Distributed Computing Environment for Standards Based Multimedia Healthcare Systems

by

Yasser al-Safadi

A Dissertation Submitted to the Faculty of the

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

In Partial Fulfillment of the Requirements
For the Degree of

DOCTOR OF PHILOSOPHY

In the Graduate College

THF UNIVERSITY OF ARIZONA

1 9 9 5

Distributed Computing Environment for Standards Based Multimedia Healthcare Systems

by

Yasser al-Safadi

A Dissertation Submitted to the Faculty of the

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

In Partial Fulfillment of the Requirements
For the Degree of

DOCTOR OF PHILOSOPHY

In the Graduate College

THF UNIVERSITY OF ARIZONA

1 9 9 5

UMI Number: 9622976

UMI
300 North Zeeb Road
Ann Arbor, MI 48103

THE UNIVERSITY OF ARIZONA
GRADUATE COLLEGE

As members of the Final Examination Committee, we certify that we have read the dissertation prepared by Yasser Haycam Al-safadi

entitled Distributed Computing Environment for Standards Based Multimedia Health care Systems

and recommend that it be accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy

Ralph Martinez — 9/29/95
Dr. Ralph Martinez — Date

Douglas R. Dunn — 9/29/95
Date

Ken McNeill — 9/29/95
Date

Bob Zeigler — 9/29/95
Date

Date

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copy of the dissertation to the Graduate College.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it be accepted as fulfilling the dissertation requirement.

Ralph Martinez — 10/13/95
Dissertation Director — Date
Dr. Ralph Martinez

STATEMENT BY AUTHOR

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: ____________________

ACKNOWLEDGMENTS

I would like to thank Dr. Ralph Martinez for his constant support and encouragement. His vision in developing Global PACS systems lead me throughout this project. Additionally, I express appreciation to Dr. Kevin McNeil and Dr. Bill Dallas and my colleagues at the Radiology Research Lab their help and support.

Also, I want to thank Dr. Larry Peterson for enjoyable courses and Dr. Olivia Sheng who taught me database courses. I want to thank my wonderful teachers: Bernard Zeigler, Ahmed louri, Ming-Kang Liu, Richard Schlichting, and Mary Bailey. Finally, I want to thank Dr. Adela Allen, Associate Dean of Graduate College, who without her initial support I would not have been in graduate college.

To my parents, whose patience and kindness made it worthwhile,

I express my greatest appreciation.

TABLE OF CONTENTS

TABLE OF CONTENTS – *Continued*

TABLE OF CONTENTS – *Continued*

LIST OF FIGURES

LIST OF FIGURES – *Continued*

LIST OF FIGURES – *Continued*

LIST OF TABLES

ABSTRACT

The Open Software Foundation (OSF) Distributed Computing Environment (DCE) is an integrated set of services that facilitates the construction, use and maintenance of distributed applications in a heterogeneous computing environment. The OSF DCE services include remote procedure calls, naming service, threads service, time service, and security service. Several OSF DCE toolkits are currently available from computer and software vendors.

The Global Picture Archiving and Communication System (Global PACS) operates in a medical environment for managing digital images over a large geographical area. This dissertation presents an approach to developing a platform to support multimedia Global PACS applications using the OSF DCE services and toolkits. Dynamic sequences such as Ultrasound are retrieved from a scalable video service over a TCP/IP connection. The Comprehensive Chart and the Remote Consultation and Diagnosis system are multimedia Global PACS applications that demonstrate the utility of this approach.

The Comprehensive Chart is a multimedia medical record browser that provides a comprehensive view of patient data. The user of the Comprehensive Chart is authenticated using DCE Security and can access the objects only allowed by the Access Control List. System resources locations are transparent to the user and are located using the DCE Directory Service. Patient data privacy is maintained during communication through the use of secure remote procedure calls.

The Remote Consultation and Diagnosis system was developed under a National Science Foundation project headed by Dr. Ralph Martinez, University of Arizona. It allows medical experts at different geographical locations to view the same image and exchange synchronized voice and image annotation commands. The current version uses the DCE Directory

Service to dynamically locate session participants. These participants are authenticated and they can access objects only allowed by access control lists. The DCE Time Service will hide time zone differences among participants, and support the timestamp mechanism for the synchronization of voice and image annotation commands.

The use of the OSF DCE approach features an open architecture, heterogeneity, security, scalability, and technology independence. This approach can be used to develop general purpose multimedia delivery applications. Finally, this design and implementation provides the foundation for extending medical services to rural areas.

CHAPTER 1

Introduction

1.1 Statement of the Problem

A global picture archiving and communication system (Global PACS) operates in a medical environment for managing digital images over a large geographical area. This computing environment is highly a heterogeneous, as depicted in Figure1.1. This , multi-vendor environment is comprised of different hardware and software architectures on different computing platforms and communication networks. Besides this heterogeneity in the computing platform, there is heterogeneity in the information and imaging platform. Most medical systems are proprietary, making them isolated islands of data. However, there has been a push towards standardization of the interfaces to these systems. Most prominent are the Health Level 7 interface to textual based medical information systems, and DICOM interface for image based medical systems. In the computing arena the Distributed Computing environment is becoming a widely used middle that provides heterogeneity, scalability and security. There is a realization that no single vendor can provide the best possible solution in every area of Global PACS. Thus, there is a need to utilize a standards based approach to Global PACS design and development.

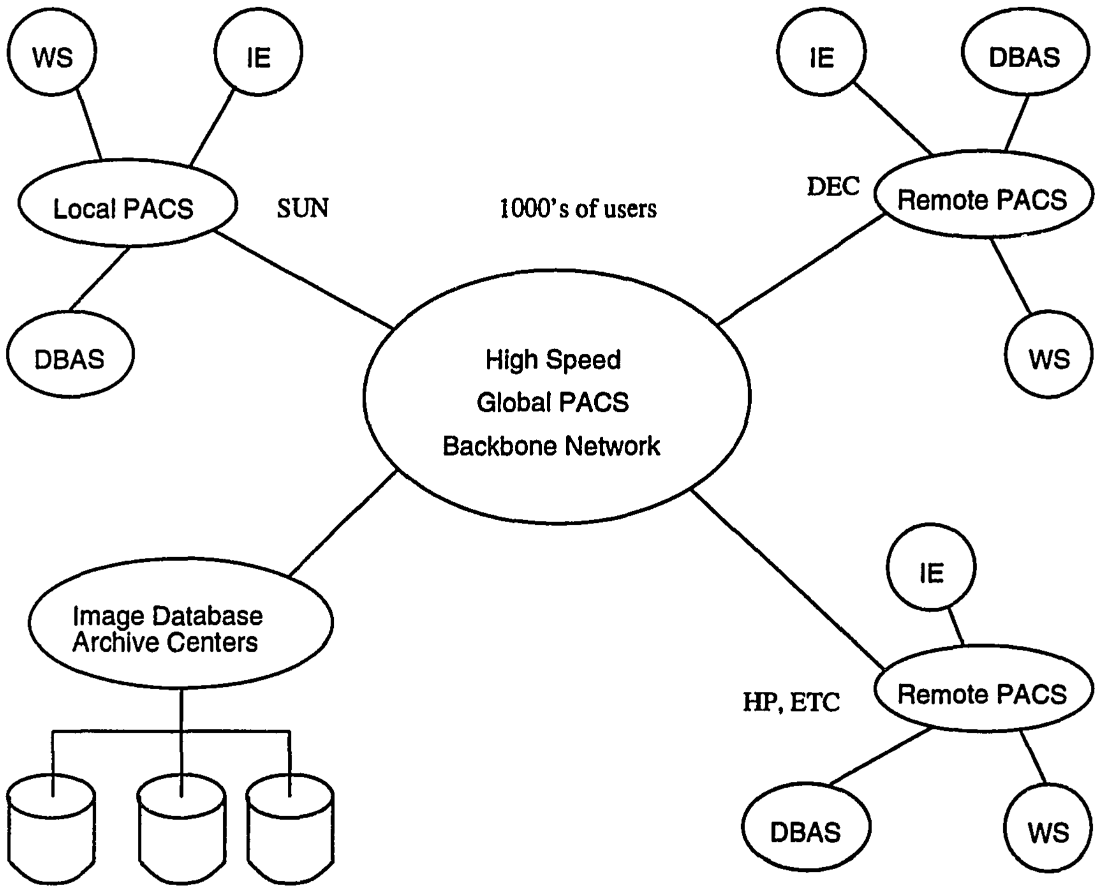


Figure 1.1: Distributed Environment of Global PACS

1.2 Objective

To develop a standards based interface to support the development of image based applications. More specifically, to develop the OSF DCE platform for a Global PACS environment. Global PACS is the complex distributed system used as a vehicle to demonstrate the use of DCE Services.

1.3 Approach

In this document, we present our approach to developing Global PACS applications using the Open Software Foundation Distributed Computing Environment (OSF DCE) services and toolkits. The OSF DCE services include remote procedure calls, naming service, threads service, time service, file management services, and security service. Several OSF DCE toolkits are currently available from computer and software vendors. To date, we have developed the remote consultation and diagnosis (RCD) as a Global PACS application using socket communication. Designing distributed Global PACS applications using the OSF DCE approach will feature an open architecture, heterogeneity, and technology independence for Global PACS remote consultation and diagnosis applications.

Our design approach revolves around the following points:

a. Standards based approach to systems and networks integration.

b. Scalable approach to distributed system design.

c. Uses commercially available workstation products and Unix-based operating systems.

d. Evolutional approach in introducing new technology in system design.

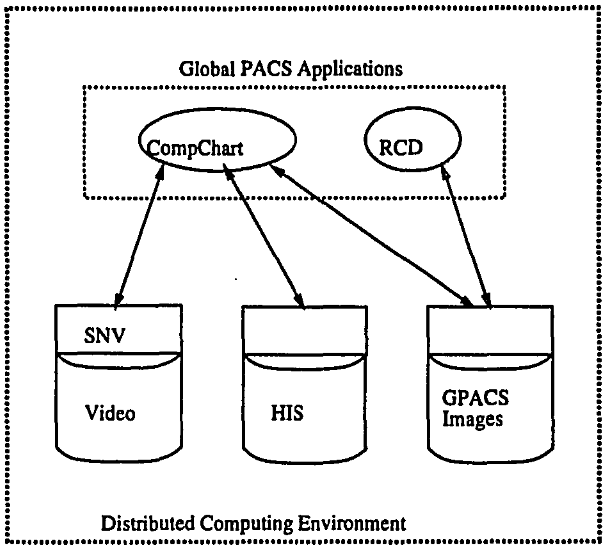


Figure 1.2: Components of System Design

The modular design consists of several components, as depicted in Figure 1.2.

a. a video interface to Scalable Video Server to retrieve dynamic sequences,

b. a Comprehensive Chart system, a Global PACS application that integrates the previous three components,

c. an OSF DCE mapping of the services to support the Global PACS applications,

Finally, the testbed demonstrates prototypes of OSF DCE mapped Global PACS Applications, namely the Comprehensive Chart and the RCD system.

1.4 Road Map to Document

Chapter 1, is the introduction. It provides a statement of the problem, the research objective and the design approach. Chapters 2, and 3 are introductory chapters. Each introduces an important component of the problem or the solution. Chapter 2, introduces the Global Picture Archiving and Communication System. This chapter demonstrates some user scenarios, and discusses security, privacy and confidentiality issues. It discusses Global PACS distributed computing issues A mapping diagram of these issues and DCE based solutions is depicted in Figure 1.3. The chapter then characterizes medical data, and introduces the Remote Consultation and Diagnosis System. Chapter 3, provides an overview of a fundamental element of our design, namely the Distributed Computing Environment. This chapter describes the Distributed Computing Environment architecture and its different technology components. Chapter 4, presents the system design which consists of several components. It contains the design of a Scalable Network Video delivery mechanism. These design components are integrated in the design of an exemplary Global PACS application, the Comprehensive Chart. It provides an interface to access and visualize a complete repository of all data relevant to the health of a patient. Then the chapter defines the mapping of the Distributed Computing Environment services to Global Picture Archiving and Communication System environment. Chapter 5, this mapping is implemented at two testbedsused at the University of Arizona. One testbed is for the Comprehensive Chart and the other is for the RCD system. This chapter presents these testbeds and the scalable network video experiments, and analysis of the results. Chapter 6, presents the conclusion of this research and identifies future research directions. A road map to this document is depicted in Figure 1.4.

