provided to the user, by PACS is proprietary, with each vendor having separate user interface. This report describes the implementation and usage of World Wide Web interface to a DICOM based PACS that allows the user to query, select, move and display the images available in DICOM archive. The system supports querying of archive from any workstation (such as Windows, Mac) that supports a WWW browser.

In order to use the system, the user runs a WWW browser (such as Netscape, Internet Explorer) and specifies the URL of the server machine. On receiving the request, the server sends a query form to be displayed on the client. The query form contains three text input fields for patient name, patient ID and birthdate. The user may specify any or all the fields as well as wildcards in the name field.

Once the form is completed, the user clicks a button to submit the request. The HTML form submits the query to a CGI program that executes on the Unix server. This program accepts as input the form field values that the user specified. It then communicates with the archive via DICOM requests to determine those patients that match the search criteria. The user may then choose a patient, which in turn causes the studies for this patient to be displayed.

Finally, the user may select a study that causes those images to be retrieved from the archive and displayed via the Web browser. The result of this system is an easy to use interface to a DICOM PACS with the option to query or move images from PACS.

Table of contents

List of Figures

Chapter 1

Introduction

Distribution of images to referring clinicians outside the radiology department is critical. As radiology departments embark on developing filmless systems at the enterprise level, the need to distribute images electronically to various departments in a hospital becomes apparent. Traditionally, referring physicians have relied on signing out films from radiology department film libraries to communicate with patients, family members, and other physicians.

1.1 Medical Image Management

Images generated by devices like CT, MRI scanners, are inherently digital. They are read by Radiologists, on workstations or on lightbox after obtaining a hardcopy in the form of film. Then the images in digital form are archived by storing them in image archive. Images in film form are filed in image library. Archival is necessary for future reference and legal purpose. Picture Archiving and Communication System (PACS), is a means to acquire, store and transmit digital images. It provides access to these images through an interface, which usually has a format specified by the vendor [2]. A few disadvantages with such systems are

•Difficult to install and maintain
•Platform (and vendor) dependent
•Close system

Following a standard protocol for image format and communication resolves platform and vendor dependence and provides open architecture. Interfacing the system with World Wide Web makes it easier to install and maintain [3].
The concept of World Wide Web distribution is now gaining popularity and acceptance due to its ubiquity, cross-platform portability, and server-client distribution model. The Internet and intranet has increasingly become the technology base for image management and distribution. Current generations of PACS have not been designed around web technology, but vendors are now including web-based solutions for enterprise-wide image distribution. Web servers allow any physician, whether at home or in the clinical setting, to use a

desktop computer as a virtual light box for viewing radiological studies [1]. As of now, the only practical method for providing images to clinicians is this approach.

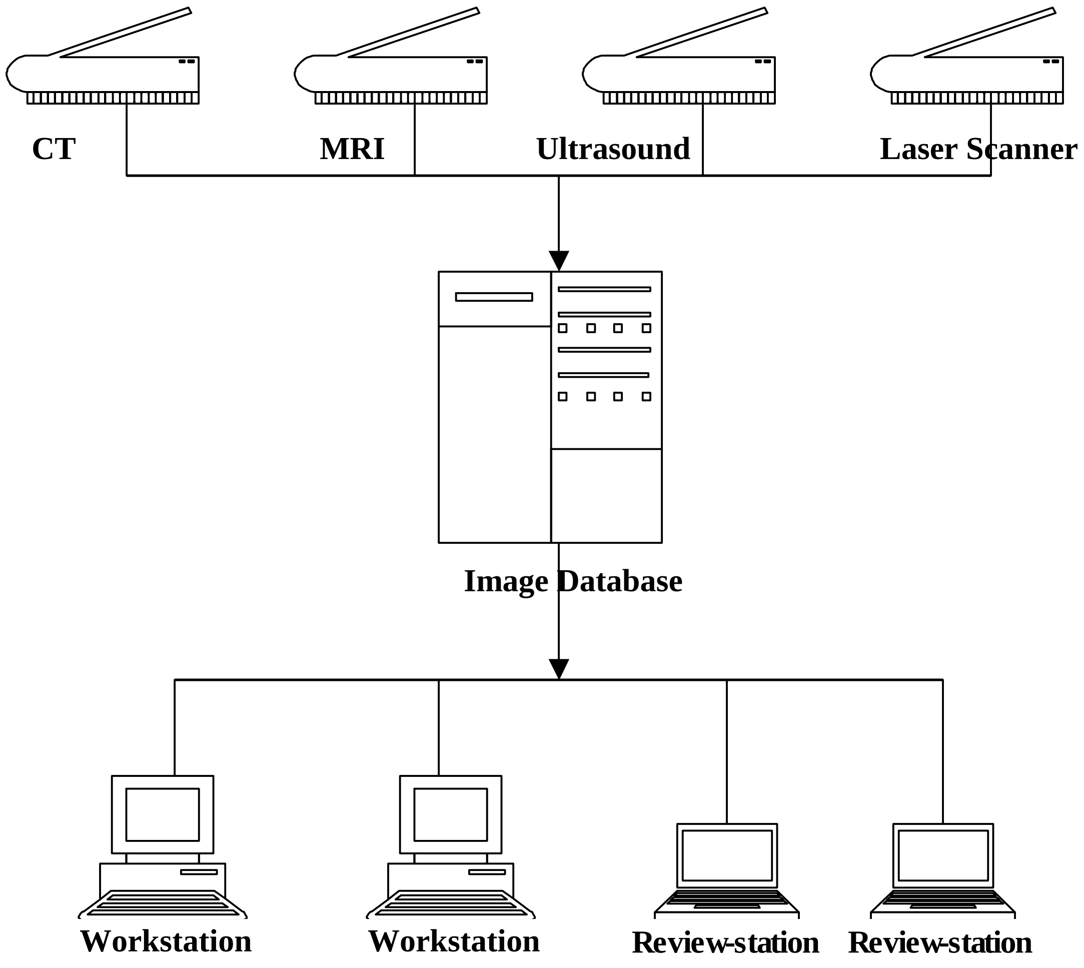


Figure 1: **Picture Archiving and Communication System**

1.2 Scope of the project

The digital images generated by medical imaging device used to be in proprietary format, it is difficult and sometimes impossible for other medical devices like radiology workstations, image archives and image printers to read the proprietary format. With a step to resolve this problem, the American College of Radiologists and National Electrical Manufacturers Association drafted a standard for medical image format and, communication of images and image related information on a network. The standard, known as DICOM, is getting to be accepted and implemented by a majority of vendors [4] for digital image storage and transmission in medicine. This has greatly helped in achieving inter-connectivity of DICOM compliant devices.

Open Source Software are available that implement parts of DICOM standard. These public domain libraries serve as stable DICOM implementation but the server can only be accessed in command line mode. This means the user has to know technical aspects of the implementation and remember various commands in order to execute separate modules. The title of the calling AE, called AE and the port number to which it is listening to, all have to be remembered to issue each command. The end users of our proposed system are doctors who may not be technically inclined and may be unwilling to remember various commands and their parameters to connect to the image server and query/retrieve images. In order for doctors to use the system, a graphical user interface has been developed which is simple and easy to use. It provides support for all query parameters that are required in making an efficient image query and the resulting images can be visualized in browser with the help of plug-in. The interface provided is browser based, which would be familiar to users.

1.3 Organization of the report

Chapter 1 gives a brief introduction about the management of digital medical images. The need for these images to be distributed to referring physicians is explained. Chapter 2 describes the problems encountered in accessing images, which are in proprietary format and how DICOM, the standard for medical image format and communication, has taken initiatives to resolve this problem. The technical aspects of this standard are discussed. Chapter 3 describes DICOM Toolkit, the implementation of DICOM standard that is available as open source software. The architecture of our system, DicomAccess, to query and retrieve images is explained in chapter 4. Chapter 5 describes the implementation aspect of all components involved in DicomAccess. It explains program modules that are executed on the server side when a query from the user is sent. The result obtained using various modules is reported in chapter 6. Screen snapshots of query interface and its output are shown. Chapter 7 discusses how the system can help physicians to decrease reporting time and manages digital medical images efficiently. It summarizes current work and functionalities that can be incorporated in future work.

Chapter 2

Digital Imaging and Communication in Medicine

With the increase in use of digital imaging modalities and computer applications in clinical domain, the American College of Radiology (ACR) and National Electrical and Manufacturers Association (NEMA) recognized the need for a standard method for transferring images and associated information between devices manufactured by various vendors.

2.1 Medical Imaging Standard

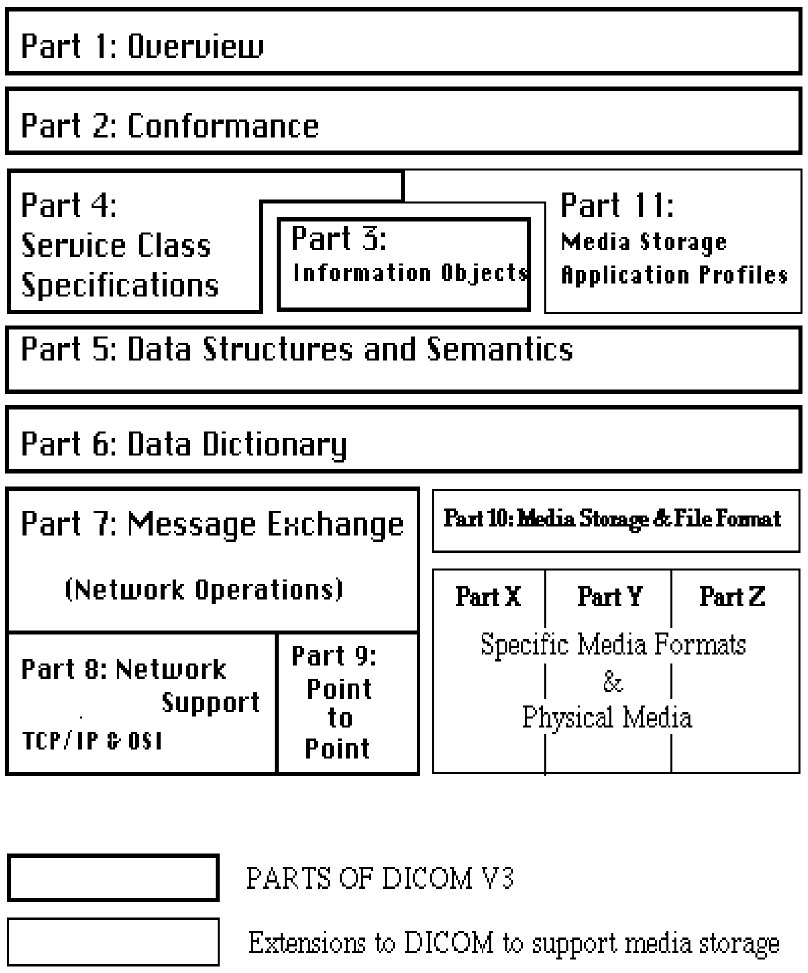
The ACR-NEMA formed a joint committee in 1983 to develop a standard to:

- Promote communication of digital image information, regardless of device manufacturer
- Facilitate the development and expansion of picture archiving and communication systems (PACS) that can also interface with other systems of hospital information
- Allow the creation of diagnostic information data bases that can be interrogated by a wide variety of devices distributed geographically.