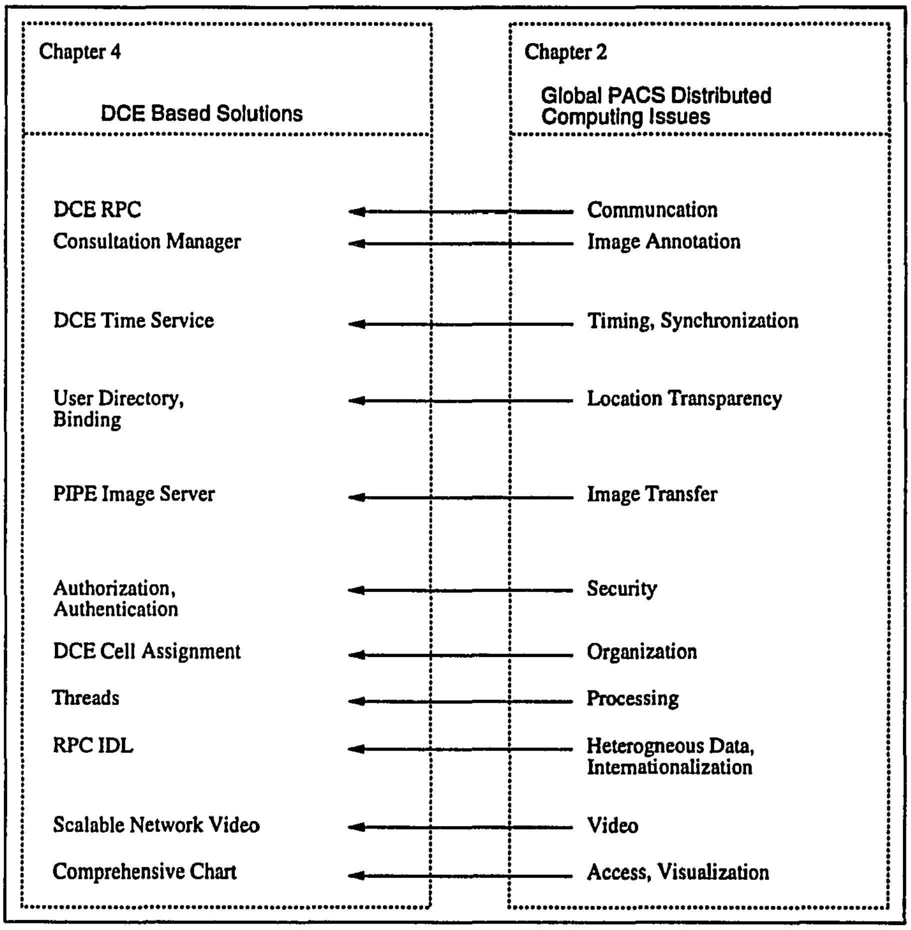


Figure 1.3: Mapping of Global PACS Distributed Computing Issues

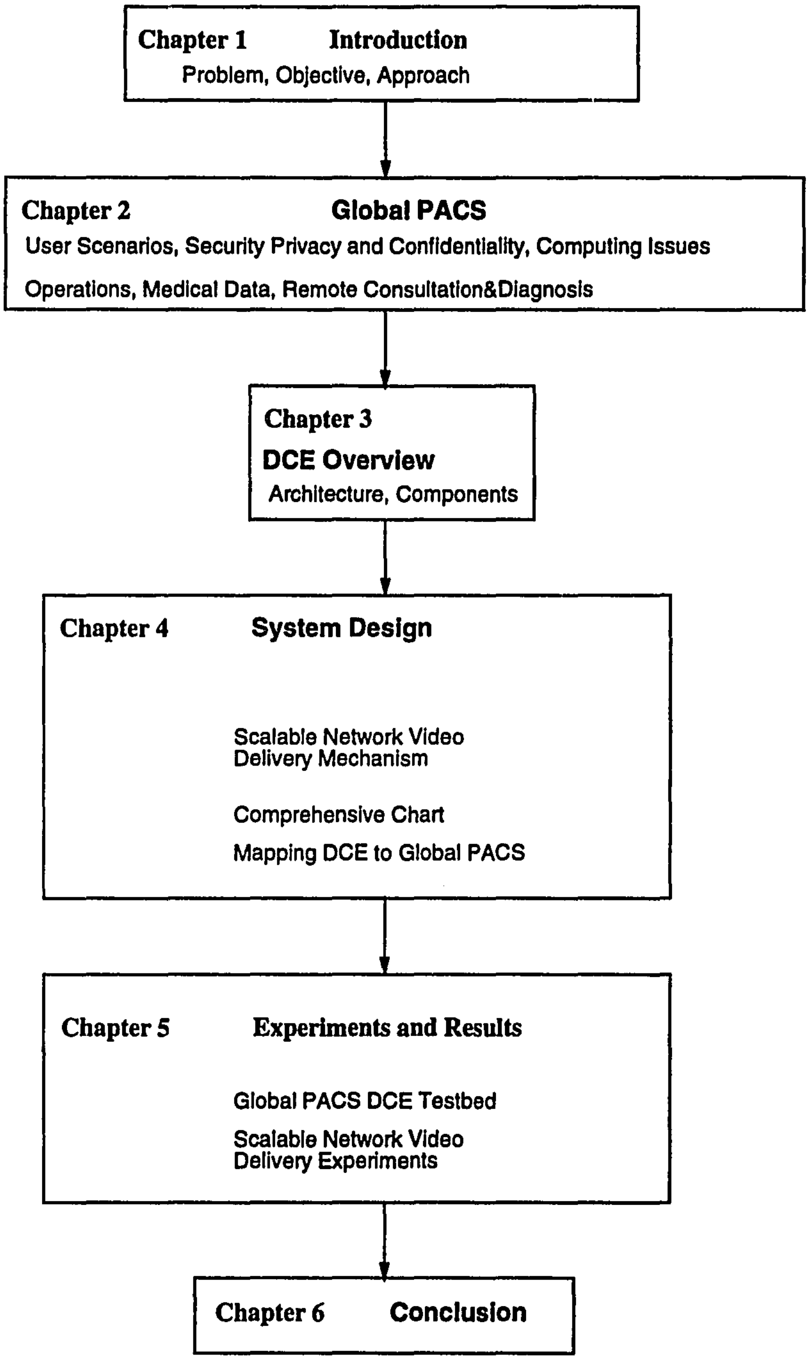


Figure 1.4: Dissertation Road Map

CHAPTER 2

Global Picture Archiving and Communication System

A picture archiving and communication system (PACS) operates in a medical environment for managing digital images [1]. The components include: imaging equipment for acquiring the images, workstations for viewing the images, database archive system for storage and retrieval of images, and communication network for connecting all of these components into an integrated system. Global PACS can provide teleradiology and telepathology services to remote geographical locations. Moreover, it will allow interactive consultation among experts located at different sites. This facility will also provide easy access to image archives for research and education.

This chapter will provide some example user scenarios. Then it characterizes medical data, and estimates network bandwidth and storage requirements for using this data. Finally, it briefly describes the Remote consultation and Diagnosis system.

2.1 User Scenarios

This section introduces scenarios of Global PACS users. These scenarios are user location independent, i.e. no distinction is made whether the user is by the bedside, in the office, or in a classroom. These scenarios are based on existing practices in hospital departments. For further analysis on user scenarios the reader is referred to [2].

a. Rural image storage and retrieval: A Rural and Global PACS provides its services in real-time and non real-time. Images transferred are usually provided in non real-time, requested prior to the consultation session. In this scenario, medical images and patient information are transferred to the destination within minutes. This is an example of non real-time image transfer for storage or retrieval. Requests for image storage and retrieval occurs off-line and images are exchange between an imaging equipment or workstation on a Local PACS and the database archive system on a Remote PACS. Local PACS users have access to an image directory service for image location. This scenario is used to retrieve image sets for research, education and diagnosis.

b. Rural remote consultation and diagnosis: The rural remote consultation scenario involves two or more medical experts each located respectively at a Local and Remote (rural) viewing workstations. These workstations can be located anywhere in the country.a The physicians establish a consultation session for the purpose of examining images and recording a diagnosis. They both see the same image on their workstation screen. Each viewing workstation provides a pointer that allows the physician to point to parts of the image. When the physician moves the pointer, the Remote site sees the resulting movement of the pointer on their display. Additionally, workstations have image processing capabilities that allow the physicians to manipulate and analyze the image. These features are also displayed on the other consoles at the same time as the originating workstation. Voice interaction between the two physicians takes place interactively. In order for all this to take place in real-time, only the pointing commands, voice packets, and image processing parameters are transmitted to the remote site.

c. Interactive video conferencing: In this scenario, a video conference is conducted between the local and Remote physicians. Two uses are possible: (1) display video of the two consulting physicians; and (2) display a live or recorded video of an operative procedure. This video is displayed in a sub-window during a consultation session. The digitized video is recorded at the Local WS site and can be saved as a part of the patient record for later review. This scenario will require at least a 45 Mbps (T3) network to the rural sites, in order to display near full motion video.

2.2 Distributed Computing Environment Issues

This section outlines distributed computing environment issues of Global PACS.

a. Communication: Global PACS spans many networks. Writing software directly to native network protocols is fine for a select few communications programmers. Usually developers end up with hacked-in, network-specific code, making it next to impossible to maintain, port or re-use.

b. Processing: The Global PACS environment consists of users at workstations requesting operations from server, such as a user at a workstation requests an image from an image server. Thus Global PACS environment provides an inherent opportunity for parallel processing. Using the client/server model and a threads capability, a server process can handle many clients at the same time.

c. Heterogeneous data: The Global PACS applications will be running on a variety of computers with different architectures and compilers, leading to differences in byte ordering, data formats, and padding between data items. For example, the applications should handle different format for floating point or integer number representation.

d. Internationalization: Even character sets have become extremely heterogeneous as Global PACS becomes truly global and uses different natural languages. Currently, most applications support alphabets that use 8-bit character sets. The internationalization aspect forces applications to handle multibyte character sets.

e. Timing: In a single, homogeneous system there is usually one clock to provide the time of day to all applications. In a Global PACS distributed environment, where each machine has it's own clock, and van be a in different time zone. It is unreasonable to assume multiple clocks will keep the same time. This is a problem for Global PACS applications that care about the ordering of events. It would be difficult to determine the ordering of events of each machine has its notion of the current time.

f. Synchronization: When multimedia applications with video and voice are included in Global PACS, the image annotation operation must be synchronized to the voice and video streams. Synchronization problems are caused by network delays, difference in device drivers, and availability of system resources such as CPU time.

g. Location Transparency: Global PACS contains multimedia data objects scattered over a wide geographical area. A directory service is is paramount in locating these objects in a location-independent way. Additionally, it allows users of global PACS to find alternate resources that are available to them. This allows for better load balancing and resource sharing and frees the user from keeping track of these objects.

h. Security: In a Global PACS environment it is paramount to control access to computer systems and preventing unauthenticated access to programs and data. This can be achieved through the use of user identification names and passwords.

i. Confidentiality: In a Global PACS environment patient confidentiality controls the amount of patient's record made available to medical personnel involved in the care of the patient. For example, an administrative user may be able to review only demographic and billing data and is not authorized to access clinical data.

j. Privacy: In a Global PACS patient data should remain private and protected from people not directly related to that patient's care [31]. For example, medical data in a public accessible media is encrypted to protect privacy concerns.

2.3 Operations

The physician users perform these operations in a Global PACS environment:

a. Exchange of image annotation: Participants in a remote consultation can use image annotation like pointing circling or boxing interesting area of an image as shown in Figure 2.1. Three modes are available for image annotation commands:

   (a) Fixed size framing: Allows the user to point to the point of interest. The user places the mouse cursor at the desired location on the display screen. and clicks

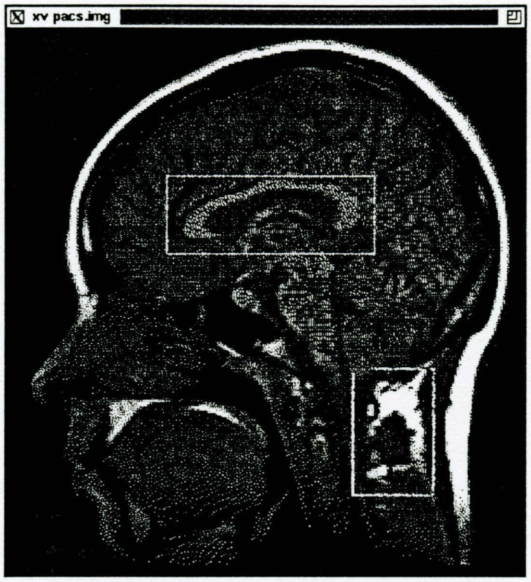


Figure 2.1: X-Ray Image with Variable Annotation

a mouse button. A small circle appears at the desired location and is displayed on the local and remote workstations.

(b) Variable size framing: Allows the user to box the area of interest using a variable size rectangle.

(c) Free hand drawing: Allows the user to draw any free hand drawing. This allows for maximum flexibility for users to express their annotations. the free hand drawing does not have to be a closed loop.

b. Voice and Video Interaction: The voice and video interaction among Global PACS users is essential. It helps users determine the degree of certainty about a topic. A medical expert interacting with a patient can become certain about a patient's response. Similarly, medical experts consulting using voice and video interaction can become certain about each one's views.

c. Access to Patient Information: Managing health care services is an information intensive endeavor. In a hospital, information systems usually handle tasks such as patient registration, admission, discharge, and transfer (ADT); clinical laboratories; radiology; and billing; in addition to specialized departmental applications. The Global PACS medical expert needs to access these data to reach a better diagnosis. Similarly, a Global PACS administrator needs to access these data for billing and managing the health care institution.

2.4 Characteristics of Medical Data

Medical data can be classified in terms of usage as clinical data or interactive data as suggested by [3].

1. Clinical Data: These data are generated by a medical device or an image acquisition device, and is included as part of the patient's record:

   (1) text (patient information, diagnosis report),

   (2) static medical images (x-ray, computed tomography, computed radiography images),

   (3) dynamic medical images (digital subtraction angiography, digital flouroscopy, ultrasound),

   (4) sounds (bodily sounds),

   (5) numeric (digitized signals, EKG, heart rate),

2. Interactive Data: These data are generated by medical experts as the engage in consultation sessions. Parts of this data may be included into a patient's record for further reference.

   (1) video sequences (live video, prerecorded video),

   (2) voice (digitized voice),

   (3) image annotation (pointing, circling, or boxing interesting area of image),

   (4) text (typed notes).

Global PACS has a great appetite for network bandwidth and storage capacity. Additionally, it is characterized by a requirement for high throughput and low latency. Therefore, the characteristics of the medical data used are explored in the following subsections.

2.4.1 Clinical Data

2.4.1.1 Text

Clinical textual data include patient information, diagnosis reports, and laboratory results. Data types include strings, text data, formatted text, date, time, time stamps, and telephone numbers. Usually, textual data units are relatively small (on the order of 1-5 Kbyte).

2.4.1.2 Static Images

Digitized images like x-ray images, are very large as some imaging equipment are capable of generating images up to 27 MByte [5] but usually an average size image is of the order of 50 Mbits, Traffic patterns of images within University of Arizona Radiology Department have been analyzed. In the graph in Figure 2.2, which was derived from raw data from [4], we see the expected image generation traffic over the network by year 1994 caused by static modalities generating traffic. The traffic was computed assuming a growth of 10% per year, as was estimated by the Radiology Department. Traffic patterns change over the days of the week and over hours of the day. As shown in Figure 2.2 a considerable volume of traffic takes place between 1 - 4 p.m. with a peak around 2:30 p.m.