2.1.1 Development of DICOM

ACR-NEMA Standard version1.0 was published in 1985 It was revised and version 2.0 was published in 1988. Version 3.0 of ACR-NEMA was approved in 1993 with the name, DICOM 3.0, which is the de facto international standard for the interchange of biomedical images and image-related information. DICOM Standard defines principles that implementations claiming conformance to the Standard shall follow. It specifies the general requirements, which must be met by implementation claiming conformance. The standard does not specify a testing/validation procedure to assess an implementation's conformance to the Standard. A Conformance Statement consists of three major parts [5]:

Set of Information Objects which is recognized by this implementation
Set of Service Classes which this implementation supports
Set of communications protocols that this implementation supports.



Part 1: Overview
Part 2: Conformance
Part 4: Service Class Specifications
Part 3: Information Objects
Part 11: Media Storage Application Profiles
Part 5: Data Structures and Semantics
Part 6: Data Dictionary
Part 7: Message Exchange (Network Operations)
Part 10: Media Storage & File Format
Part 8: Network Support TCP/IP & OSI
Part 9: Point to Point
Part X
Part Y
Part Z
Specific Media Formats & Physical Media
PARTS OF DICOM V3
Extensions to DICOM to support media storage

Figure 2: **Parts of DICOM Standard**

2.1.2 Information Model

The main functions provided by DICOM are:

- On-line and off-line transfer of images and associated information between image capture, processing and rendering devices
- Image and study (groups of related images) management
- Print management

DICOM performs these functions through combinations of information objects and operations performed on these objects (application services), supported by standardized image formats, a common information model, communications protocols and conformance specifications (application profiles).

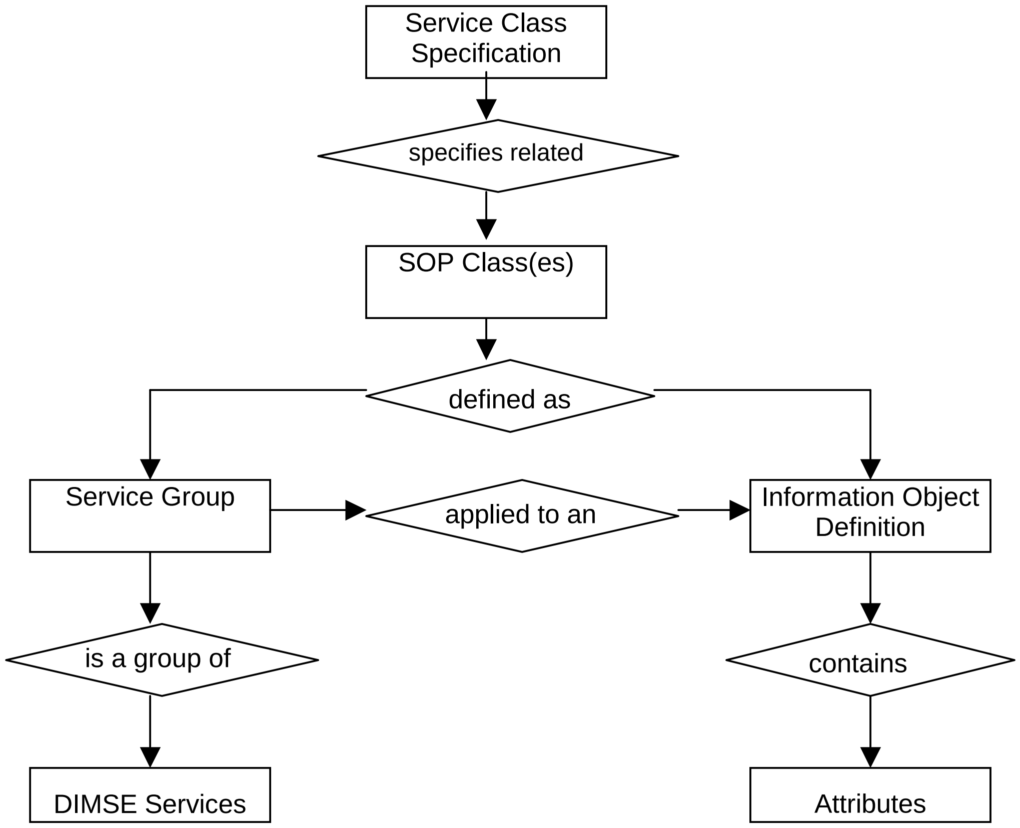


Figure 3: **Structure of DICOM Information Model.**

Information Objects are described in DICOM by Information Object Definitions (IODs) that specify groups of related data items. Each type of Information Object has its own set of characteristics or attributes, each of which can have its own value [6]. These attributes and their corresponding values differentiate between multiple incidents of the same information object. All information objects of the same type have the same attributes; it is their values that make them unique. Attributes can be mandatory, conditional or optional. The attributes of information objects are grouped into logical units called modules, representing sets of attributes, which belong together because, for example, of the type of information they describe or they depend on each other. For every information object, there is a corresponding set of modules.

IODs share common modules (for example, patient, general study, patient study, general series, general image, and image pixel) as well as containing modality-specific modules (for example, CR image, MR image). Information objects can be composite (containing related as well as inherent attributes) or normalized (containing only inherent attributes).

2.2 DICOM service class specification

DICOM Standard defines a number of Service Classes. A Service Class associates one or more Information Objects with one or more Commands to be performed upon these objects [7]. The project utilizes Query/Retrieve Service Class that defines an application-level class-of-service which facilitates the simple management of composite object instances.

Two peer DICOM AEs implement a SOP Class of the Query/Retrieve Service Class with one serving in the SCU role and one serving in the SCP role. SOP Classes of the Query/Retrieve Service Class are implemented using the DIMSE-C C-FIND, C-MOVE, and C-GET services

2.2.1 Image hierarchy

For any Query/Retrieve Information Model, an Entity-Relationship Model defines a hierarchy of entities, with Attributes defined for each level in the hierarchy (e.g. Patient, Study, Series, Object instance)

The SOP Class negotiated at Association establishment time identifies the Query/Retrieve Information Model. The SOP Class is composed of both an Information Model and a DIMSE-C Service Group.

2.2.2 Query retrieve level

Three standard Query/Retrieve Information Models are defined. Each Query/Retrieve Information Model is associated with a number of SOP Classes. The following three hierarchical Query/Retrieve Information Models are defined [7]: Patient Root, Study Root, Patient/Study Only

Patient Root Query/Retrieve Information Model

The Patient Root Query/Retrieve Information Model is based upon a four level hierarchy:
- Patient
- Study
- Series
- Object instance

The patient level is the top level and contains Attributes associated with the Patient Information Entity (IE) of the Composite IODs. Patients IEs are modality independent.

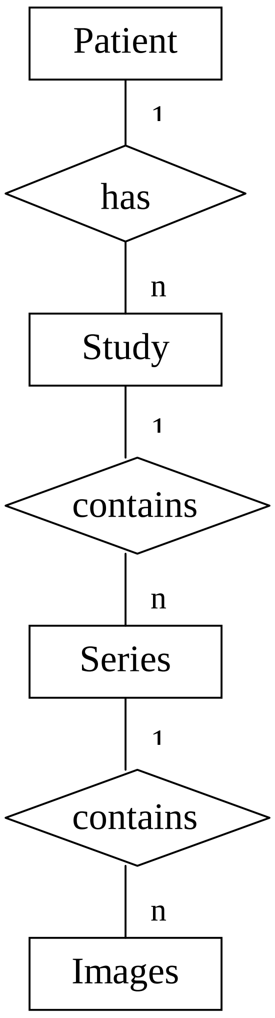


Figure 4: **Patient Root Query/Retrieve Information Model**

The study level is below the patient level and contains Attributes associated with the Study IE of the Composite IODs. A study belongs to a single patient. A single patient may have multiple studies. Study IEs are modality independent.

The series level is below the study level and contains Attributes associated with the Series, Frame of Reference and Equipment IEs of the Composite IODs. A series belongs to a single study. A single study may have multiple series. Series IEs are modality dependent. To accommodate this modality dependence, the set of Optional Keys at the series level includes all Attributes defined at the series level from any Composite IOD.

The lowest level is the composite object instance level and contains Attributes associated with the Composite object IE of the Composite IODs. A composite object instance belongs to a single series. A single series may contain multiple composite object instances. Most composite object Image IEs are modality dependent. To accommodate this potential modality dependence, the set of Optional Keys at the composite object instance level includes all Attributes defined at the composite object instance level from any CompositeIOD.

Study Root Query/Retrieve Information Model
The Study Root Query/Retrieve Information Model is identical to the Patient Root Query/Retrieve Information Model except the top level is the study level. Attributes of patients are considered to be Attributes of studies.

Patient/Study Only Query/Retrieve Information Model
The Patient/Study Only Query/Retrieve Information Model is identical to the Patient Root except the series and composite object instance levels are not supported. Even though this model does not include the composite object instance level, composite object instances may be retrieved at the patient and study level (i.e., retrieve all composite object instances for a Patient or Study).

Additional Query/Retrieve Attributes
Some optional attributes, which may be used in Query/Retrieve Information Models, that are not Attributes of an Information Object Definition and, therefore, are not defined.

2.2.3 Image Archive

The DICOM archive has a database of image files and image related information. The images that are produced by an imaging modality are stored in an image database. These files contain the pixel data and header information. The database of image related information has data regarding patient, study, series and image components. The exact location of an image file pertaining to this data is also stored.

The communication between each module is preceded by a negotiation to establish a common ground. Upon this, the communication necessary to perform the task the user has requested is built.

The client queries the server with query/retrieve service class, which specifies the following three services:

C-FIND: enables a client to query a server for matches against a template of key values. It also enables the server to return to object instance identifiers of any matching records to the clients.

C-GET: retrieves objects that match a set of key values.

C-MOVE: enables a user to initiate the transfer of DICOM objects between two locations.

Image and image related information is stored using storage service class, which specifies C-STORE service. This service enables a client to transfer (push) a DICOM object to a server or archive, for storage. During the negotiation that occurs between the client and server processes to establish a session, the client notifies the server of the class of object it proposes to transfer and the server confirms that it supports that information object class. A unique service class identifier, the (storage) SOP class UID, is defined for storage of each information object class so that the server can allocate appropriate resources.

DICOM ToolKit (DCMTK) and Central Test Node [8] are two important public domain implementation of DICOM standard. The toolkits are collection of libraries, implementing large parts of DICOM standard. They are made available as Open Source software.

Central Test Node has good features for basic implementation but this library lacks needed functionality in several areas. One obvious point is that patient data can only be accessed at the 'Study' level. A study can consist of close to 100 DICOM files, containing approximately 700 fields apiece. DCMTK provides access to images at Patient, study and series and level, besides a particular image can be queried based on a few IOD attribute values.

Chapter 3

DICOM Toolkit

DICOM Toolkit is a collection of libraries and applications implementing large parts of the DICOM standard for medical image communication. It includes libraries for constructing and converting DICOM image files, handling offline media, sending and receiving images over a network connection, as well as demonstration image storage servers. DCMTK includes complete source code and is written in a mixture of ANSI C and C++. The toolkit is made available as Open Source software. DCMTK has been used at numerous DICOM demonstrations to provide central, vendor independent, image storage servers. It is used by hospitals and companies all over the world for a wide variety of purposes ranging from being a tool for product testing to being a building block for research projects, prototypes and commercial products [9].

The DCMTK is a open source software which can be downloaded via http://www.offis.uni-oldenburg.de/projekte/dicom/

Configuration

The Application Entities, which interact with image server over a network, are represented in the configuration file. The image server looks into this file and allows association only to those AE mentioned in this file. Each DICOM application entity is defined by a triple consisting of Application Entity Title, host name and TCP/IP port number. The configuration file also contains information about image storage areas, located in the specified file system directory.