At a viewing workstation, users generally need to see groups of images for each patient, as presented in Table 2.1. The image traffic is bursty and is characterized by very large data sets. At the University Medical Center four film sizes were observed with the relative frequencies given in Table 2.2. For larger hospitals the volume of images generated estimates have been made of up to 12.5 gigabytes per day for a fully digital radiology department in 1986 [6]. The previous Figure 2.2 shows image traffic from the imaging equipment to the database archive system. Meanwhile, a similar amount of traffic accompanied by old image data sets, will be transmitted from the database to viewing workstations for radiologists to use in diagnosis. Additionally, referring physicians will

Table 2.1: Number of X-Ray Images per Folder Request

Number of Images	Fraction of Total Requests
4	.40
6	.40
8	.15
12	.05

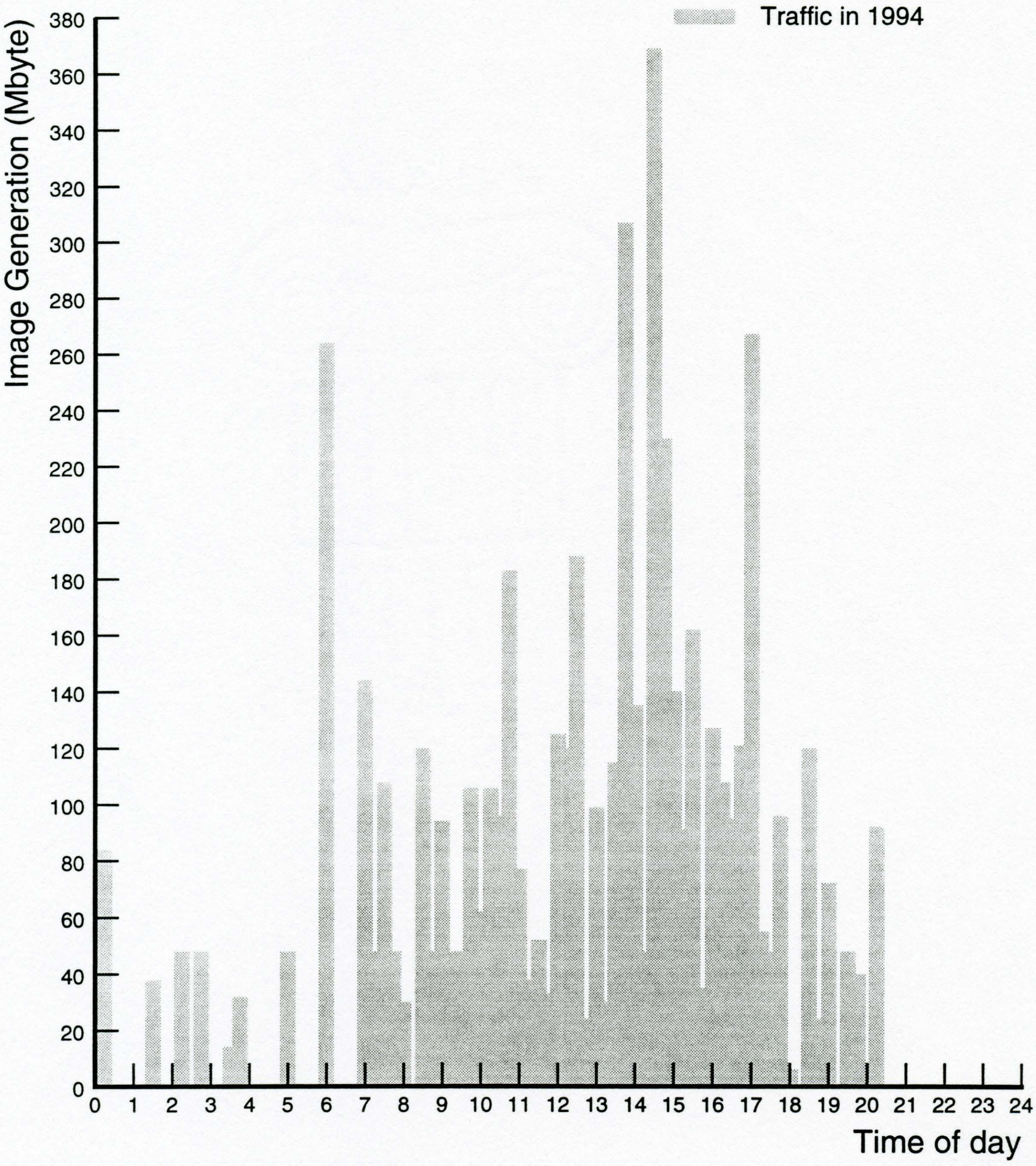


Figure 2.2: Image Generation Volume During Hours of the Day

Table 2.2: Size and Frequency of Request of Different X-Ray Film Types

Film Type	Film Size	Length (in bits)	Fraction of Films of this Type
1	14" x 17"	2048 x 2048 x 12	.60
2	11" x 14"	1609 x 1688 x 12	.15
3	10" x 12"	1462 x 1448 x 12	.22
4	8" x 10"	1170 x 1204 x 12	.03

Table 2.3: Network Bandwidth and Storage Requirements for Static Images

Modality	Image Size (bits)	Number of Images min – avg – max	Network BW (Mbps) min – avg – max	Storage (Mbytes) min – avg – max
Computed Tomography	512 X 512 X 16	20 – 50 – 120	40 – 100 – 240	10 – 25 – 60
Ultrasound	512 X 512 X 8	20 – 25 – 75	20 – 25 – 75	5 – 6 – 19
Magnetic Resonance Imaging	512 X 512 X 16	20 – 80 – 200	40 – 160 – 400	10 – 40 – 100

retrieve old and new images sets at their respective departments. This will add a similar volume of traffic. Thus, altogether, we find the traffic volume over the network is:

$$(imagesgenerated) + (imagesgenerated + olddatasets) \times 2$$

Table 2.3 reflects the storage and network bandwidth requirements for static image modalities. The network bandwidth is calculated based on the observation that radiologists prefer to see studies appear at their viewing workstations within 2 seconds [10]. The Ultrasound modality in static mode save a subset of images for later diagnosis.

Table 2.4: Network Bandwidth and Storage Requirements for Dynamic Images

Modality	Image Size (bits)	Session Length (seconds)	Network BW (Mbps)	Storage (Mbytes)
Real Time Ultrasound	512 X 512 X 8	60	60	450
Digital Fluoroscopy	512 X 512 X 8	60	60	450
Digital Subtraction Angiography	1024X1024X 16	60	480	3600

2.4.1.3 Dynamic Images

Dynamic images are generated by dynamic modalities like digital subtraction angiography (DSA). Dynamic modalities generate images at a constant rate, which are viewed in real time by the technician and inspected by a radiologist. In DSA two images are made: one before contrast medium has been injected in the patient and one afterward. Assuming no motion the only difference between these two images is the contrast added to the second image. By subtracting them one can eliminate all other structures from the images, greatly enhancing the contrast filled blood vessels. By applying this method to a sequence of frames, one can observe the diffusion of contrast material in patient's body. In a totally digital radiology department assuming 1k $\times$ 1k $\times$ 16 bits per frame transmitted at a rate of 30 frames per second, the network must sustain 480 Mb/s per one dynamic session. At the University Medical Center at the University of Arizona three dynamic sessions can be running in parallel.

2.4.1.4 Compression

Many data compression techniques have been designed over the last twenty years. The best known is the block-based transform coding technique. The Joint Photographic Expert Group (JPEG) standardized a block-based transfer coding for compressing still images [11, 12]. Similarly, the Moving Picture Experts Group (MPEG) standardized a motion-compensated interframe block-based transform coding for compressing video sequences [13]. Application of compression techniques for medical data is limited by legal constraints inhibiting the use of lossy compression. Lossy compression introduces artifacts into the image making it unacceptable for diagnosis [14]. Lossless compression only gives about 2.5-3:1 compression ratio and introduces a great deal of processing overhead for compression and reconstruction of image and corresponding delay. Furthermore, compression ratio is dependent on the site of body and modality [34]. Some body parts like the chest and digestive organs in the abdomen, are not adaptive to high compression ratio. Therefore, until we obtain low overhead compression technology without the introduction of artifacts, it is preferable if images are transmitted in uncompressed form.

2.4.2 Interactive Data

There are many and varied reasons for integrating multimedia communication in health care systems design. In particular, among others, video and voice communication will allow a point-to-point conversation between a radiologist, physician and technician [15]. They can use image annotation like pointing circling or boxing interesting area of an image as shown in Figure 2.1. Conference calls, such as these, to diagnose an image will save time and energy, since physicians and radiologists can participate in the discussion from remote sites. In the context of health care delivery, this facility will enable conferences on a national/international level. Moreover, the advent of mobile medical units with multimedia capabilities will bring expert radiologists to rural areas without long and possibly economically infeasible travel time. Finally, voice communication can satisfy medical and legal demands which require multiple presentation of the same data [16].

Voice entry medical systems have an advantage. For example, one of the shortcomings in a radiology department is that the style of a diagnostic record varies from radiologist to radiologist. Voice entry radiologic database systems are becoming readily available. Such a system uses voice-to-text macros to create highly structured and accurate radiology reports [17]. This can be incorporated within a radiology department database archive system. One advantage of doing this is that it will lead to a reduction of transcription costs. Furthermore, such a scheme could be used to generate reports from 4 to 48 hours faster than current transcriptionist-mode dictating stations allow. Having reports instantaneously available to referring physicians following dictation has enormous value. This report generation system will significantly reduce the number of phone calls requesting patient results that the radiology department receives each day [18].

The network will handle video traffic between consulting sessions. Video traffic will require a rate of 20–30 frames per second per video user, with a frame size of 310×160 which is comparable to the Mpeg standard. Voice traffic between consulting sessions requires 64 Kb/s per voice user, we do not find the voice traffic to constitute a substantial traffic volume

[19]. For image annotation, only the commands are transmitted since they are of a small size (< 100 Bytes).

2.5 Global PACS Remote Consultation and Diagnosis System

The Global PACS Remote Consultation and Diagnosis (RCD) system was developed under a National Science Foundation grant by Martinez [2]. It allows medical experts at remote locations to access medical data, view medical images, and perform a live consultation session as depicted in Figure 2.3. The system has two types of nodes: 1) RCD Participant, and 2) RCD Service Provider. A participant can be a local workstation or a remote workstation where medical experts engage in a session. A service provider offers easy access to needed medical data during a session (GPACS Server). An example can be a database containing patient records such as a hospital information system or radiology information system, or a file system for storing medical images. The steps leading to establishing a remote consultation are shown in Figure 2.4. The participants recognize each other and obtain the study multimedia data from the service provider. The interactive data previously described are exchanged in a live multimedia remote consultation and diagnosis session. In addition a multimedia session information (synchronized voice, image, and image annotation) are recorded. The multimedia session information represents related data streams in spatial and real-time events. In order to store these related streams for playback we need to add synchronization information. A variety of user scenarios in an RCD session are presented in [2], [3].

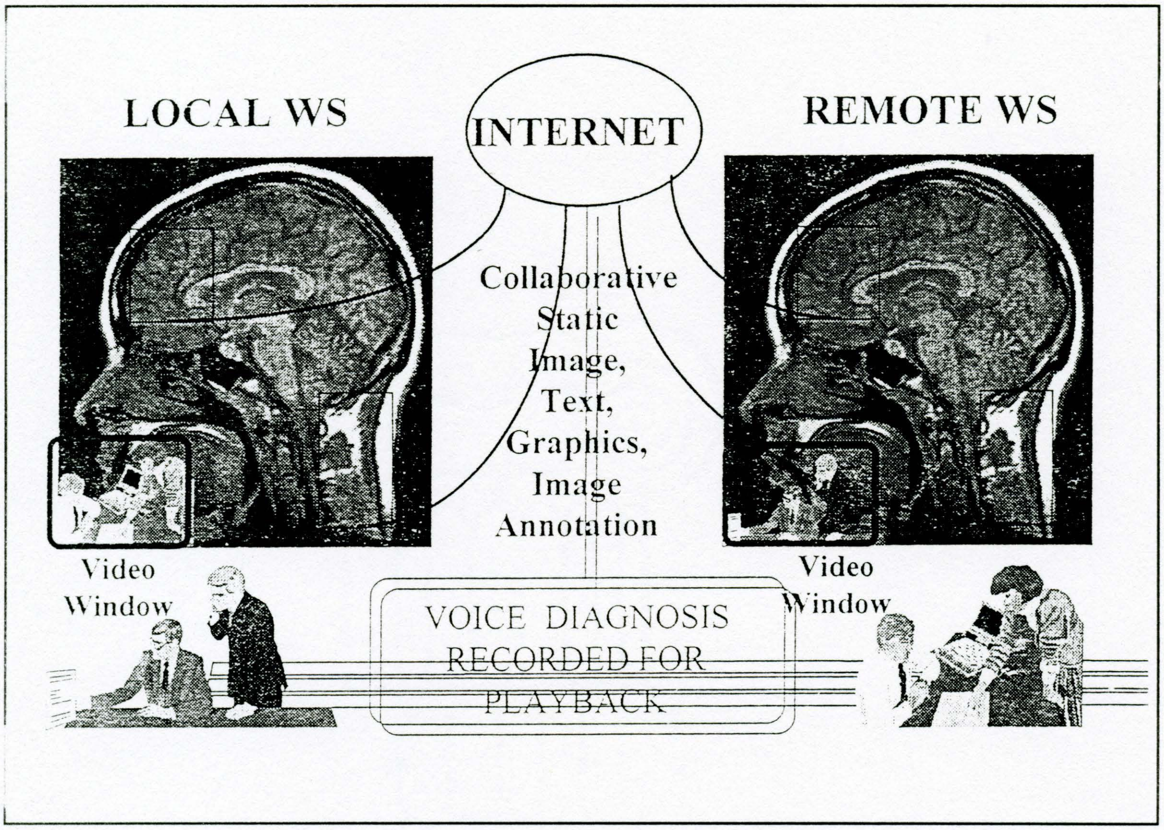


Figure 2.3: Global PACS Multimedia RCD Session

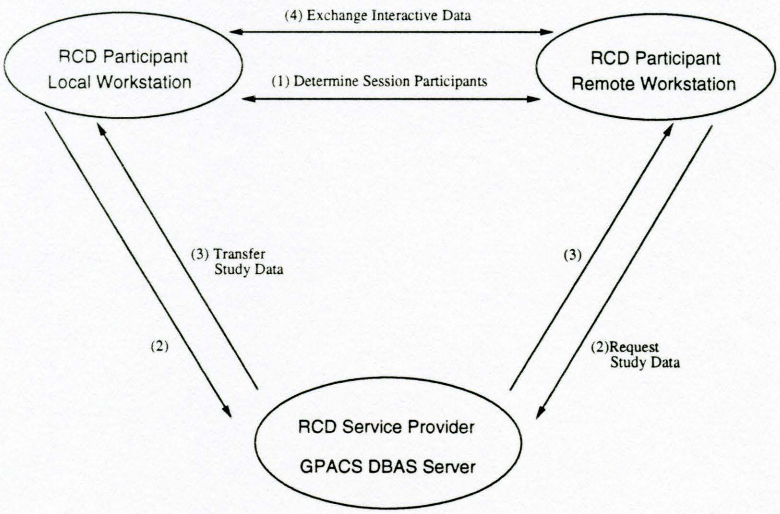


Figure 2.4: Steps in Establishing Remote Consultation Session

CHAPTER 3

Distributed Computing Environment: An Overview

This chapter provides an overview of the Distributed Computing Environment (DCE). Certain definitions and concepts that are used in our prototype are explained. The reader is referred to [65]-[71] for further information.