3.1 Imagectn

The Image Central Test Node (imagectn) Application implements a simple image archive. It manages a number of storage areas and allows images to be stored in these storage areas using the DICOM Storage Service Class. It also allows image attributes to be queried and images to be retrieved using the DICOM Query/Retrieve Service Class. The imagectn application also implements access restriction rules to limit operations to specific peer application entities [10].

Usage
imagectn
[options] [port]

port: specifies the tcp/ip port number to listen on when waiting for association requests.

options:

```
-v      --verbose   verbose mode
-c      --config    [f]ilename: string (default: configrc)
```

This option allows a configuration file to be specified. By default, the configuration file in the current working directory with the name "configrc" will be used.

Operation
imagectn waits for another application to connect at the presentation address (port number) specified in its configuration file (or overridden by a command line option). When another application connects, imagectn expects it to be a DICOM application and to use calling and called AE Titles specified in imagectn's configuration file.

imagectn accept associations with Presentation Contexts for SOP Classes of the Verification Service Class, Storage Service Class and Query/Retreive Service Class. Associations will be rejected or Presentation Contexts will be refused if the peer application does not have appropriate access rights as specified by imagectn's configuration file. imagectn will receive images on Presentation Contexts of the Storage Service Class, write them to a configurable storage area associated with the class AE title, extract attributes from these images and store them in a database. imagectn will receive query requests and generate query responses on Presentation Contexts of the Query/Retrieve Service class. imagectn will receive retrieve requests and generate retrieve responses on Presentation Contexts of the Query/Retrieve Service class.

Under normal operations imagectn will never exit, it keeps on waiting for new associations until killed.

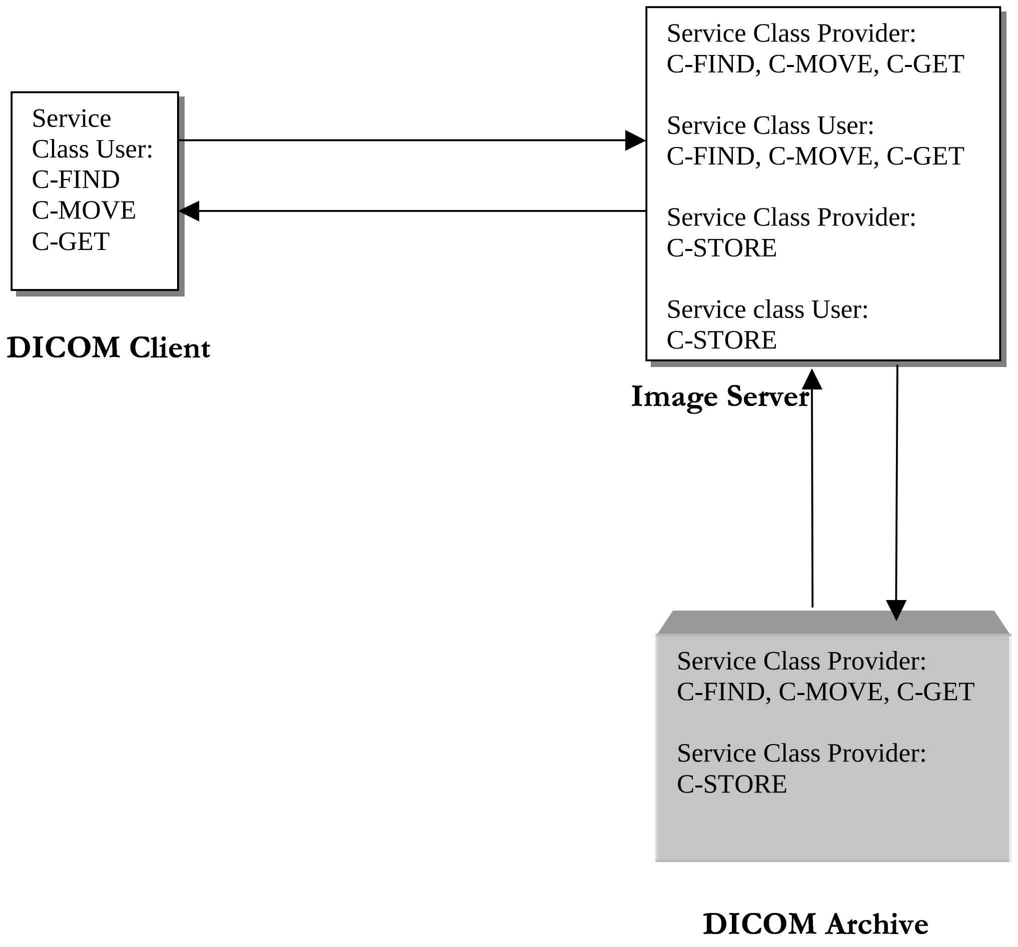


Figure 5: **Components of DICOM Toolkit (DCMTK).**

3.2 Storescu

The storescu application implements an SCU for the Storage Service Class. For each image file command line parameter it sends a C-STORE message to Storage SCP and waits for a response. The application can be used to transmit DICOM images.

Usage

storescu [options] peer port dcmfile_in...

peer	hostname of DICOM peer
port	tcp/ip port number of peer
dcmfile_in	DICOM file(s) to be transmitted

options

-v	--verbose	verbose mode, print processing details
-aet	--aetitle	aetitle: string set calling AE title
-aec	--call	aetitle: string set called AE title of peer

Operation

The default behavior of storescu is to propose two-presentation contexts for each supported SOP class (abstract syntax) – one with the preferred transfer syntax and one with all other uncompressed transfer syntaxes. The default preferred transfer syntax is explicit VR with byte ordering corresponding to the local byte ordering of the machine on which storescu is running

3.3 Findscu

The findscu application implements an SCU for the Query/Retrieve Service Class. Findscu only supports query functionality using the C-FIND message. It sends query keys to an SCP and awaits responses. The application can be used to test SCPs of the Query/Retrieve Service Class.

Usage

Findscu [options] peer port [dcmfile_in...]

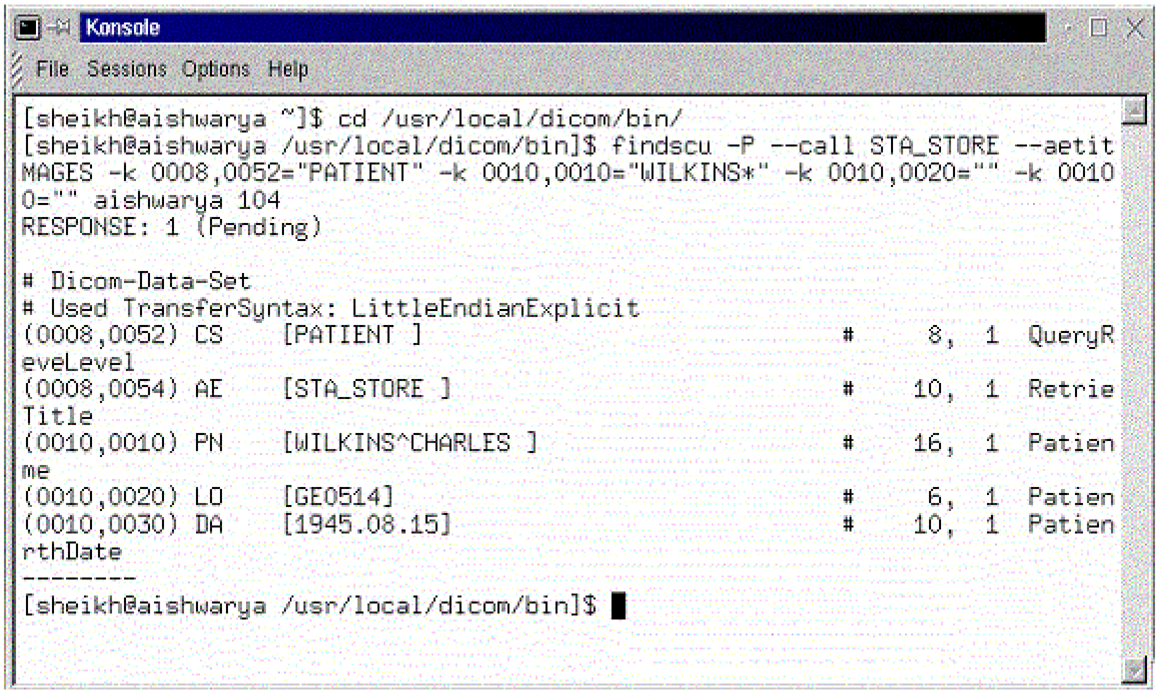
peer	hostname of DICOM peer
port	tcp/ip port number of peer
dcmfile_in	DICOM query file(s)

options

-v	--verbose	verbose mode, print processing details
-k	--key	key: gggg,eeee="string" matching key

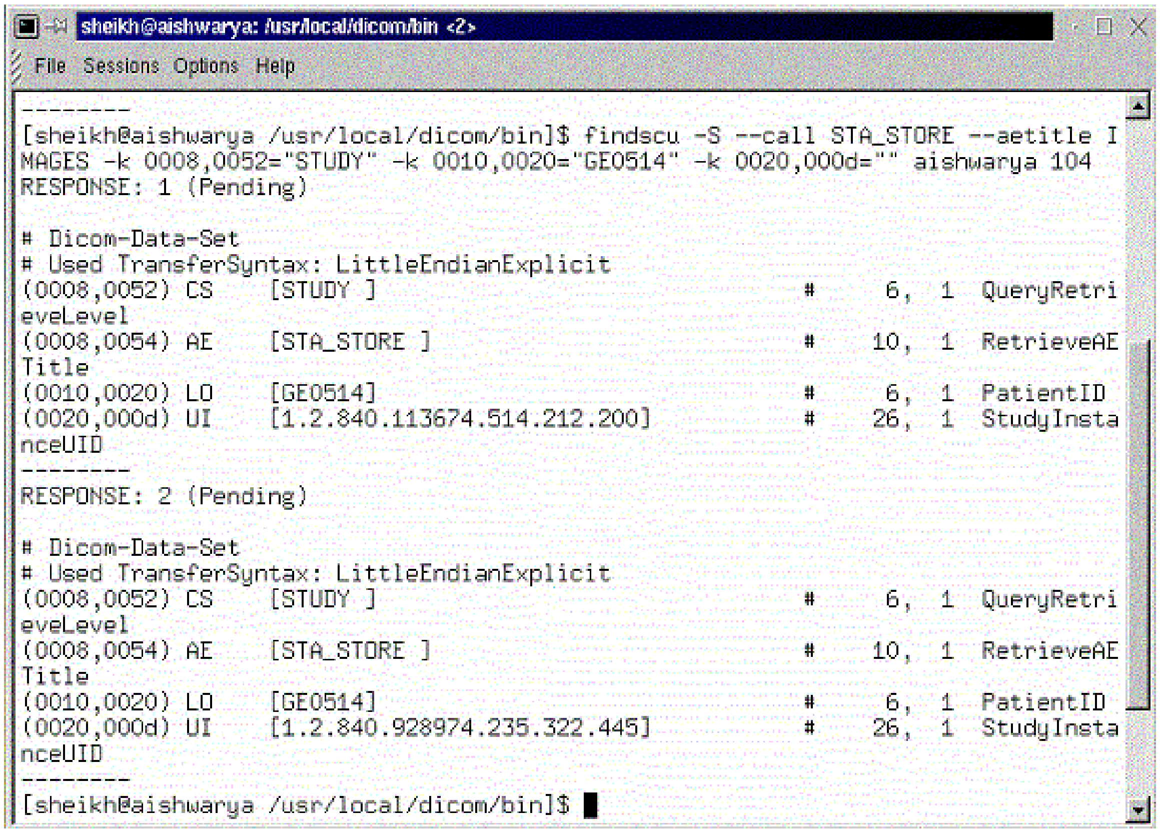
query information model

-W	--worklist	use modality worklist information model (default)
-P	--patient	use patient root information model
-S	--study	use study root information model



```
[sheikh@aishwarya ~]$ cd /usr/local/dicom/bin/
[sheikh@aishwarya /usr/local/dicom/bin]$ findscu -P --call STA_STORE --aetit
MAGES -k 0008,0052="PATIENT" -k 0010,0010="WILKINS*" -k 0010,0020="" -k 0010
0="" aishwarya 104
RESPONSE: 1 (Pending)

# Dicom-Data-Set
# Used TransferSyntax: LittleEndianExplicit
(0008,0052) CS    [PATIENT ]                              #    8,  1  QueryR
eveLevel
(0008,0054) AE    [STA_STORE ]                            #   10,  1  Retrie
Title
(0010,0010) PN    [WILKINS^CHARLES ]                      #   16,  1  Patien
me
(0010,0020) LO    [GE0514]                                #    6,  1  Patien
(0010,0030) DA    [1945.08.15]                            #   10,  1  Patien
rthDate
---------
[sheikh@aishwarya /usr/local/dicom/bin]$ █
```

Figure 6: **Query and result of findscu for a patient named WILKINS**


sheikh@aishwarya: /usr/local/dicom/bin <2>
File Sessions Options Help

```
---------
[sheikh@aishwarya /usr/local/dicom/bin]$ findscu -S --call STA_STORE --aetitle I
MAGES -k 0008,0052="STUDY" -k 0010,0020="GE0514" -k 0020,000d="" aishwarya 104
RESPONSE: 1 (Pending)

# Dicom-Data-Set
# Used TransferSyntax: LittleEndianExplicit
(0008,0052) CS    [STUDY ]                                #    6,  1  QueryRetri
eveLevel
(0008,0054) AE    [STA_STORE ]                            #   10,  1  RetrieveAE
Title
(0010,0020) LO    [GE0514]                                #    6,  1  PatientID
(0020,000d) UI    [1.2.840.113674.514.212.200]            #   26,  1  StudyInsta
nceUID
---------
RESPONSE: 2 (Pending)

# Dicom-Data-Set
# Used TransferSyntax: LittleEndianExplicit
(0008,0052) CS    [STUDY ]                                #    6,  1  QueryRetri
eveLevel
(0008,0054) AE    [STA_STORE ]                            #   10,  1  RetrieveAE
Title
(0010,0020) LO    [GE0514]                                #    6,  1  PatientID
(0020,000d) UI    [1.2.840.928974.235.322.445]            #   26,  1  StudyInsta
nceUID
---------
[sheikh@aishwarya /usr/local/dicom/bin]$ █
```

Figure 7: **Query and result of studies, using findscu**

Each file supplied on the command line will be sent to the SCP as part of a C - FIND request. The query file must be a valid DICOM data set containing the dataset part of a C-FIND-RQ message.

3.4 Movescu

The movescu application implements both an SCU for the Query/Retrieve Service Class and an SCP for the Storage Service Class. Movescu supports retrieve functionality using the C-MOVE message. It sends query keys to an SCP and awaits responses. It will accept associations for the purpose of receiving images sent as a result of the C-MOVE request. The application can be used to test SCPs of the Query/Retrieve Service Class. The movescu application can initiate the transfer of images to a third party or can retrieve images to itself. The use of the term "move" is a misnomer. The C-MOVE operation actually performs an image copy (no images will be deleted from the SCP) [10].

Usage

```
movescu [options] peer port [dcmfile_in...]
peer              hostname of DICOM peer
port              tcp/ip port number of peer
dcmfile_in        DICOM query file(s)
```

options

```
-k    --key           key: gggg,eeee="string" override key
```

query information model

```
-P   --patient        use patient root information model (default)
-S   --study          use study root information model
-O   --psonly         use patient/study only information model
```

application entity titles

```
-aet  --aetitle       aetitle: string set  calling AE title
-aec  --call          aetitle: string set called AE title of peer
-aem --move           aetitle: string set move destinat. AE title
```

Each file supplied on the command line will be sent to the SCP as part of a C - MOVE request. The query file must be a valid DICOM data set containing the dataset part of a C-MOVE-RQ message. For all uses of movescu, the SCP must be configured to "know" about the AE title where the images are to be transmitted. This is required since the C-MOVE request only contains the AE Title of the target and the SCP must be able to convert this AE Title into TCP/IP address and port number in order to actually send the images over a separate association.

The C-MOVE operation of the Query/Retrieve Service Class is able to retrieve images or to initiate a copy of images to a third party [2].

The user can issue the required command to image server. The system is set in such a manner that the imaging modalities that generate images, can communicate with the server using storescu. Users from different departments of a hospital use findscu and movescu for communicating with the server. Next chapter describes the architecture of the system.

Chapter 4

System Design

DicomAccess, the system we have developed, consists of two basic parts: the first part performs query and navigation through the DICOM archive image folders. The second part does the image access and display. While the first part deals with low data traffic, it involves many database transactions. The second part is simple as far as access transactions, but requires much more data traffic and display function [11].

DicomAccess allows the query and retrieval of information stored in DICOM archive. It caters to the needs of Radiologists and Referring Physicians for querying and viewing images.

4.1 Client

There are two clients –WWW client and –DICOM client. The WWW client can be any Personal Computer with permission to access the web server. A web browser is installed on this client through which a graphical user interface to the system is provided. The user logs on to the web server and is provided with Patient query form [12]. The form contains patient query fields like patient name, patient ID and date of birth. Fields related to Study are also provided. On filling out any/all query fields, the form is submitted by clicking on "Search" button. The "Clear" button clears entries made in all the fields. The user can request for a specific list of patients whose result is output in the form of a table. A particular patient is selected by clicking on the hyperlink that contains the patient ID, which is unique. On clicking the patient ID, a table is displayed that contains information about all the studies and series available for the patient., the study ID, date on which the study was carried out and the series number. The user has a choice of viewing the whole study or any series contained in a study, by clicking on the hyperlink of study ID or series number respectively. Using study query fields, the user can query for studies say, performed between two specified dates or studies performed after or before a specified date. The output is in the form of table with hyperlink to study ID or series number.

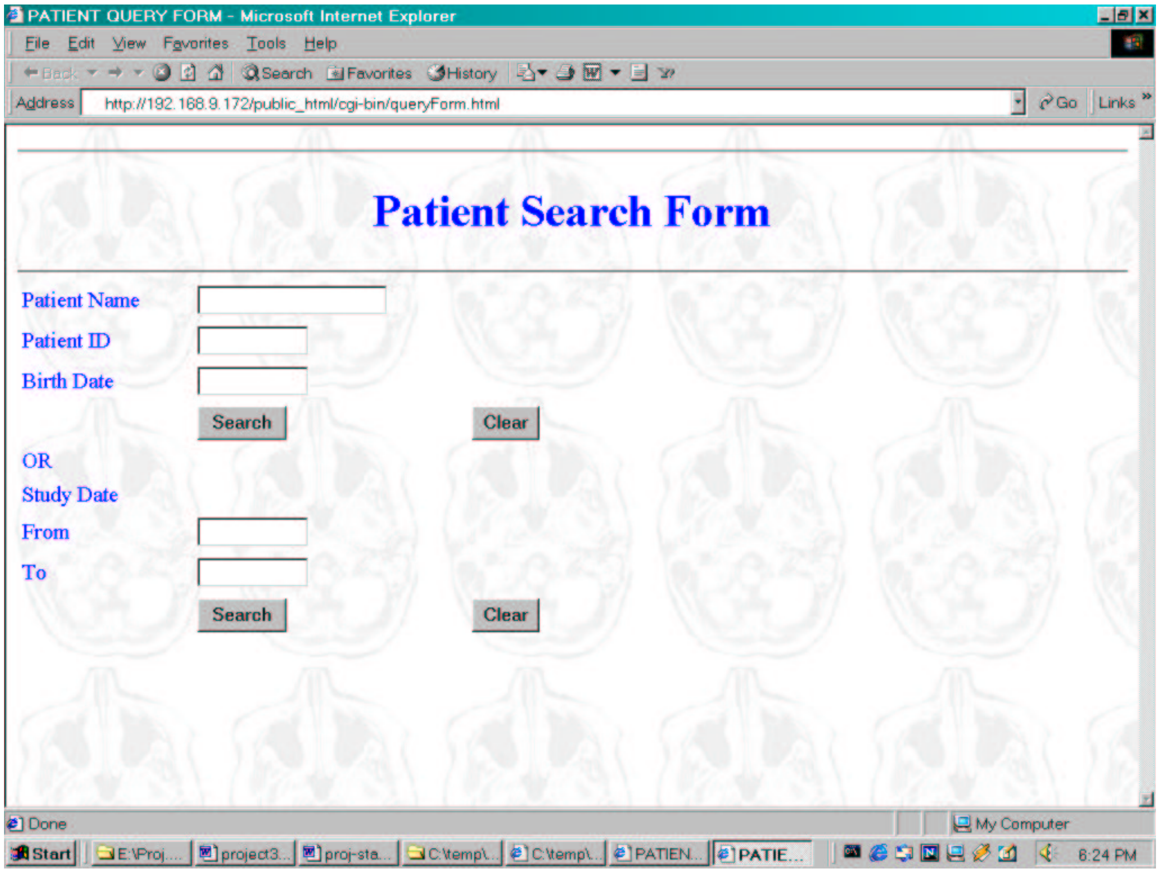

PATIENT QUERY FORM - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://192.168.9.172/public_html/cgi-bin/queryForm.html
Patient Search Form
Patient Name
Patient ID
Birth Date
Search
Clear
OR
Study Date
From
To
Search
Clear
Done
My Computer
6:24 PM

Figure 8: **Screen showing a patient search form. The user can select several criteria to search for a specific list of patients.**

The DICOM client essentially establishes a connection with DICOM image server using DICOM protocol. It takes the statement from CGI modules and queries the image server for studies available of a particular patient or studies taken from the specified dates. It uses findscu module of DICOM toolkit for querying patient and study details. Once a study or series is selected, movescu module of DICOM toolkit is used to retrieve it from the Image archive.

4.2 Web Server

The web server consists of HTTP server and CGI modules. HTTP server collects the parameters passed from WWW client and passes them to various CGI modules based on the file name specified [13]. The parameters are passed using POST or GET methods. CGI programs read this information and extract the values of various parameters using parsing function. The parameters obtained are used to a build query, which is passed on to DICOM client. A temporary file receives the output from DICOM server and this is read by CGI program. This is stored in a file and processed to extract required information. Information thus obtained is used to generate dynamic HTML pages that are sent to WWW client to be rendered with a browser.