3.1 DCE Architecture

Distributed computing is a method of computing that involves the cooperation of two or more machines and processes communicating over a network. The Open Software Foundation (OSF) DCE provides services and tools that support the creation, use, and maintenance of distributed applications on heterogeneous computers, operating systems and networks, as shown in Figure 3.1. Some of the features that DCE offers are:

1. provides tools and services that support distributed applications.

2. offers an integrated and comprehensive set of services.

3. provides interoperability and portability across heterogeneous platforms.

4. supports data sharing.

5. supports participation in a global computing environment.

DCE follows the client/server model as shown in Figure 3.2. The server is typically implemented as a continuous process (daemon). The client is usually implemented as a

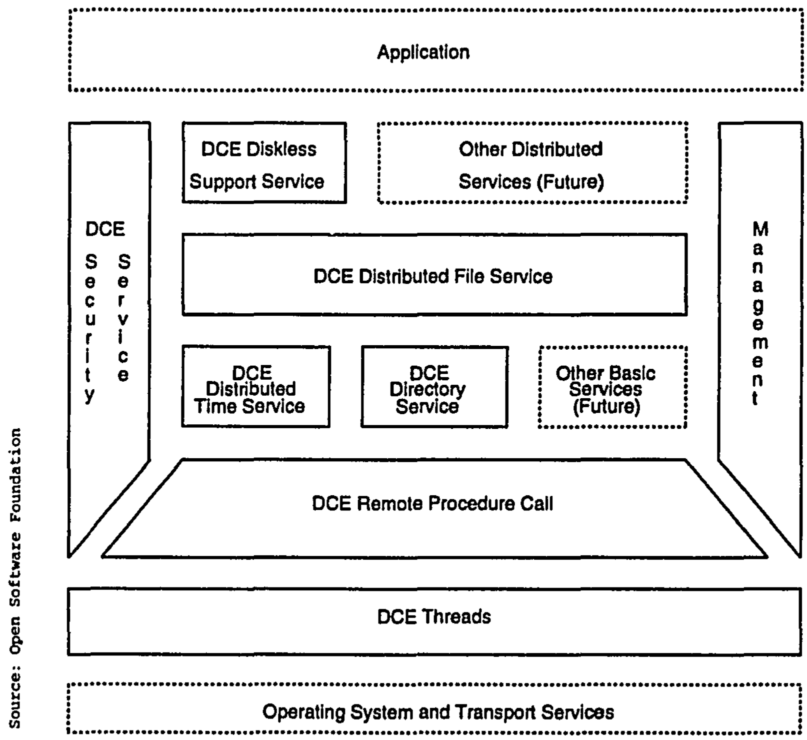


Figure 3.1: Distributed Computing Environment Architecture

library. The DCE environment is categorized into distributed programming facilities and distributed services. Figure 3.3 lists these facilities and services.

3.2 DCE Technology Components

The OSF DCE comprises several technology components as shown in Figure 3.3.

1. **DCE Threads:** DCE Threads is provided as a user space library. It consists of an interface that allow programmers to create and manipulate threads. Other technology components of OSF's DCE assume the availability of threads support [66]. However, this part is transparent to the DCE user.

2. **DCE Remote Procedure Call:** The DCE Remote Procedure Call (RPC) facility consists of a development tool and a runtime service. The development tool consists of a language (and its compiler) that supports the development of distributed applications following the client/server model. The runtime service implements the communication protocols between the client and server.

3. **DCE Directory Service:** The DCE Directory Service is a central repository for information about sources in the distributed system. The DCE Directory Service

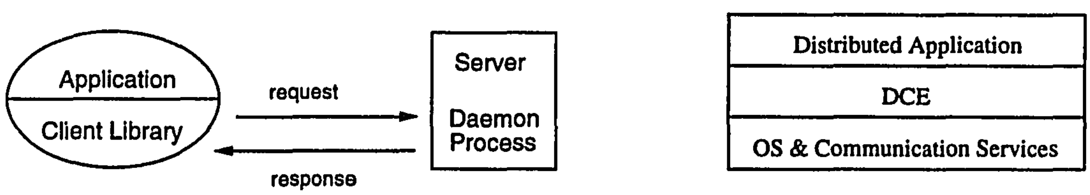


Figure 3.2: Distributed Computing Environment Model

Distributed Programming Facilities	Distributed Services
o DCE Threads o DCE Remote Procedure Call implement Application Programming Interfaces (APIs)	o DCE Directory Service o Distributed Time Service o Security Service o Distributed File Service o Diskless Support Service

Figure 3.3: DCE Facilities and Services

comprises several parts: the Cell Directory Service (CDS), the Global Directory Service (GDS), and the Global Directory Agent (GDA). The CDS manages a database of information about the resources in a group of machines called a cell. The GDS implements an international standard directory service, and provides a global name space that connects the local DCE cells into one worldwide hierarchy. The GDA acts as a go-between for cell and global directory services. A *cell* is a group of DCE machines that work together, share the same name space, and are administered as a unit. The DCE environment is a group of one or more DCE cells that can communicate with each other. Each cell must have at least one each of the following servers in order to function: 1) Cell Directory Server (CDS), 2) Security Server, 3) Distributed Time Server (DTS). Other DCE servers may be present in a given DCE cell to provide additional functionality.

4. **DCE Distributed Time Service** The DCE Distributed Time Service (DTS) provides a mechanism to synchronize the clocks on different participants in a DCE cell. Furthermore, it provides a way of keeping this synchronized time close to the *coordinated universal time*. Different participants in a DCE environment will have the same

idea what time it is, ensuring correct ordering of events in this distributed system. Finally, DTS supports reception of correct time from an *external time provider* device such as radio or telephone.

5. **DCE Security:** DCE Security uses Kerberos, an industry standard. Security will be maintained using the *authentication* of user identities, checking for *authorization* to access system resources using *access control list* facilities, and protecting data whether in file format or in transit (RPC) via *encryption.*

6. **DCE Distributed File Service:** DCE Distributed File Service (DFS) is a client/server application that utilizes other DCE services. It uses the DCE CDS service to organize directories, DCE RPC to move data among clients and servers, and DCE Security service to enforce privacy and authorization rules. The DCE DFS integrates with these services to provide a uniform file access using a global name space. The location of files maintained by DFS is transparent within one DCE cell, allowing better load balancing and higher availability through replication.

7. **DCE Diskless Support Service:** DCE Diskless Support Service allow nodes without disks to fully participate in a DCE environment. Thus, functionalities that the local disk supported are now provided across the network. The main functionalities supported by DCE are: a) sending a copy of kernel for booting, b) sending configuration data to setup the node, c) full support of DFS across the network, and e) providing swapping facilities for data on processes blocked on the diskless node.

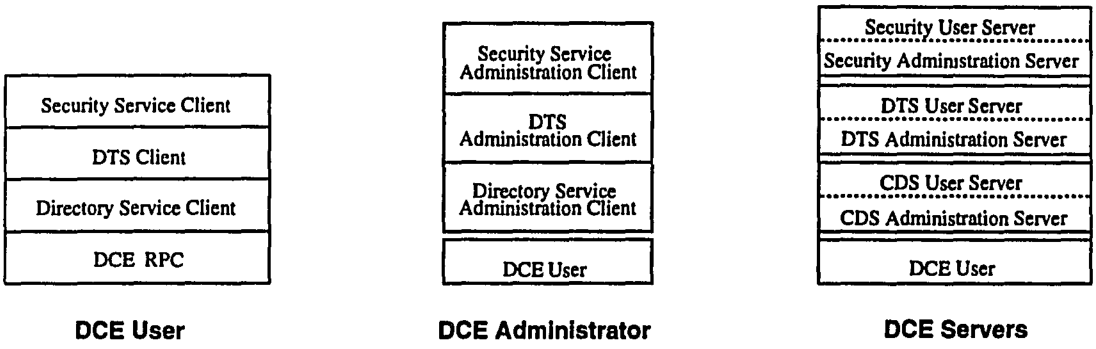


Figure 3.4: Prototype DCE Setup

There are three basic types of participants in a DCE environment. An experimental setup of these DCE participants is shown in Figure 3.4.

1. DCE User: it will participate as a client in the DCE environment.

2. DCE Administrator: it will enable an administrator to control servers running in the environment.

3. DCE Server: it runs one or more of the DCE services.

CHAPTER 4

System Design

The system design aims to provide a platform with a standardized interface for Global PACS applications. In this chapter we present the modular design using several components. Each component was designed and tested individually, then all of the components were integrated to provide the platform.

This chapter presents design and implementation of a Scalable Network Video delivery mechanism. This will provide an interface to dynamic sequence generating modalities. These design components are integrated in the design of Comprehensive Chart. The Comprehensive Chart offers an open architecture to access and visualize a complete repository of patient data. This section will present the conceptual approach, the available views data, the viewing tools, and the system architecture. Then the chapter defines the mapping of the Distributed Computing Environment services to Global PACS environment.

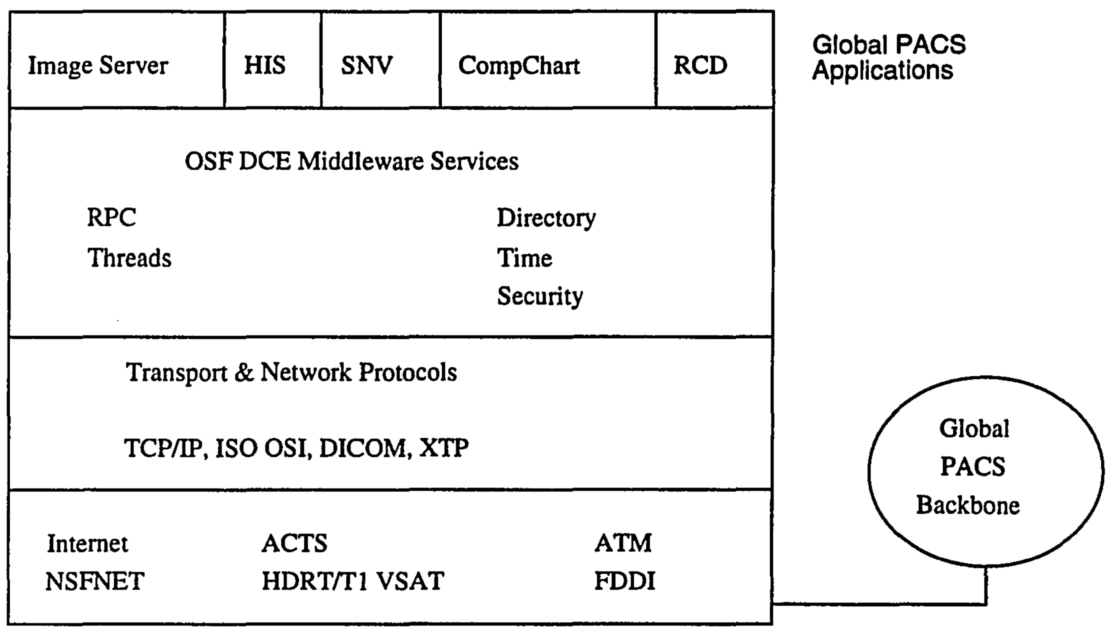


Figure 4.1: DCE is the Middleware for System Design

4.1 Scalable Network Video Delivery Mechanism

The world wide web (WWW) is a "wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents". The WWW provides a new protocol called the hypertext transfer protocol (HTTP), which is an interface to various Internet protocols such as the file transfer protocol (FTP). Hypertext provides links to pieces of text through hot spots in a document. A hot spot is a special portion of a document that can be linked to other documents and selected with a mouse pointer. Hypermedia documents contain not only links to textual data, but also to other forms of media like images, audio, and video. Documents or resources are located by means of uniform resource locators (URL)s which specify the protocol to retrieve the document, the server providing it, and a path name to that particular document. To provide a uniform interface to the WWW the national center for supercomputing applications (NCSA) developed a versatile multi-platform interface called Mosaic. The Mosaic interface became the most popular browser on the web.

4.1.1 The HTTP Approach

The HTTP setup for delivery of video sequences compressed according to the motion pictures expert group (MPEG-1) standard is illustrated in Figure 4.2. A sample URL to retrieve a video file is:

FTP://zax.radiology.arizona.edu/medical-videos/Ultrasound1.Mpeg1

where FTP is the protocol for retrieving the document, zax.radiology.arizona.edu is the server name, and medical-videos/Ultrasound1.Mpeg1 is the pathname to the video file. The client uses FTP to bring the video file and saves it in a /tmp directory. Consequently, an MPEG Player is spawned to read the file, decode the MPEG sequences and display them. This simple mechanism has two limitations. The first limitation is that the client user has to wait for the whole video file to be transferred to the local hard disk before starting to view the video (response time). For example, assume that we have a video file containing Ultrasound sequences at a rate of 30 frames per second, with 320×240 bits per frame. For a typical session of 5 minutes, this yields a video file size of 86.4 MByte. The response time over an Ethernet network will be a minimum of 69 seconds. This response time grows proportionally with file length, making browsing through video libraries cumbersome. The second limitation is that the client should reserve enough disk space for the /tmp directory to accommodate all of the video file. Video files tend to be large and the user will not be able to view them without sufficient dedicated disk space.