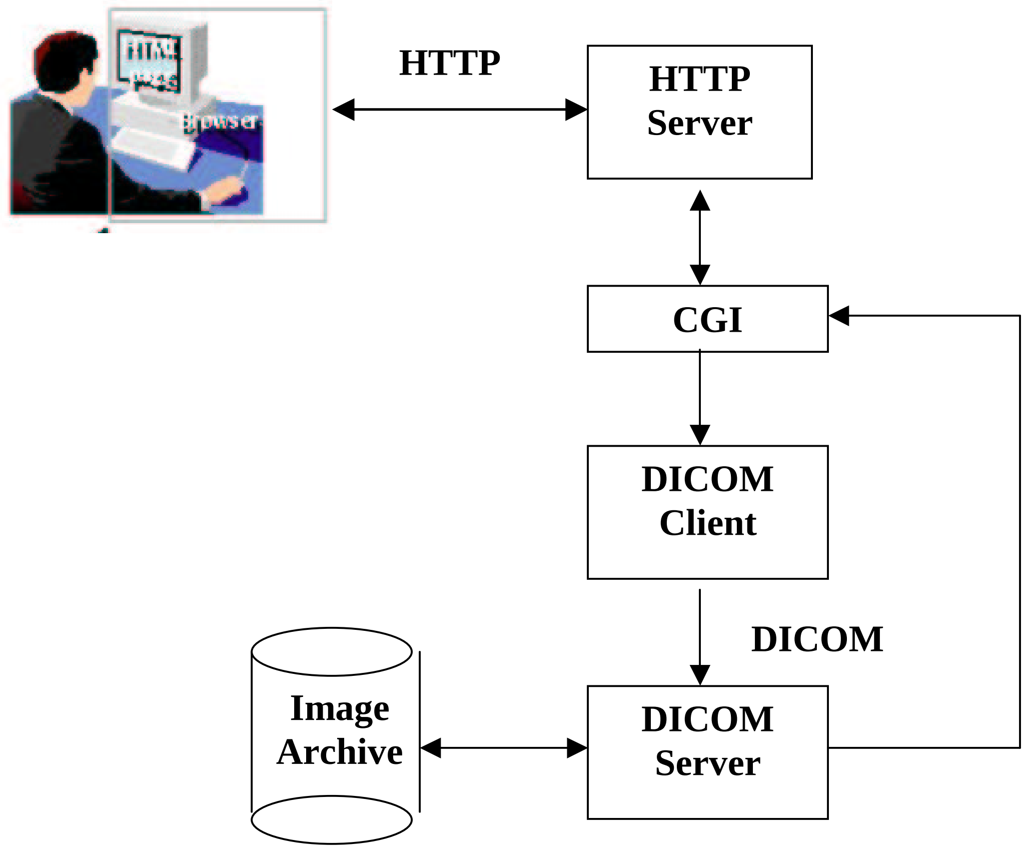


Figure 9: **Architecture of DicomAccess**

4.3 DICOM Image Server

The DICOM image server waits for another application to connect at the presentation address (port number) specified in its configuration file (or overridden by a command line option). When another application connects, the image server expects it to be a DICOM application and to use calling and called AE Titles specified in server's configuration file.

The image server accepts associations with Presentation Contexts for SOP Classes of the Verification Service Class, Storage Service Class and Query/Retrieve Service Class. Associations will be rejected or Presentation Contexts will be refused if the peer application does not have appropriate access rights as specified by configuration file which contains the Application Entity, the hostname it is running under and the port number it is listening to. Once the client name, its hostname and port number matches with the entry in configrc file, an association is built for further communication.

The DICOM Image server runs on default port no 104. It continuously listens to request made from DICOM clients.

Chapter 5

Implementation

The Web server is based on Unix operating system. Linux 6.2 [14] is used with Apache HTTPD WWW as the Web server [13]. CGI modules were developed in perl to extract the parameters specified by the user and to generate queries that are passed on to DICOM client. The perl program reads the output from DICOM server and customizes it. It then generates HTML pages on the fly, which are rendered to the user with the help of web browser. A DICOM plug in is installed in the browser that makes it possible to show images in DICOM format.

5.1 Configuration and Installation of components

Components of the system are mostly public domain software, which is downloaded from World Wide Web. All modules for Common Gateway Interface were written as part of the project.

5.1.1 DICOM Plug-in

The plug-in was downloaded from the site http://www.accusoft.com/dicom/examinet.html This has to be installed on the client in order to visualize images in DICOM format. The plug-in downloaded supports Internet Explorer. The following files are copied to Windows System directory:

IGMed81.dll - support DLL for ExamiNet
npEN32ps.HLP - help file
npEN32ps.CNT - used with help file

The plug-in enables the browser to render images in DICOM format with MIME-type image/DICOM or image/x-DICOM [15]. The image displayed can be manipulated using the functions provided by plug-in like window/level control, contrast inversion, rotate/flip, zoom-in/zoom-out etc. After viewing the images, the physician has a choice to store them in the local machine as DICOM files or as files whose format is supported by the plug-in. Selected images can be sent to printer to obtain a hardcopy.

5.1.2 Apache Server

Apache server is distributed with Linux operating system. A binary distribution of Apache will supply the file, httpd in the src directory.

The next step is to install the program and configure it.. Apache is designed to be configured and run from the same set of directories where it is compiled. If desired, it can be made to run from somewhere else by making a directory and copying the conf, logs and icons directories into it.

The next step is to edit the configuration files for the server. This consists of setting up various directives in up to three central configuration files. By default, these files are located in the conf directory and are called srm.conf, access.conf and httpd.conf [13]. There are same files in the conf directory of the distribution, called srm.conf-dist, access.conf-dist and httpd.conf-dist. These files should be copied or renamed without the -dist. Then each file has to be edited. Comments are provided in these files to help set up the files. Failure to setup these files correctly could lead to the server not working or being insecure. An additional file in the conf directory called mime.types should be present.

The httpd.conf is edited to set up general attributes about the server: the port number, the user it runs as, *etc.* Next the srm.conf file is edited; this sets up the root of the document tree, special functions like server-parsed HTML or internal imagemap parsing, *etc.* Finally, the access.conf file is edited to at least set the base cases of access.

In addition to these three files, the server behavior can be configured on a directory-by-directory basis by using .htaccess files in directories accessed by the server.

5.1.3 DICOM Toolkit

The DCMTK was downloaded as gzip compressed tar archive from http://www.offis.uni-oldenburg.de/projekte/dicom/soft-docs/soft01_e.html

Compilation
All the neccessary configure scripts are included in this distribution. The configure scripts examine the system capabilities and automatically generate include files and Makefiles.

The following steps from the top-level (dcmtk) directory were performed to compile and install the software [10]:

Step 1:

```
cd config
./rootconf
cd ..
```

Step 1 generates a Makefile and a configure script in the top-level dcmtk directory.

Step 2:

```
./configure
```

Step 2 executes the configure scripts in each subdirectory. First the system capabilities are examined and then Makefiles are generated. By default, executables and libraries will be installed (in Step 4) in the directory /usr/local/dicom in the bin and lib subdirectories. If another install prefix is required, the --prefix=<path> flag to configure has to used. e.g., to install underneath home directory in ~/dicom/bin and ~/dicom/lib, the configure should be started as:

```
./configure --prefix=$HOME/dicom
```

Step 3:

```
make all
```

Step 3 will build the libraries and executables.

Step 4:

```
make install
```

Step 4 will install the executables and data dictionary.
If the libraries and include files needs to be installed then "make install-lib" should be used.

Step 5:

```
make distclean
```

Step 5 will revert the source tree to the state prior to Step 1. If the object files and local executables should be discarded "make clean" is used instead.

5.1.4 DCMTK Client

The client consists of storescu, findscu and movescu sub-modules. Storescu is usually used by an imaging modality to store images generated by it, into an image archive. The operator of the scanner selects a menu option "Send selected images to STA_STORE" or words to that effect, and DICOM images are transferred to the archive. In order for this to work, two things have to be accomplished. First, the scanner has to be configured to have that menu item, and to send the selected data specifically to STA_STORE when it is selected. Typing the IP number and a service port number into a configuration file, connecting that configuration entry to the menu button, does this. The archive should be capable of accepting the DICOM images, which is done by running a program which listens on the above mentioned service port number for DICOM image storage service requests, and services those requests by storing the images onto disk. The storescp sub-module of DICOM toolkit does this.

Once storescp is up and running, it listens for incoming storage requests and stores the images in the directory from which it was started (or a subdirectory named CT or MR, which it will create).

5.1.5 Image Server

Imagectn of the DICOM toolkit acts as image server. It establishes communication with a DICOM client. Once the client is authenticated the server accepts the query from client, which is parsed and the parameters passed are matched with those available in the archive. Patient and study details of any matched parameter is displayed on standard output. Imagectn is configured to run on the server and listen to default port no 104.

5.1.6 Osiris

Osiris is portable and expandable software for interactive display and manipulation of medical images from different modalities. It was developed at The University Hospital of Geneva, and implemented on different hardware platforms. Three major operating systems are supported [16]:

1) Unix-based workstations with X-11 system enhanced by OSF-Motif widgets,
2) Apple Macintosh workstations (68K and PowerPC based),
3) PC running Microsoft Windows (Windows 3.11, Windows 95 and Windows NT).

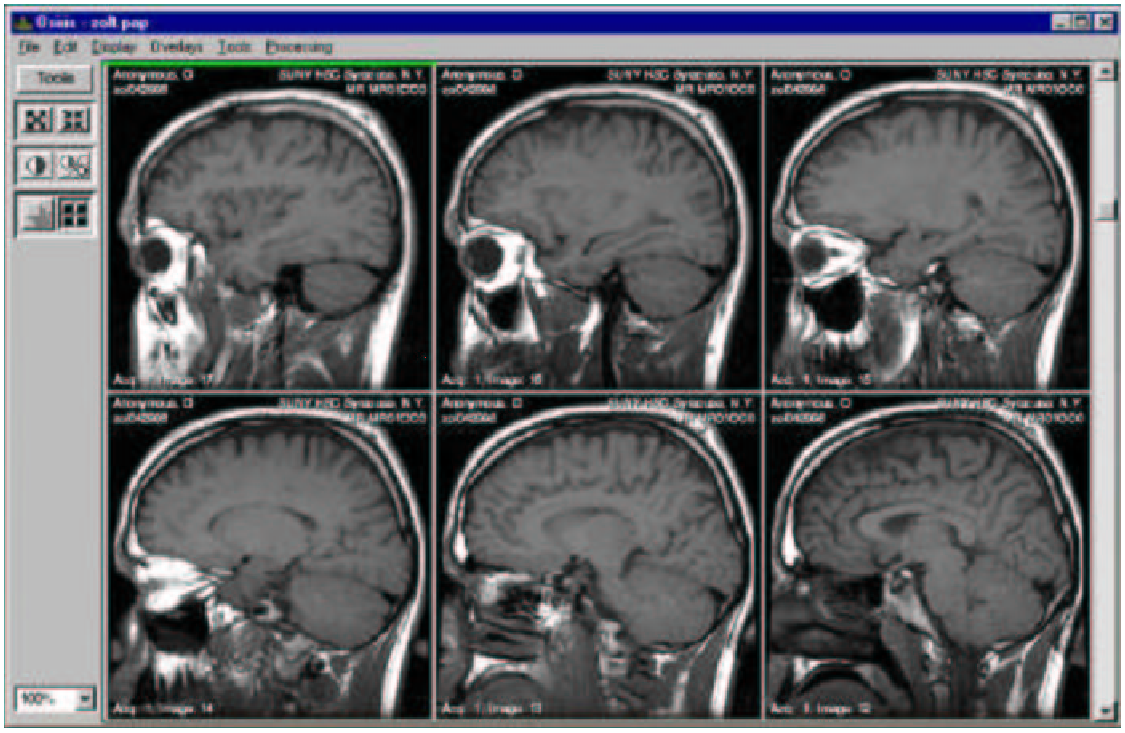


Figure 10: **Screenshot of Osiris, a DICOM viewer.**

Installation

The installation of the OSIRIS software is specific to each type of workstation:

Macintosh:OSIRIS software as well as the OsirisPrefs folder should be copied on onto the hard disk. These two items must be in the same directory. The

application is started by double clicking on the OSIRIS icon. System 7 or higher is required. 8 MBytes RAM are required but it is recommended to have at least 16 MBytes of RAM.

Windows: Setup.exe program included on the distribution should be run to install Osiris on the hard disk. It is recommended to have 16 MBytes of RAM, a Pentium processor and Windows 95 or Windows NT, and a multicolor display driver (256 colors are required).

OSIRIS is able to read raw data files and files containing a single DICOM image. Version 3.0 of Osiris is also able to read the DICOM extension for cardiologz D.I.S.C. (DICOM Imaging Standard for Cardiology) on recordable CD ROMs. This includes reading DICOMDIR files containing the directory of DICOM files recorded on the CDR. The Macintosh version has also the ability to read PICT file format, while the PC/Windows version can read TIFF images.

5.2 Server Side Modules

Programs on the server are executed when the HTTP server captures information from the client and passes this to CGI modules. The CGI module program accepts the information and parses it to obtain each query parameter. These parameters are used to construct a query, invoke DICOM client and pass the query.

5.2.1 CGI Modules

The URL is typed in the browser, which executes accessForm.html. This forms prompts the user for authentication. After the user authenticates with valid username and password, a patient query form (queryForm.html) is displayed on the browser.

The flow of server program execution according to user input is as follows

User keys in Patient attributes
    executes patientQuery.cgi
Selects a patient
    executes patientStudies.cgi
Keys in Study attributes

```
        executes studyQuery.cgi
Selects a study
        executes moveStudy.cgi and showImage.cgi
Selects a series
        executes moveSeries.cgi
        executes showImage.cgi
Selects Image or Quits
        executes showImage.cgi or quit.cgi and queryForm.html
```

5.2.2 queryForm

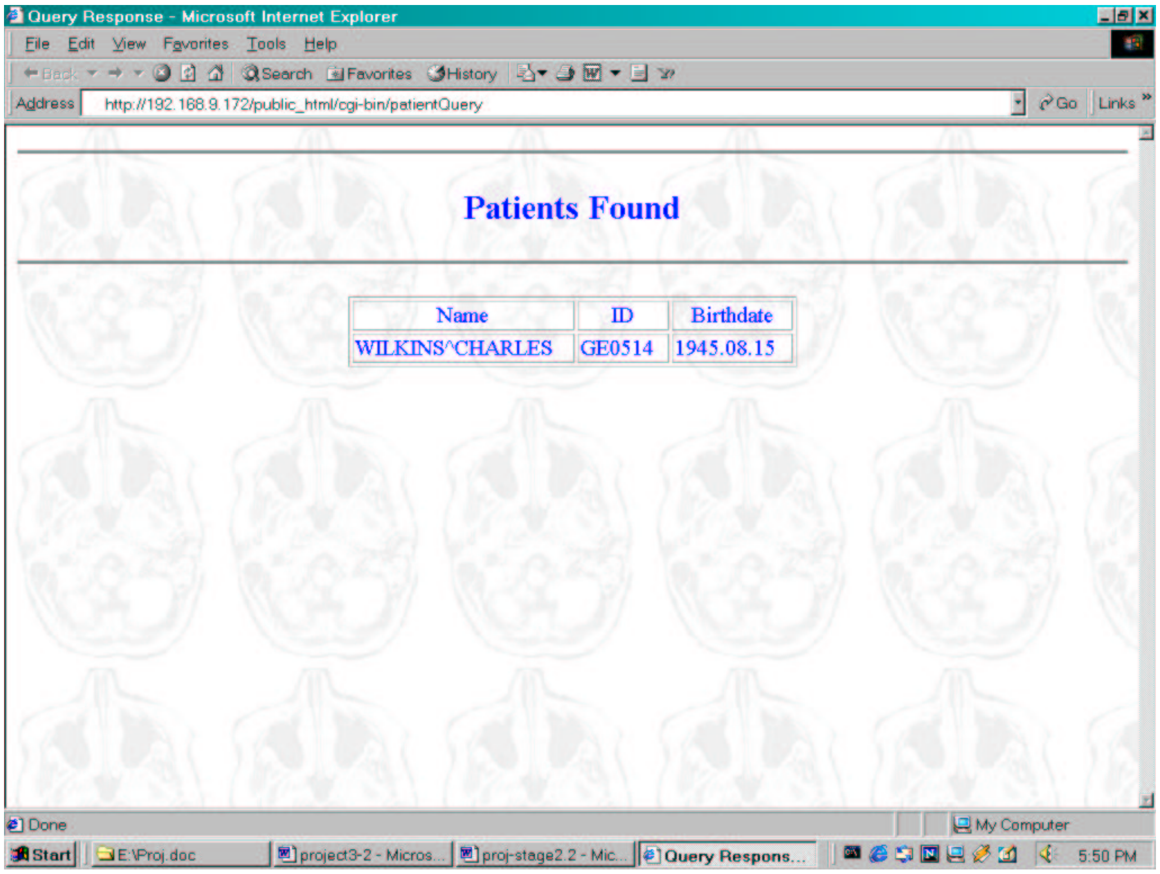
queryForm.html is a HTML form with two set of fields that allow the user to specify patient or study attributes respectively. The patient name and ID fields support the use of wildcards. Each form has a submit button and clear button.

Part of HTML code for this form is as follows [17]:

```
<FORM METHOD="POST"
    ACTION="http://192.168.9.172/~sheikh/public_html/cgi-bin">
    Patient ID          <INPUT NAME="patient_id">
    Patient name        <INPUT NAME="patient_name">
    .
    .
    .
    <INPUT TYPE="submit" VALUE="search">
 </FORM>
```

5.2.3 patientQuery

When the "search" button is pressed, the server executes the patientQuery.cgi program. This program is stored in the cgi-bin subdirectory on the machine. To insure that programs are called in the correct manner, the Web server sets an environment variable, REQUEST_METHOD, to the value `POST'. The archive-request program verifies this and then retrieves the values that were specified by the user in the form fields. To accomplish this, the server sets an environment variable, CONTENT_LENGTH [17], to be the number of arguments/values. The archive-request program from standard input reads the actual values. Arguments are passed as name/value pairs of the form name1=val1&name2=val2&...&namen=valn& where namen is the form field name and valn is the value that the user specified that is associated with namen. Once the arguments have been parsed by patientQuery, a query is generated and a findscu program is invoked with patient root query/retrieve model, which queries the DICOM archive and outputs a list of those patients that match the query criteria specified on the form. The program is described in detail below. Each line that this program outputs is of the form "patientName^patientID^birthDate^". The output is redirected to query_result_patient file.


Query Response - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://192.168.9.172/public_html/cgi-bin/patientQuery
Go Links
Done
My Computer
Start E:\Proj.doc project3-2 - Micros... proj-stage2.2 - Mic... Query Respons... 5:50 PM

Patients Found

Name	ID	Birthdate
WILKINS^CHARLES	GE0514	1945.08.15

Figure 11: **Screen showing a list of patients that was obtained by the Web browser after querying the archive.**

The pattientQuery program then reads this temporary file and writes HTML code to standard output in the form of one query URL for each patient ID. For example, if the output is "WILKINS^GE0514^19450815^" then patientQuery outputs the corresponding query URL as the following:

```
<a href="http://192.168.9.172/~sheikh/cgi-bin/patientStudies.cgi?Name =$
patientName;id=$patientID;">patientID</a>
```

The server then interprets this HTML code and the user's browser displays it. The user is shown the patient ID as a hypertext link and the patient's name and birthdate. If the user clicks on the patient ID then patientStudies will be invoked.

5.2.4 findscu with patient root model

Given the input arguments of patient ID and patient name, this program queries the archive and lists those patients that matched the query criteria to standard output. Patient Root Query/Retrieve level is used in querying.

Each output line is of the form `patientName^patientID^birthDate^'. This program is built on the DCMTK implementation of DICOM by the university of Oldenberg, OFFIS, Germany. It requests to be a user of the service class of DICOM_SOPPATIENTQUERY_FIND, requests an association be made between DICOM client process and DICOM server process, and then proceeds with a C-FIND to determine all of those patients that match. Each time a patient matches, a callback routine is called which builds a list of matches. (Callback routines are the method by which set oriented data are handled. To retrieve a set of data, the user specifies the address of a function that will be called repeatedly for each element of the set with arguments that indicate the specific element.) When all matching patients have been added to the list (representing the set), this program writes the contents of the list to standard output, which is redirected to query_result_patient file.

5.2.5 patientStudies

When the user clicks on a patient ID whose study and series information is required, the server executes the patientStudies.cgi program. This program is stored in the cgi-bin subdirectory on the machine. To insure that programs are called in the correct manner, the Web server sets an environment variable, REQUEST_METHOD, to the value `GET. The parser verifies this and then retrieves the values that were specified by the user. To accomplish this, the server sets an environment variable, QUERY_STRING, to be the number of arguments/values.

The parser program from standard input reads the actual values. Arguments are passed as name/value pairs of the form name1=val1&name2=val2&.. &namen=valn& where namen is the form field name and valn is the value that the user specified that is associated with namen. Once the arguments have been parsed, a query is generated and a findscu program is invoked with study root query/retrieve model, which queries the DICOM archive and outputs a list of studies and series that match the query criteria specified. The output is redirected to query_result_study file.

The patientStudies program then reads this temporary file and writes HTML code to standard output in the form of one query URL for each patient.

5.2.6 findscu with study root model

Given the input argument of patient ID, this program queries the archive and lists all the patient's studies and their corresponding series information to standard output. Each output line is of the following form:

patientName^patientID^studyID^studyInstanceUID^studyDate^SeriesInstan eUID1 ^seriesModality1^...^SeriesModalityn^SeriesInstanceUIDn^

This program requests to be a user of the service class of DICOM_SOPSTUDYQUERY_FIND, requests an association be made between DICOM client process and DICOM server process, and then proceeds with a C-FIND to determine all of those studies that are archived for the specified patient. Each time a study is found, a callback routine is called which builds a list of studies. When all studies have been added to the list, the program is invoked again by patientStudies with Series query/retrieve model. This repeats the above process for each study to determine all of the series associated with each study via another C-FIND. Once again a callback routine is specified which builds a list of series for the given study. This program writes the contents of the study and series lists to standard output, which is redirected to file query_result_study.

5.2.7 moveStudy

moveStudy.cgi is invoked when the user clicks on the hypertext link for study, when study and series information is presented. This program is also stored in the cgi-bin subdirectory on the machine that is our WWW server. To insure that programs are called in the correct manner, the Web server sets an environment variable, REQUEST_METHOD, to the value `GET. This program verifies this and then retrieves the query values by getting the environment variable, QUERY_STRING, which is of the form "patientID=$patientID&studyUID=$studyUID".

Once these arguments have been parsed, moveStudy.cgi causes the study to be retrieved from the archive. Then writes HTML code to standard output that allows the user to access all of the images in the study.

5.2.8 moveSeries

This program is invoked when the user clicks on the hypertext link for the series when study and series information is presented. This program is also stored in the cgi-bin subdirectory. The environment variable is set and query values obtained as mentioned in moveStudy. After parsing QUERY_STRING, which is of the form `patientID=$patientID&studyUID=$studyUID&series UID=$seriesUID' , movescu is invoked to cause the series to be retrieved from the archive. Then moveSeries writes HTML code to standard output that allows the user to access all of the images in the series via the browser.