The design suggested here will be used to disseminate Ultrasound MPEG video sequences at the University of Arizona. This mechanism is called scalable network video (SNV) since it overcomes the two limitations previously noted, since it requires a fixed response time and no reservation of disk space at the client machine. The following sections explain the system concept, its components, and the SNV mechanism.

4.1.2 System Design

The SNV delivery system architecture is illustrated in Figure 4.3. The system components are: the user machine that will run the Mosaic client, the Mosaic server, the video server, and the communication network. One design goal was to maintain the Mosaic

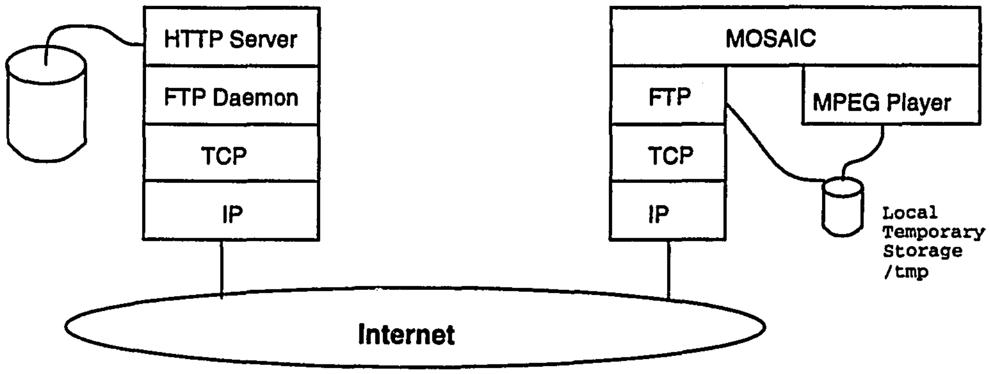


Figure 4.2: Mosaic Mechanism for Delivering Video

browser as a user interface and provide facilities around it to support better video delivery. The video server can support these video services:

- **Realtime Video:** The server will deliver real time video to the client. This is similar to a TV station delivering live coverage of an event.

- **Playback Video:** The server fetches the requested video from the video repository and delivers it to the client.

The video server will perform these functions:

1. **Request Handling:** The server accepts requests from Mosaic clients and prepares the service (real-time or playback video) for delivery.

2. **Delivery:** The server sends segments of MPEG sequences to the client, and the client is responsible for playing it. In the current implementation the server establishes a TCP connection to the client providing a stream of MPEG segments.

The steps leading to delivery of video are shown in Figure 4.4. First, the user clicks a menu button requesting a video service, using HTTP mechanism. Second, using the HTTP

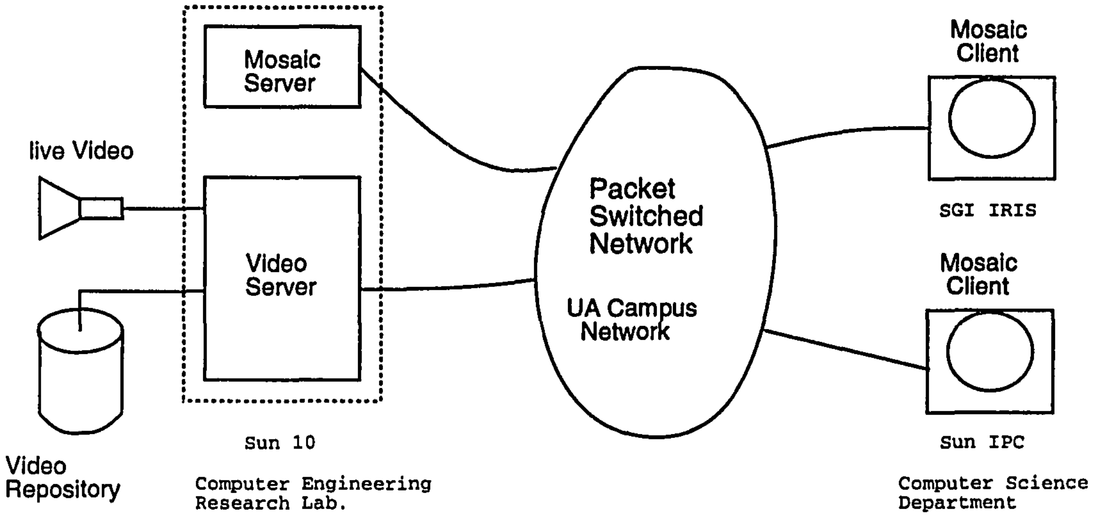


Figure 4.3: Video Delivery System

the Mosaic server returns the address of the video server that may provide the requested service. In the third step, the user forwards the request over a TCP connection to the video server. Finally, if it can satisfy the request the video server establishes a connection to the client machine and starts sending video segments. *Nmpeg-play* is an MPEG player that accepts input from the network connection and plays the video.

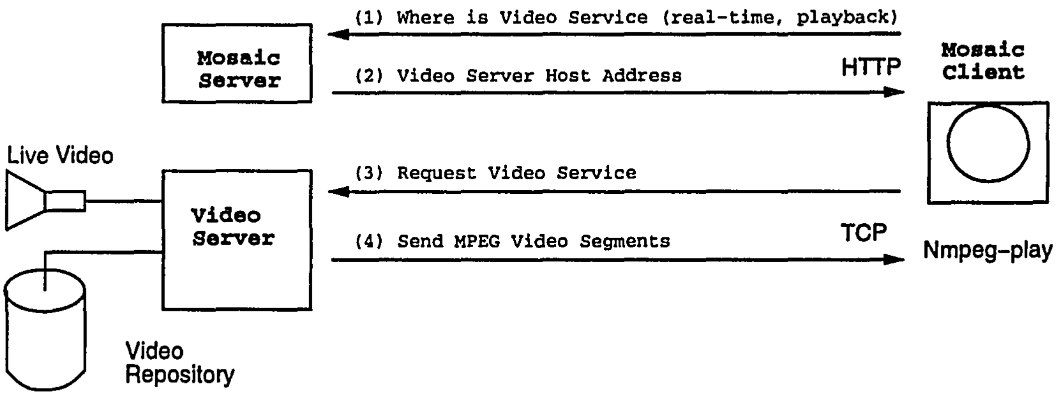


Figure 4.4: Information Flow Between Mosaic Client, Mosaic Server, and Video Server

4.2 Comprehensive Chart

The Comprehensive Chart (CompChart) provides an interface to access and visualize a complete and accurate repository of all data relevant to the health of a patient. The user interface is intuitive. It provides a complete picture of the patient's current medical condition. The system provides access to previously stored data as well as live data as it is being captured. The CompChart multimedia data are accessed from a variety of information systems such as the Hospital Information System (HIS), Radiology Information System (RIS), Picture Archiving and Communication System (PACS), or directly from medical instruments (e.g. vital signs), or image acquisition devices (e.g., ultrasound). The patient record is presented to the user using the World Wide Web (WWW) technology, namely the HyperText Markup Language (HTML). The multimedia patient data is displayed using WWW players.

This chapter will present the conceptual approach of the CompChart system. The CompChart system uses the concept of views to trim down the amount of data presented to the end user. It presents some viewing tools such as the View Generator, and explains some of the elements in a view. It describes a sample of players used to display medical data, and presents potential user scenarios. Finally, it describes the CompChart system architecture.

4.2.1 Conceptual Approach

a. Open Architecture: the CompChart uses the HTML script to describe a patient record. This allows vendors to design their own medical browsers. Additionally, they can introduce different players for medical data.

b. Modular: The CompChart system can use different communication protocols such as HTTP, DICOM, and HL7, to name a few. It can use additional information sources as they are added to the system and employ different data players.

c. User Driven: The CompChart design allows users to shape their personal views of the system to match their needs.

d. Ease of Modification: the CompChart is easy to modify either by system designers, administrators, or by users. This is an important feature since the system will undergo an iterative process of changes.

e. Evolutionary: The CompChart system does not require hospital departments to immediately change their practices for the system to be successful. As departments start keeping digital records of patient data, that data can be added to the system. It also preserves the ownership of data to its respective departments. It allows each department to set its policy of who can get the data and what can be viewed.

f. Heterogeneity: The CompChart design allows the access and visualization of data from different vendor systems. Even medical systems with proprietary interfaces can be integrated in the CompChart system.

4.2.2 Comprehensive Chart Views of Patient Record

The CompChart system provides a complete picture of the patient record. Nevertheless, this amount of data can be overwhelming and distracting to the medical expert. CompChart uses the concept of views to trim down the amount of data presented to the end user. The CompChart provides these views of a patient record:

a. Chronological View: The Chronological View of patient data allows temporal relationships between events to be graphically visualized. Currently, this view provides a linear relationship among events and data elements. Future developments may display complex relationships between events (e.g., is related to, cause effect, contributes to, etc.). The display of these complex relationships will allow the medical expert to visualize higher order patterns in order to reach a better diagnosis.

b. Modality View: The Modality View of patient data allows personnel from modalities to view patient data related to a particular modality.

c. Problem View: The Problem View allows the medical expert to concentrate on patient data related to a particular illness or problem.

d. Anatomy View: The Anatomy View of patient data presents the body part on which exams were performed.

e. Organ System View: The Organ System View of patient data presents the organ system on which exams were performed.

f. Lifetime View: The Lifetime View summarizes the medical history of a patient. It compresses the Chronological View and eliminates lots of small detail. This will enable the medical expert to visualize long-term problems.

g. Combined View: The Combined View provides the capability of viewing combinations of different views. For example, it can provide a combined view of an Organ System View and an Anatomy View of patient data, chronologically ordered.

4.2.3 Comprehensive Chart Viewing Tools

Upon successful deployment of the CompChart system, the amount of medical data related to a patient can be overwhelming. We anticipate that most users will prefer a combined view of the patient record.

4.2.3.1 View Generator

To generate the interface for the combined view we use an object-oriented generator (Figure 4.5). This generator will aid the medical experts in tailoring their view of the patient data. The medical expert will have a personalized combined view generator, as depicted in the CompChart View Generator (Figure 4.6).

4.2.3.2 Elements of a Comprehensive Chart View

Here is the description of the elements of a CompChart view:

a. Red Icon of a Modality: This data acquisition is going on right now. The input can come from an ultrasound scanner, an EKG, a vital signs bedside monitoring device,