5.2.9 ShowImage.cgi

This program reads image files stored on web server. For each image read, it generates an HTML file in which the image is embedded.

Embedding of DICOM image in HTML page (For Internet Explorer)

```
<object classid="clsid:1924751C-085A-11D2-85AF-00A0C96B3CB7"
width="970" height="550">
<param name="Src" value="su_ctr.dcm">
</object>
```

5.3 Security

Access to confidential patient information is restricted via a method that is built into the NCSA HTTPD WWW server. This allows or restricts access at the domain and/or IP address level in addition to user/password level access. We are currently using IP address security to restrict or access specific hosts within the campus network.

Before an HTML document for querying can be accessed, the server checks for the presence of a file named. .htaccess in the directory [13] where the HTML document is located. If this file exists, access will be restricted according to the contents of this file. The following is an example. htaccess file.

```
<Limit GET>
order deny,allow
deny from all
allow from 144.16.106
allow from bme.iitb.ernet.in
.
</Limit>
```

Example file for restricting access.

The first line indicates that we wish to restrict access. The second line indicates that we will process deny entries first and allow entries second. The next line indicates that we wish to disallow access for everyone. Then the following allow lines list those that we wish to allow.

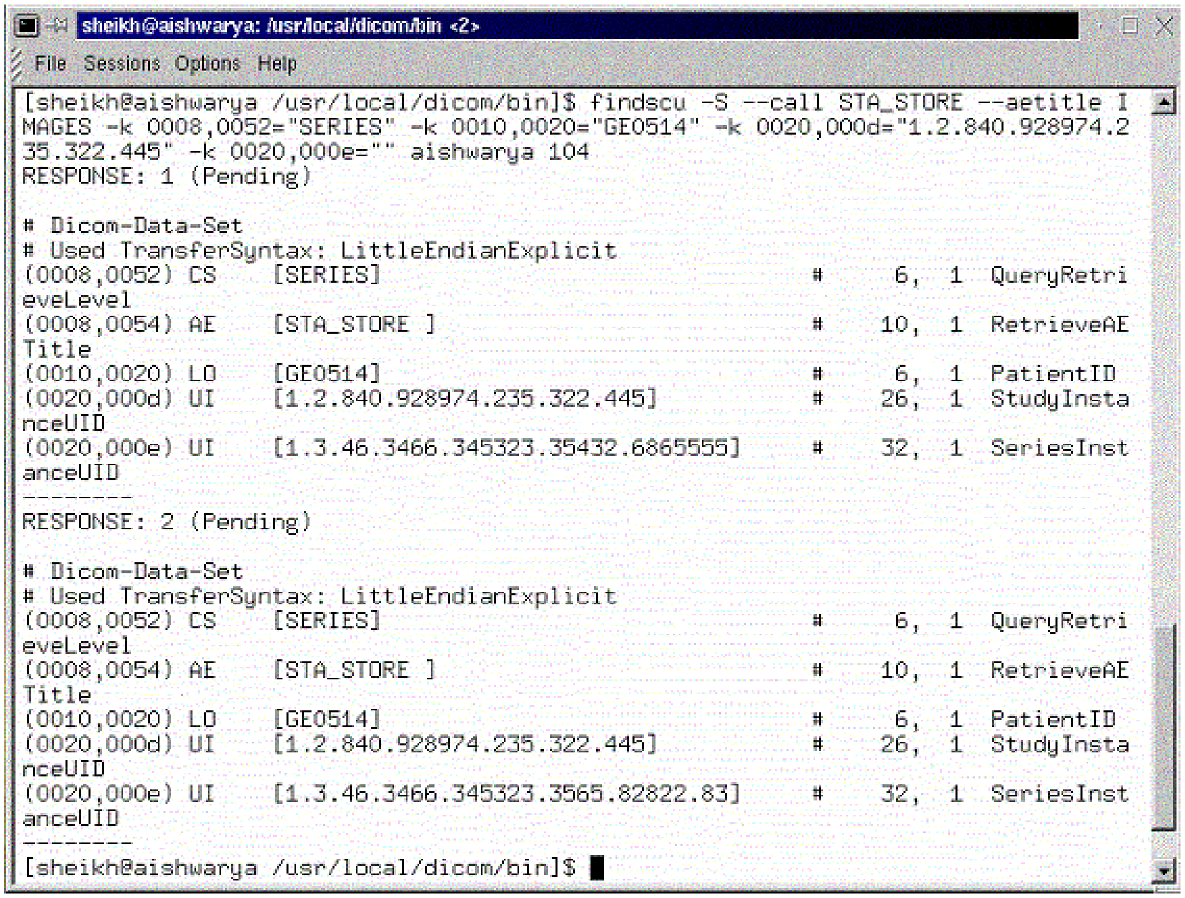
Access to the Web server is restricted, with security implementation, only authorized users, identified by username and password, can use the system.

Chapter 6

Experiences with DicomAccess

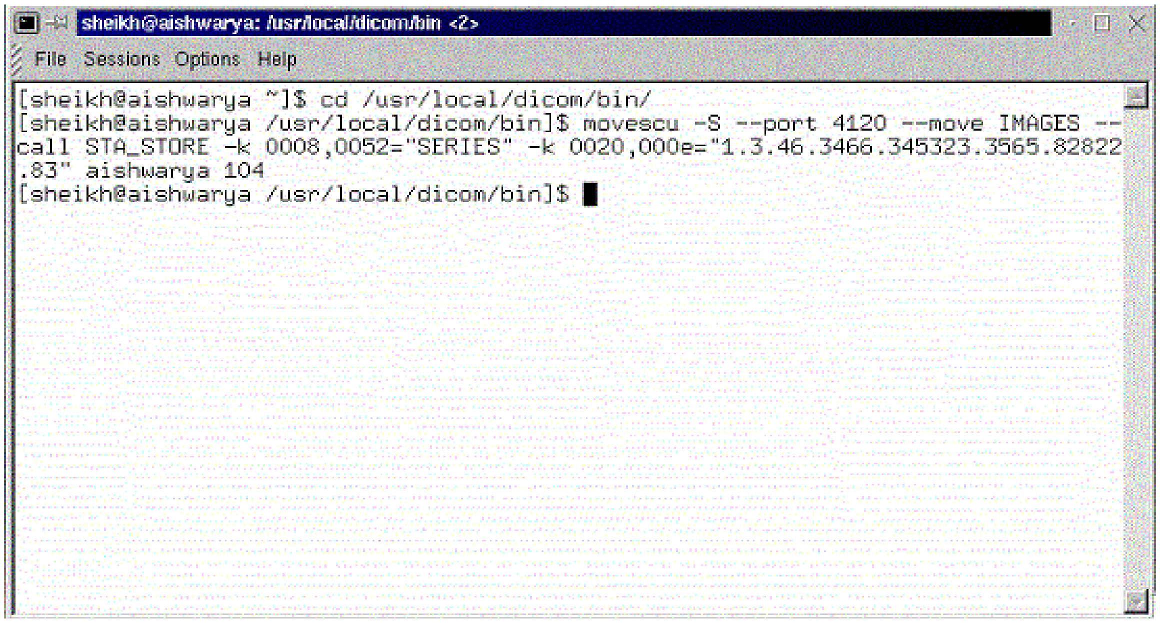
The implementation was tested with several queries relating to patient and study parameters. Command mode operation using DCMTK client and the result obtained is shown in figure 12 and 13 for comparison. The ease with which querying can be done is shown in following figures. By using several selection criteria and a Web browser, the user can request a specific list of patients (Fig 8). The Web server invokes CGI modules to generate queries and pass these to DICOM client, which queries DICOM database and obtains the list of patients. This is converted to an HTML page, which is then sent back to the user's Web browser (Fig 11).

The user can select any of the patients on the list, and after the Web server invokes another module for query generation and sends it to the DICOM client, the list of studies and series for the selected patient is displayed in the Web browser (Fig 14). Afterward, the user need only select one of the studies or series, and the appropriate images are retrieved from the DICOM database by the Web server and displayed in the Web browser with the help of DICOM plug-in (Fig 15). Any Web browser that supports DICOM plug-ins can be used (e.g., Internet Explorer, Netscape Communicator) but the plug-in currently installed supports Internet Explorer for image visualization.


sheikh@aishwarya: /usr/local/dicom/bin <2>
File Sessions Options Help

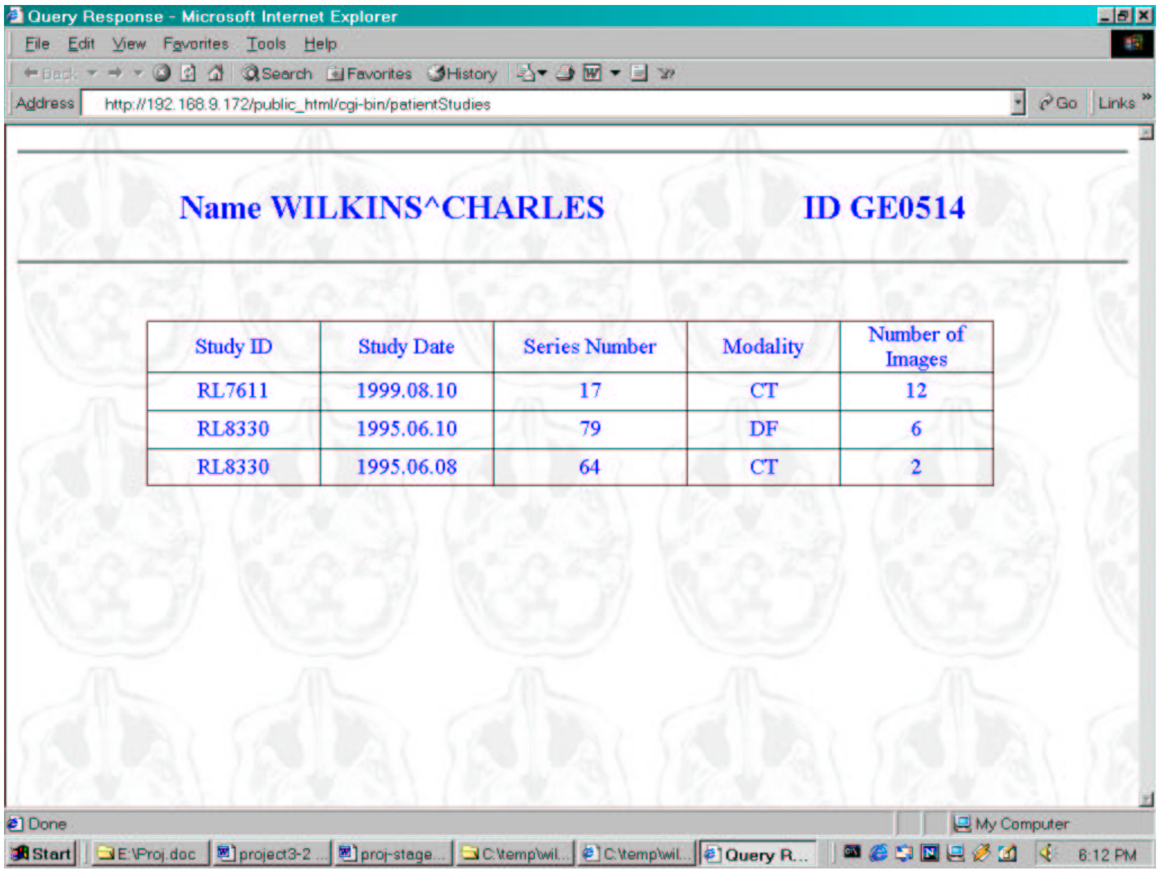
```
[sheikh@aishwarya /usr/local/dicom/bin]$ findscu -S --call STA_STORE --aetitle I
MAGES -k 0008,0052="SERIES" -k 0010,0020="GE0514" -k 0020,000d="1.2.840.928974.2
35.322.445" -k 0020,000e="" aishwarya 104
RESPONSE: 1 (Pending)

# Dicom-Data-Set
# Used TransferSyntax: LittleEndianExplicit
(0008,0052) CS    [SERIES]                              #     6,  1  QueryRetri
eveLevel
(0008,0054) AE    [STA_STORE ]                          #    10,  1  RetrieveAE
Title
(0010,0020) LO    [GE0514]                              #     6,  1  PatientID
(0020,000d) UI    [1.2.840.928974.235.322.445]          #    26,  1  StudyInsta
nceUID
(0020,000e) UI    [1.3.46.3466.345323.35432.6865555]    #    32,  1  SeriesInst
anceUID
--------
RESPONSE: 2 (Pending)

# Dicom-Data-Set
# Used TransferSyntax: LittleEndianExplicit
(0008,0052) CS    [SERIES]                              #     6,  1  QueryRetri
eveLevel
(0008,0054) AE    [STA_STORE ]                          #    10,  1  RetrieveAE
Title
(0010,0020) LO    [GE0514]                              #     6,  1  PatientID
(0020,000d) UI    [1.2.840.928974.235.322.445]          #    26,  1  StudyInsta
nceUID
(0020,000e) UI    [1.3.46.3466.345323.3565.82822.83]    #    32,  1  SeriesInst
anceUID
--------
[sheikh@aishwarya /usr/local/dicom/bin]$ █
```

Figure 12: **Query and result of study details, using findscu (DCMTK)**


sheikh@aishwarya: /usr/local/dicom/bin <2>
File Sessions Options Help

```
[sheikh@aishwarya ~]$ cd /usr/local/dicom/bin/
[sheikh@aishwarya /usr/local/dicom/bin]$ movescu -S --port 4120 --move IMAGES --
call STA_STORE -k 0008,0052="SERIES" -k 0020,000e="1.3.46.3466.345323.3565.82822
.83" aishwarya 104
[sheikh@aishwarya /usr/local/dicom/bin]$ █
```

Figure 13: **Usage of movescu command (DCMTK)**



Query Response - Microsoft Internet Explorer

http://192.168.9.172/public_html/cgi-bin/patientStudies

Name WILKINS^CHARLES ID GE0514

Study ID	Study Date	Series Number	Modality	Number of Images
RL7611	1999.08.10	17	CT	12
RL8330	1995.06.10	79	DF	6
RL8330	1995.06.08	64	CT	2

Figure 14: **Screen shows the list of studies for the patient chosen from the list in Figure 11. The list was obtained after the client queried DICOM archive for a second time.**

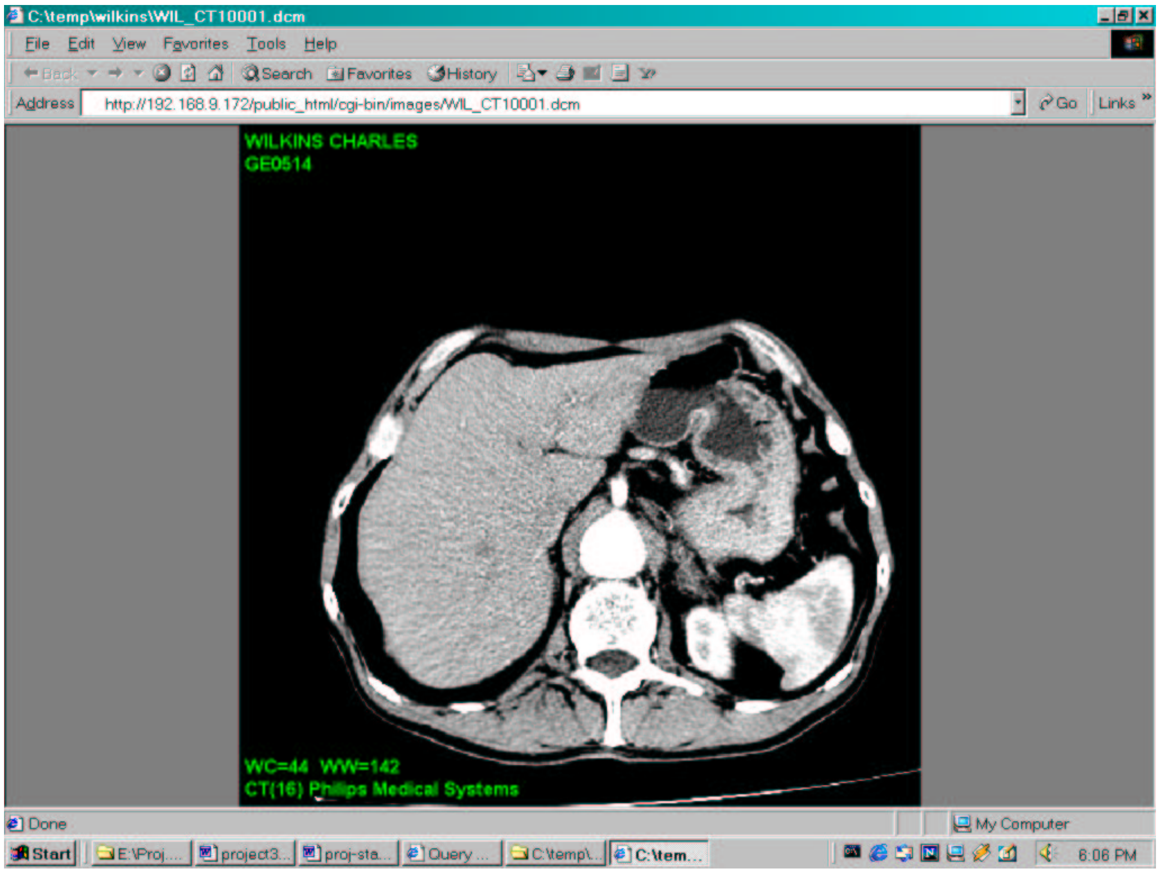

C:\temp\wilkins\WIL_CT10001.dcm
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://192.168.9.172/public_html/cgi-bin/images/WIL_CT10001.dcm
Go Links
WILKINS CHARLES
GE0514
WC=44 WW=142
CT(16) Philips Medical Systems
Done
My Computer
Start E:\Proj... project3... proj-sta... Query ... C:\temp\... C:\tem...
6:08 PM

Figure 15: **Screen showing a CT scan displayed in a web browser with the help of DICOM plug-in.**

The WWW interface has made querying DICOM archive and image visualization simpler as most of the users would be familiar with the usage of web browsers.

Chapter 7

Conclusion

The components of DicomAccess provide web based access to DICOM archive and allow for the querying of patient, study and image information with the option to display the images on a web browser or download the images and view them with DICOM viewer. Wherever possible, freely available software is used.

DicomAccess allows querying and retrieval of information stored in DICOM archive. It caters to the needs of Radiologists and Referring Physicians for querying and viewing images of a particular patient. Further, the physician can select a particular study or series of the patient and has a choice to view these images with a web browser or a DICOM viewer that provides more image manipulation functions. Access to the web server is restricted to authorized users and from PCs that have permission to access the web server. The authorized user is identified by username and password.

7.1 Future Work

The physician is presented with hypertext links to studies and images. If thumbnails of actual images can be displayed instead of hypertext links, it would make it easier for the physician to decide which images are required and only those images can be requested for transmission rather than whole study or series. This way, the network traffic would be reduced. Its importance is appreciated in networks having low bandwidth.

Other functionality includes an HTML interface to the report information itself. This would allow the physicians to query the report, patient, and exam information for specific values or in the case of the reports, keywords. A list of reports that matched the search criteria could then be displayed. Then the actual report, exam, or patient information or the images themselves could be retrieved [18]. This requires an interface of PACS with Radiology Information System (RIS).

Images can be converted to formats like JPEG or PNG that require around one tenth the space needed for DICOM images without significant loss in diagnostic quality [19]. These images can be used for teaching radiology residents, medical students and other personnel involved in image reading. The images may be archived or generated online

References

1. Wu TC, Lee SK, Peng CH, Wen CH, Huang SK. An economical, personal computer–based picture archiving and communication system. RadioGraphics 1999; 19:523-530.

2. Josep Fernàndez-Bayó, Octavio Barbero, Carles Rubies, Melcior Sentís and Lluis Donoso. Distributing Medical Images with Internet Technologies: A DICOM Web Server and a DICOM Java Viewer. *Radiographics.* 2000;20:581-590.

3. Yoad Gidron, Uri Shani, Mark Shifrin. Phased development of a web-based PACS viewer. IBM Haifa Research Lab, MATAM, Haifa 31905, Israel.

4. Official web site of DICOM - http://medical.nema.org

5. DICOM Standard version 3.0, Part 2: Conformance Statement

6. DICOM Standard version 3.0, Part 3: Information Object Definitions

7. DICOM Standard version 3.0, Part 4: Service Class Specifications

8. Central Test Node
   http://wuerlim.wustl.edu/pub/dicom/images/version3/RSNA95

9. OFFIS, University of Oldenburg
   http://www.offis.uni-oldenburg.de/projekte/dicom/

10. Documentation available with DICOM Toolkit:
    http://www.offis.uni-oldenburg.de/projekte/dicom/

11. San-Kan Lee, Chen-Hsing Peng, Chia-Hsien Wen, Shu-Kun Huang and Wen-Zhong Jiang. Consulting with Radiologists outside the Hospital by Using Java. *Radiographics.* 1999;19:1069-1075.

12. Jean-Chrétien Oberson, Ronald Welz, and Laurent Bovisi. Development of an Electronic Radiologist's Office in a Private Institute. *Radiographics.* 2000;20:573-580.

13. The Apache Software Foundation: http://www.apache.org

14. The Linux Home Page at Linux Online: http://www.linux.org/info

15. Accusoft Corporation DICOM Plug-in: http://www.accusoft.com/dicom/dicom.html

16. Osiris DICOM Viewer: http://www.expasy.ch/www/UIN/html1/projects/osiris/osiris.html

17. T. Berners-Lee, R. Fielding and H. Frystyk. Hypertext Transfer Protocol -- HTTP/1.0, May, 1996: http://www.w3.org/Protocols/Specs.html

18. Steven J. Willing and Lincoln L. Berland. A Radiology Department Intranet: Development and Applications. *Radiographics.* 1999;19:169-182.

19. Smith HJ, Bakke SJ, Smevik B, et al. Comparison of 12-bit and 8-bit gray-scale resolution in MR imaging of the CNS: an ROC analysis. Acta Radiologica 1992; 33:505 511.

20. Engelmann U, Schroeter A, Schwab M, Eisenmann U, Vetter M. Lorenz K, Quiles J, Wolf I, Evers H, Meinzer H.P. Borderless Teleradiology with CHILI. Journal of Medical Internet Research 1999;1(2).

21. CD-Based Image Archival and Management on a Hybrid Radiology Intranet, R.D. Cox, C.J. Henri, R.K. Rubin and P.M. Bret, *Journal of Digital Imaging Vol. 10(2): 168-170 Suppl 1, May 1997*

22. Implementation of a filmless mini picture archiving and communication system in ultrasonography experience after one year of use, *C.J.* Henri, R.D. Cox and P.M. Bret, *Journal of Digital Imaging, Vol 0, Suppl 1, Aug., 997, pp 80-82.*

The URLs are as on 10th March 2001.