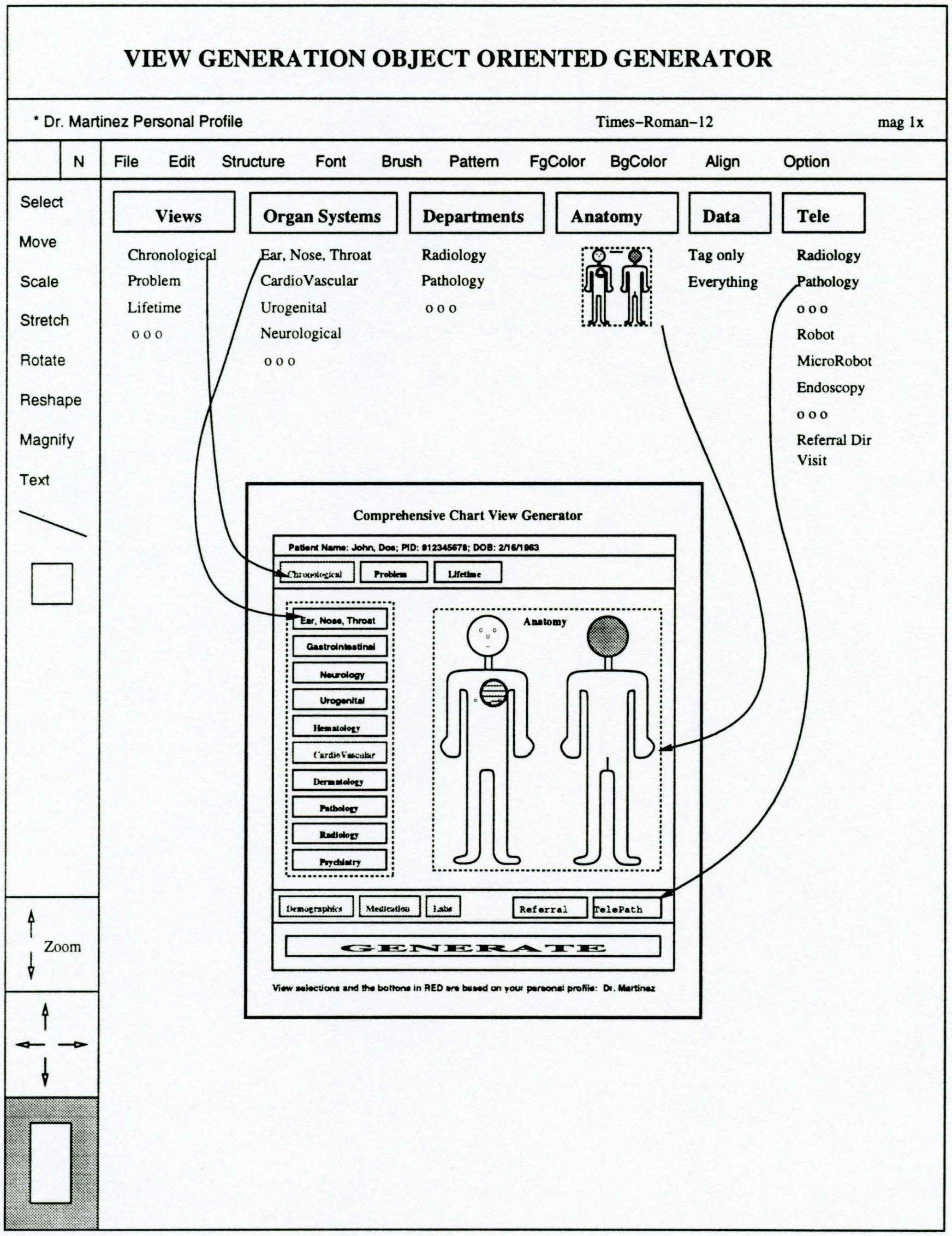


Figure 4.5: Object-Oriented Generator of the Combined View Interface

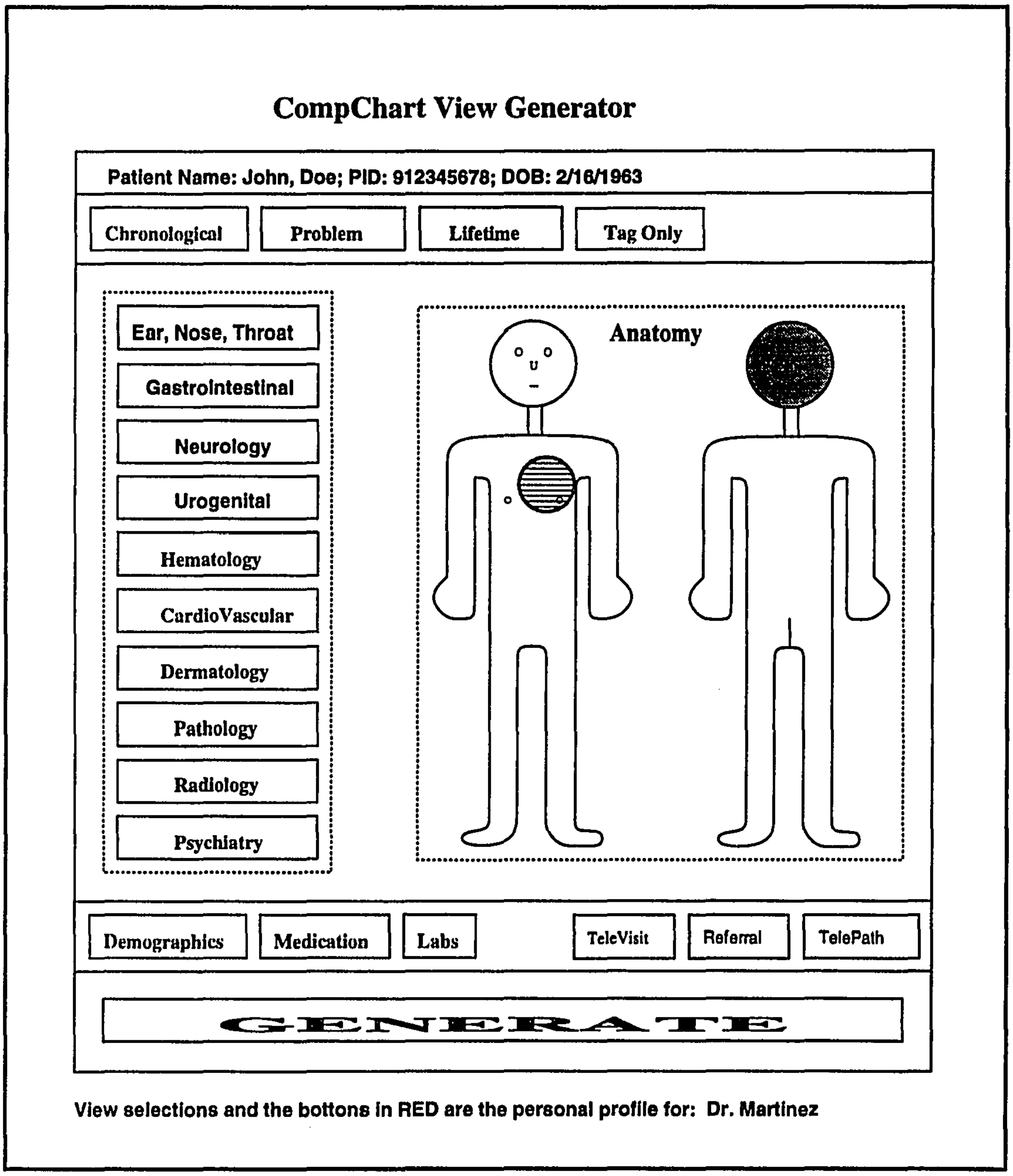


Figure 4.6: Combined View Interface

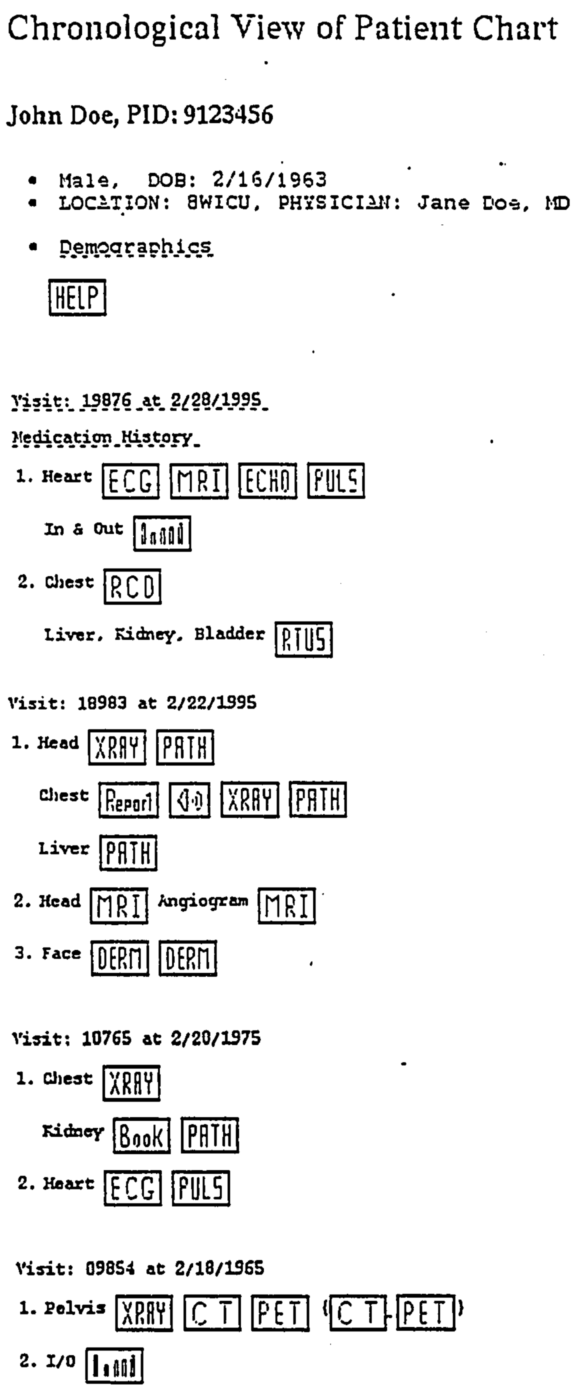


Figure 4.7: Chronological View Interface

RTUS	**Red Icon of a Modality** Click: Display the clinical data from that real time modality. Description: Data acquisition is going on right now in real time.
PATH	**Yellow Icon of a Modality** Click: Display the clinical data from that modality Description: The data has been captured but the official report is not available yet.
XRAY	**White Icon of a Modality** Click: Display the clinical data from that modality Description: The data has been captured, read by an expert and the official report is available
	**Voice Report Icon** Click: Play the recording of a voice report. Description: The following clinical data (to the right) was read by an expert and this is the voice dictation.
Report	**Text Report Icon** Click: Display the textual report. Description: The following clinical data (to the right) was read by an expert and the dictation was transcribed.
	**Histogram Report Icon** Click: Display a histogram. Description: This is a histogram representation of numerical data.
Book	**Academic Reference Icon** Click: Display the academic reference (i.e., book). Description: The following image (to the right) was interpreted after referring to this academic reference (e.g., text book case study).

Figure 4.8: Comprehensive Chart Operational Modes

or a pathology microscope. This allows the medical expert to observe remotely the output of these dynamic modalities.

b. Yellow Icon of a Modality: The yellow icon warns the user that clinical data has been captured but the official report is not available yet. This CompChart attempts to provide access to the data as they are generated, leaving the final decision of how to use it to the medical expert.

c. White Icon of a Modality: The data and the official interpretation/diagnosis are available.

d. Voice Report Icon: In some departments such as radiology, the radiologist makes a dictation of the diagnosis on tape recorder which is then transcribed into a printed official report by a transcriptionist. It usually takes 1 to 2 days for the final written report to be available. This Voice Report is available for users who do not wish to wait to get the report in a written format.

e. Text Report Icon: The CompChart provides access to the textual report once it is available.

f. Histogram Report Icon: The CompChart provides methods for the user to better visualize the data. This Histogram Report will present the numerical data as a histogram; for example, the in-take-out-take data of a patient.

g. Academic Reference Icon: In order to preserve the thought process involved in a diagnosis, the Academic Reference Icon links to text book case studies that the medical expert based the diagnosis on. This link can be to an expert system or medical handbook such as MedLine. Additionally, this is helpful for educational purposes as the teacher would provide access for his students to exemplary patient records with academic references included.

4.2.3.3 Players

The CompChart design allows for different players to visualize the data. These players can be general purpose multimedia players or specialized players designed with built-in processing functions for medical data.

a. Signal Player: This player will display stored signals, such as the EKG chart player depicted in Figure 4.9.

b. Real-Time Signal Player: This player will display real-time signals as they arrive from the source, such as the real-time vital signs player depicted in Figure 4.10.

c. Table Player: This player will display tabulated data, such as the medication report player depicted in Figure 4.11.

d. Graphical Player: This player will display data in graphs for easy visualization. Figure 4.12 displays in-take and out-take data as a histogram. In this example, the player accepts real-time data and displays it as it is being generated from the bedside device.

e. Dynamic Sequences Player: This player displays dynamic sequences or video output generated from dynamic modalities, such as the MRI player depicted in Figure 4.13.

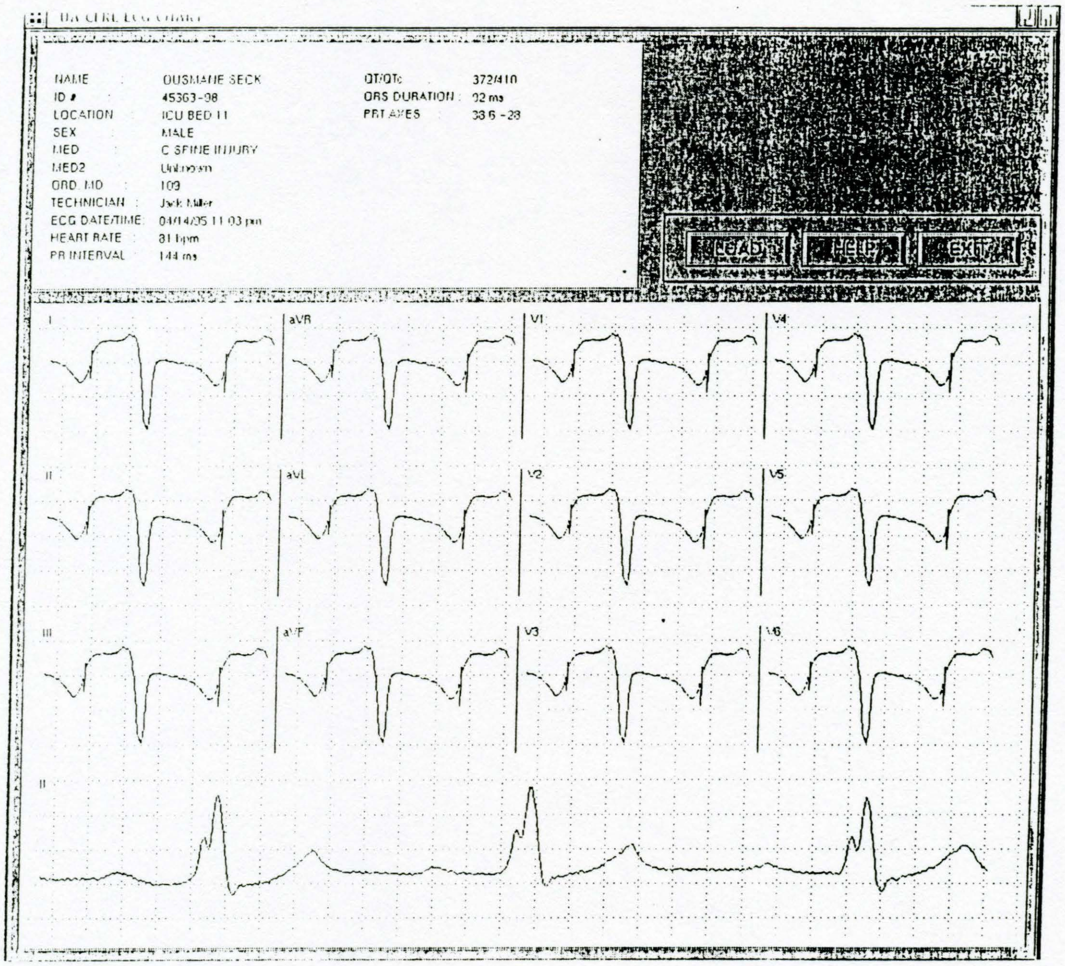


Figure 4.9: EKG Chart Player

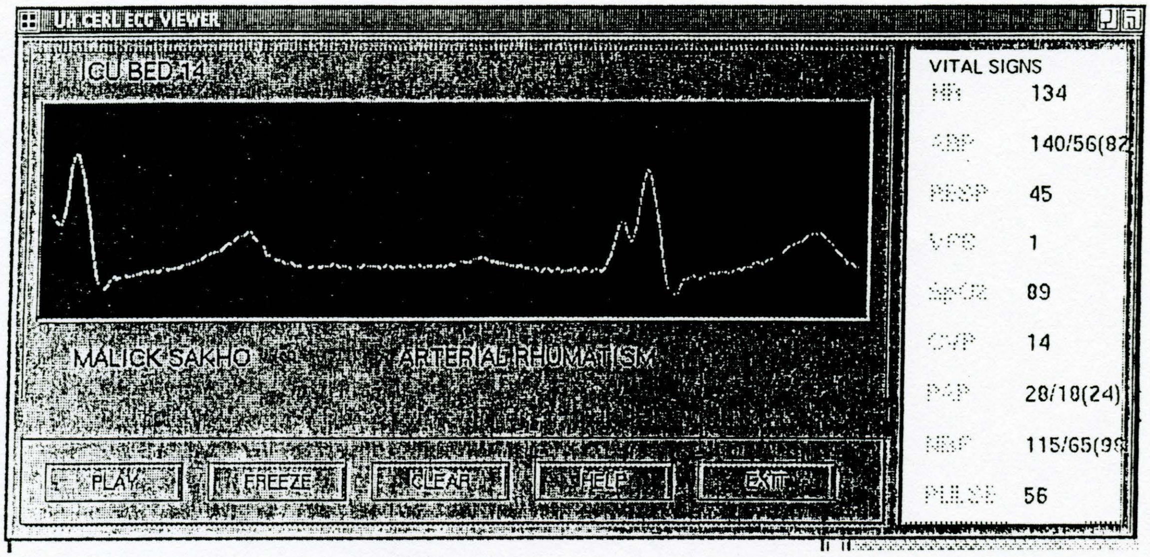


Figure 4.10: Realtime Vital Signs Player

John Doe, 912345 **Medication Report** 2/28/1995

This list is sorted showing first medication with STATUS = 'A'ctive and then sorted by STARTING DATE

Medication	Status	Starting Date	Duration	Dose	Frequency	Route	Physician
Aspirin	A	2/12/95	3 wks	100mg	2/day	mouth	Collohom, Mat
Algezal	A	1/10/95	1 month	25mg	4/day	skin	Collohom, Mat
Novacain	I	1/20/95	1 Wk	10mg	when needed	mouth	Collohom, Mat

Figure 4.11: Medication Report Player

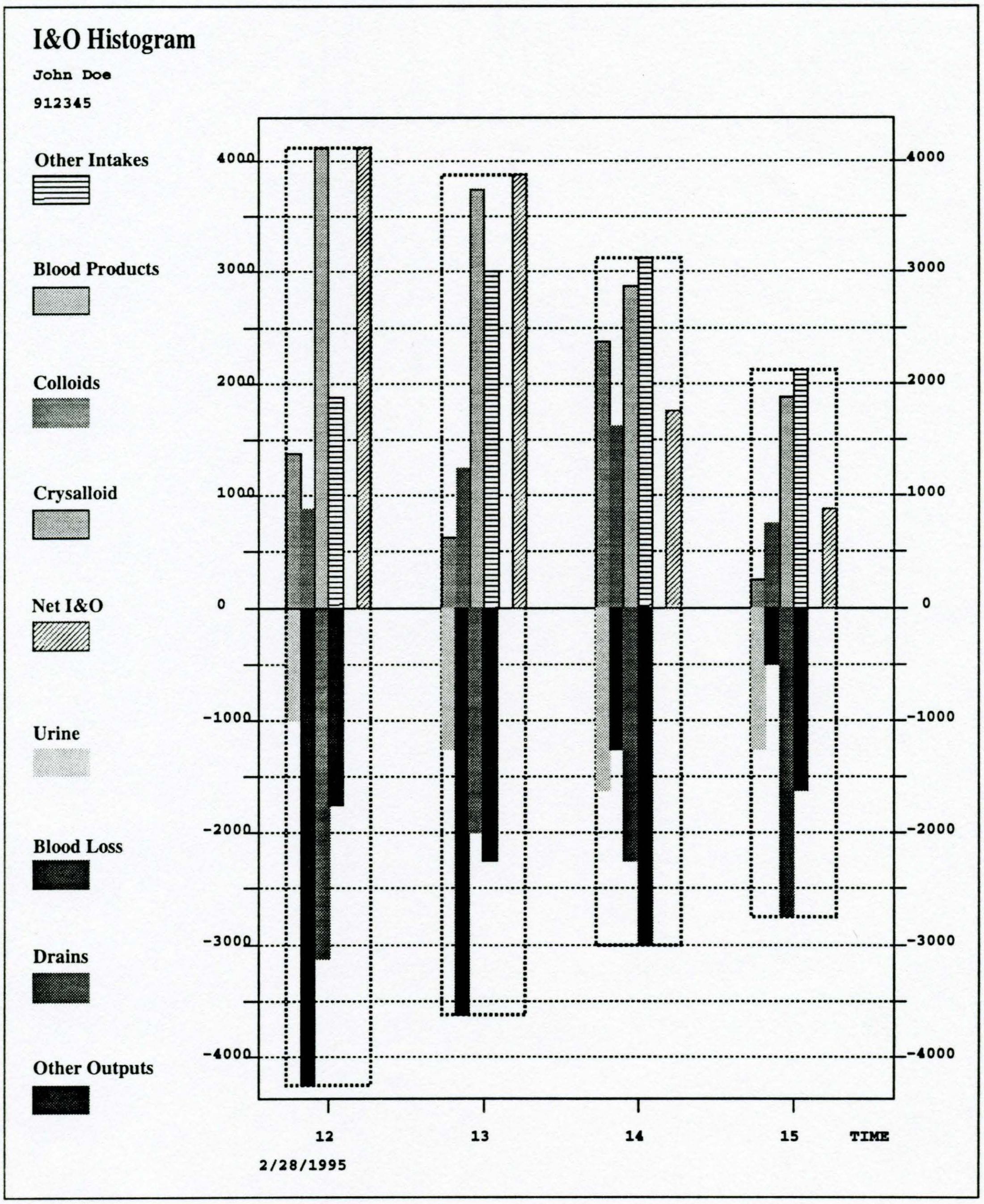


Figure 4.12: Realtime In-take & Out-take Player

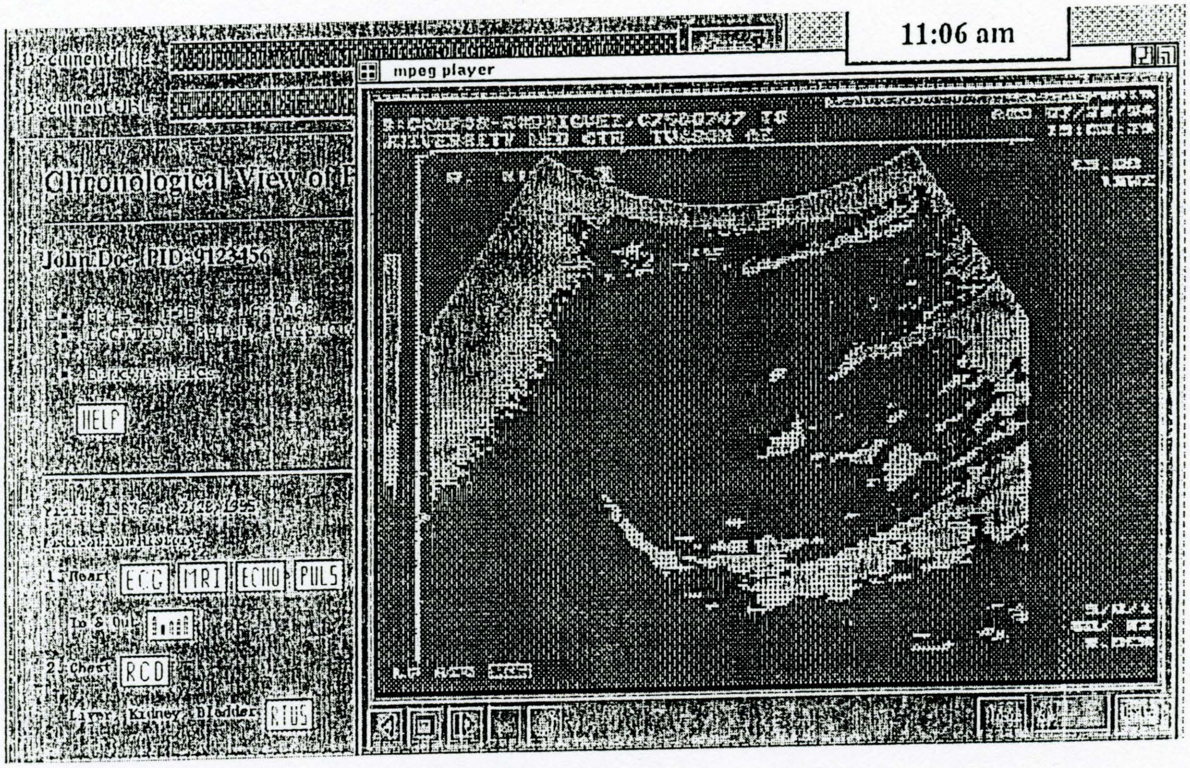


Figure 4.13: Medical Dynamic Sequences Player: MRI

4.2.3.4 User Scenarios

This section demonstrates several scenarios for the use of CompChart. The expected user is a medical expert in the clinic, or performing clinical rounds in hospital wards, or in a rural area hospital.

a. Regular visit: In the regular case the medical expert gets a complete picture of the patient history to reach a better diagnosis. A medical expert may access the patient record from the Hospital Information System. After examining, the expert may request an x-ray image with the radiology report. Additionally, the expert may request the results of some lab orders.

b. Tele-monitoring: in the tele-monitoring case the medical expert can view real-time outputs from bedside devices or ongoing procedures. The expert may check on the current status of the patient. There is no need to be physically next to the bedside. A real-time display of vital signs is made available over the hospital network.

c. Tele-visit: In the tele-visit case the medical expert can perform visual examination of the patient with the benefit of having access to the medical record. This eliminates unnecessary travel, especially for follow-up visits when the expert may share some items of the record with the patient.

d. Tele-diagnosis: In the tele-diagnosis case the medical expert has access to a data generating device, such as a microscope with a pathology specimen in it. For example, at a remote location (may be in a rural area) medical images are acquired by a technologist. The images will be transported either over a phone line or over the

Internet to a medical center where they are stored. An expert will examine these images, provide diagnosis and report the result back to the remote location.

e. Tele-consultation: In the tele-consultation case the medical expert establishes a remote consultation with a referring physician. This is based on the established referral directory.

f. Tele-operations: We anticipate the need to control a remote robot for performing a tele-surgery; or to receive data from a micro-robot; or to perform a remote endoscopy procedure. The CompChart should provide access to simple control interfaces.

4.2.4 Comprehensive Chart System Architecture

The CompChart system follows the client/server architecture. The roles of the client and server are:

a. CompChart Server: It keeps information about systems configuration and patient record. The server will keep the necessary information about an information system in order to access it as depicted in Table 4.1. The server will keep a comprehensive patient record. However, this record contains the links to data items in a variety of information systems, and not the data items themselves. The server has the sorting capability to search and organize a collection of these links as a view of the patient record. This view is presented as an HTML document.

b. CompChart Client: It has the View Generator, a browser and a set of players. The View Generator composes a description of the desired Combined View and sends it to the CompChart Server. The browser displays the view assembled by the Server, and the players display the patient data elements.

The CompChart system is based on extensive information discovery:

a. Discovery of the information systems: The information about the systems configuration is added by hand by a system administrator due to the low frequency of adding information systems. The CompChart Server keeps a table of the configurations as presented in Table 4.1.

b. Discovery of patient data: The information about patient data elements is gathered with the help of several agents. The role of these agents varies depending on the cooperation exhibited by different information systems for data discovery.

   (a) Polling Agents: Periodically ask the information system about new events.

   (b) Receiving Agents: Continuously listening for a particular data stream coming from cooperating systems. In this case the information system is notifying the CompChart Server about all of its events as they occur.

   (c) Query Agents: These are spawned to search for a particular piece of data. They will interrogate different systems until the data is found, or the search is exhausted.

The steps for accessing a patient record are depicted in Figure 4.2.4. First, the user selects the desired view components of the record using the CompChart View Generator, and hits the 'generate' button. The Server searches the patient record for the desired view components and sorts them. Second, the Server generates an HTML document where each

Table 4.1: Table of Information Systems Configuration

System	Address	Process	Participation	Interface	Comm
Hospital Information System	raptor.lpl. arizona.edu	his	Poll	HL7	DCE RPC
Radiology Information System	gpacs2.ece. arizona.edu	ris	Advertize	DICOM	TCP

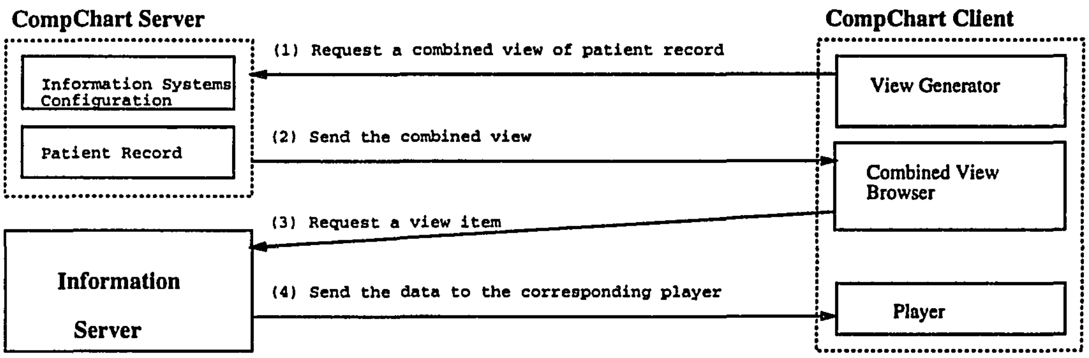


Figure 4.14: Information Flow within the Comprehensive Chart System Architecture

link in it is based on the configuration of the information systems. Third, the Client browses through this Combined View and clicks on a view item. The request is forwarded to the corresponding information server. Fourth, the information server sends the requested data to the client, which spawns a player to display it. A more detailed description of the steps for accessing a patient record are depicted in Figure 4.2.4.

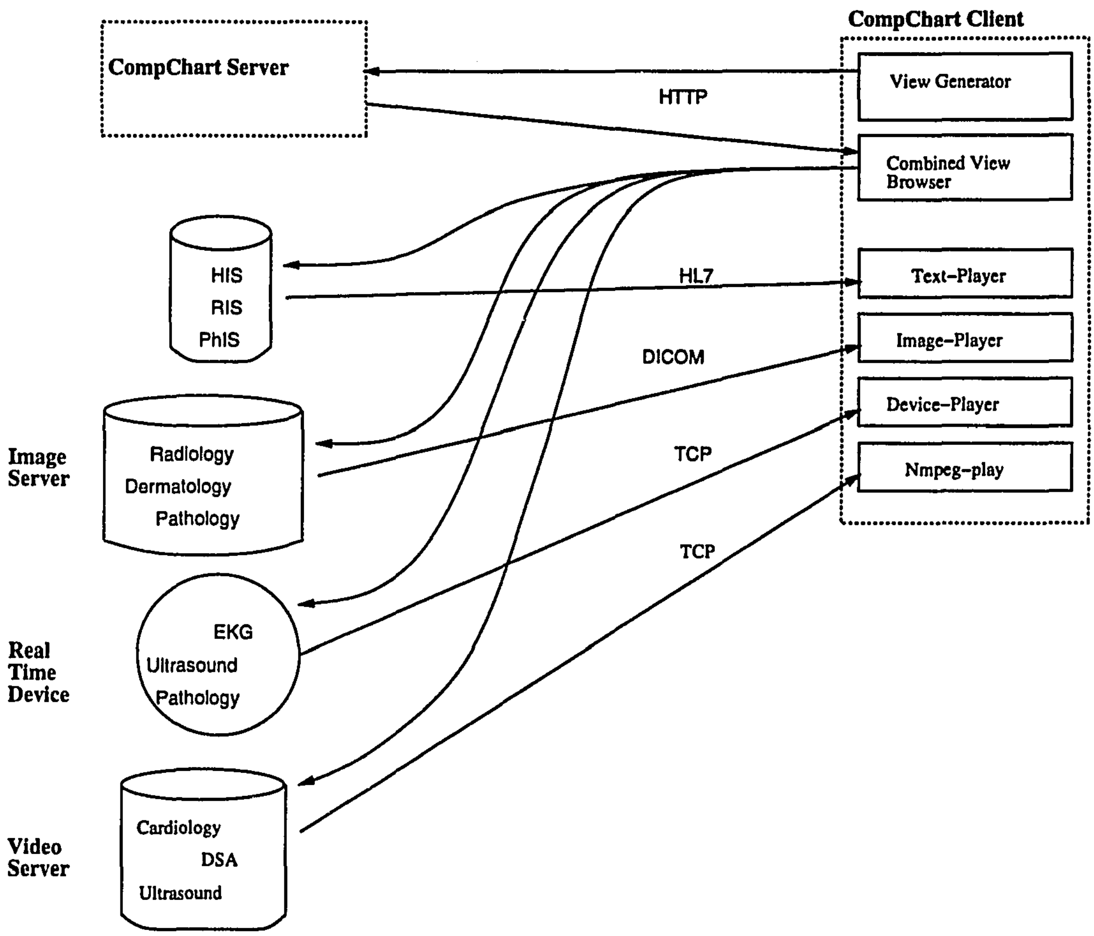


Figure 4.15: Comprehensive Chart System Architecture

4.3 Mapping DCE Services to Global PACS

Mapping Global PACS as a DCE environment will make heterogeneous hardware platforms distributed over multi-networks appear as one seamless environment. This will provide true multi-vendor interoperability and enable scalability of prototype systems to commercially available systems. Some design considerations in this mapping with a simple configuration are depicted in Figure 4.16 and discussed below.

4.3.1 DCE Cell Directory Service in Global PACS Environment

a. Global PACS DCE Cell Components:

The minimum configuration of a Global PACS DCE *cell* consists of three servers: the Cell Directory Service server, the Security server, and the Distributed Time Service server. In case of cells containing a large number of systems, replicas of the CDS and security servers can be created to improve performance and reliability. Global PACS components are divided into clients and servers. A Global PACS client can be a workstation or an imaging equipment. A Global PACS server can be the database archive system serving Global PACS clients' requests. However, in a DCE environment, all Global PACS components are mapped as DCE users. This means they will act as clients to DCE services.

b. Global PACS DCE Cell Assignment:

A hospital consisting of several departments may choose a DCE configuration that assigns a DCE cell per department. This assignment reflects the choice of each department to have administrative control of its resources. The Global Directory Agent

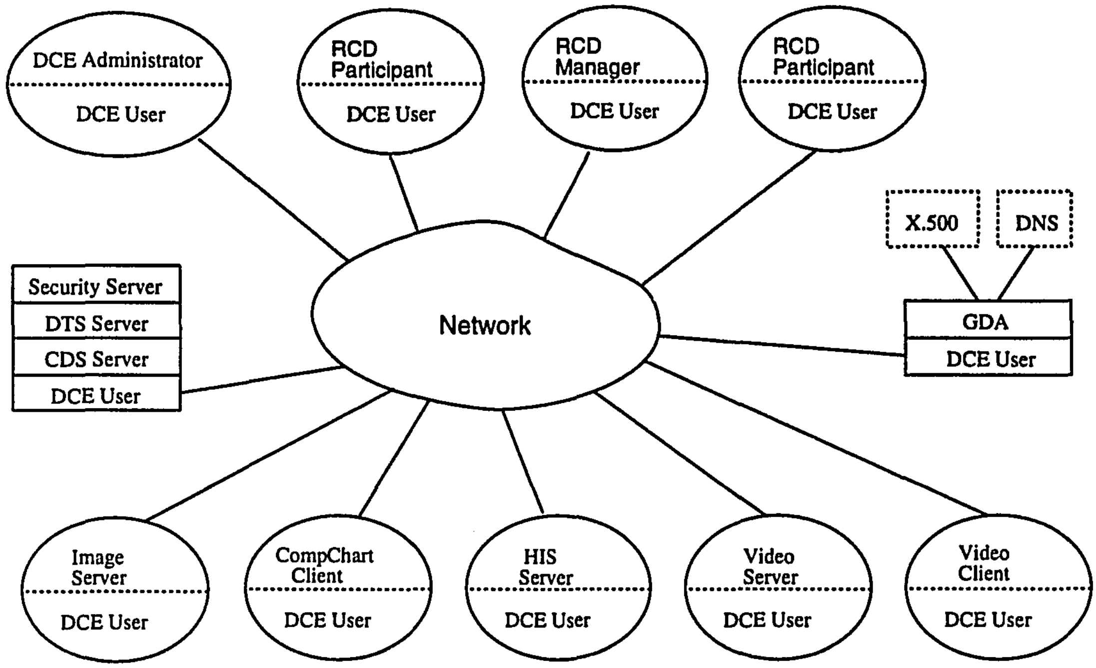


Figure 4.16: Simple Global PACS DCE Cell Configuration

(GDA) enables these departments/cells to communicate with each other, or with DCE cells at other hospitals or with non-DCE systems. The GDA is an intermediary allowing the use of:

(a) DCE Global Directory Service (GDS), which is an implementation of the X.500 directory service standard,

(b) Domain Name Service (DNS) as a global directory service.

To help RCD Participants find other participants or RCD Service Providers, the RCD application uses a namespace. The namespace is maintained by the DCE CDS. The use of CDS is convenient for large RPC environment (e.g., hospital). The initial overhead of understanding and configuring a directory service is balanced by easier management over time. The directory will keep track of *binding* information, which is a set of information that identifies a server to a client or a client to a server. In the RCD system, binding information is managed by two methods:

1. Implicit Method:

   The implicit method is used for managing bindings among RCD Participants. When a participant starts up, it registers itself with the CDS server. Thus, participants can ask the CDS server for the binding information of a particular participant. Earlier versions of RCD system provided a session between two particular host addresses. Through the use of the CDS server, a participant does not need to know the host address of the other participant, only the name of the participant will suffice. For example, a participant may request establishing a session with Dr. Martinez, instead of establishing a session with gpacs2.ece.arizona.edu. One advantage of using the implicit method here is that it allows participants to move dynamically across machines. A participant can easily get the dynamic binding information which is needed for the communication of voice, video, and image annotation.

2. Automatic Method:

   The automatic method is used for managing binding information between RCD Service Providers and RCD Participants. The RCD Service Provider exports its binding information to the namespace. When a participant issues a remote procedure call to a service, its client stub automatically manages a binding for the RPC code. The advantage of using the automatic method is that it completely hides binding management from the RCD code. Another advantage is that a disrupted call can sometimes be automatically rebound.

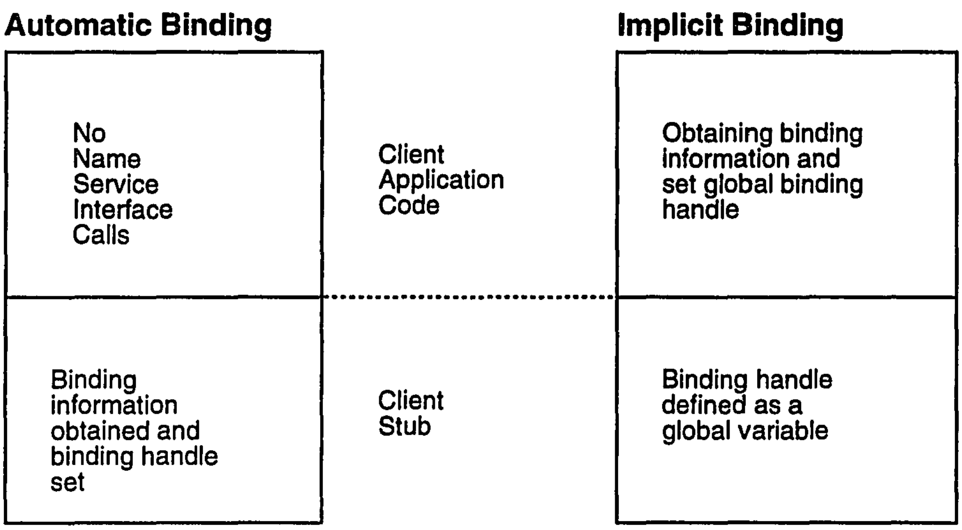
4.3.2 User Directory Server Using DCE Services

In Global PACS users move from one computer to another throughout a medical organization. That is their host machine and its address is frequently changing. The User Directory program keeps track of their location, relieving users from having to worry about this low-level detail. This server is designed based on the implicit binding method previously described. The server keeps a database of Global PACS participants. Every time a participants starts a Global PACS application, it registers the participants' coordination in a table, as depicted in Table 4.2. This server registers two end points with the RPC runtime services:

```
Add_User_Table ();     /* it adds the user name, host address,  */
                       /* and login name to table */
Get_User ();           /* given the user name, get the usr login*/
                       /* and host address */
```

4.3.3 DCE Distributed Time Service in Global PACS Environment

The DTS will provide timing services through its application programming interface (API) as depicted in Figure 4.18. Coordinated Universal Time (UTC) is widely used and is disseminated throughout the world by various standards organizations. Several



Automatic Algorithm

```
{
Stub automatically
imports binding
handle for a
compatible server
from the name
service database
}
```

Implicit Algorithm

```
setenv
RPC_DEFAULT_ENTRY
/.:/usrdir_server

{
rpc_ns_binding_import_begin ();
rpc_ns_binding_import_next ();
/* now we have binding handle */
rpc_ns_binding_import_done ();
}
```

Figure 4.17: Algorithms for Binding Methods Using DCE CDS

Table 4.2: User Directory Table of Participants

Participant Name	Host Address	Login Name
Ralph Martinez	raptor.lpl.arizona.edu	ralph
Yasser alSafadi	gpacs1.ece.arizona.edu	yasser

manufacturers supply devices that can acquire UTC time values via radio, satellite, or telephone. These devices can then provide standardized time values to computer systems. the interface between the DTS server process and the time-provider process is called the Time-Provider Interface (TPI).

Global PACS will operate in UTC with Time Differential Factor TDF = 0 (the "Z" ("Zulu") or "UTC reference" time zone, corresponding to and generally equivalent to the classical UT Greenwich Mean Time (GMT) time zone), not local civil time, because participants in Global PACS may be in different time zones. UTC is useful for measuring time across local time zones and for avoiding seasonal time changes (summer time or daylight savings time). DTS provides time stamp manipulation to client applications, as described in the following scenarios.

1. The exchange of voice and image annotation data:

   For synchronization of voice and data, the RCD system uses a *time-stamp* mechanism. When voice or annotation data packets are transmitted, a time-stamp is attached to each packet. The data and voice processes at the receiver side communicate to compare the time stamps of the received packets to achieve synchronization during the playing of the two different streams.

2. The recording and playback of a session:

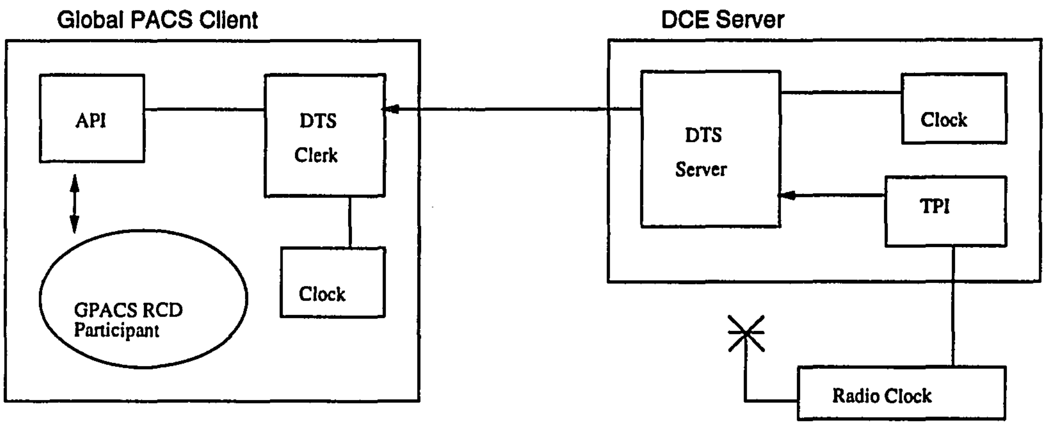
   An RCD session on a particular image can be stored as a study session for future reference or playback. This will store the static image under study, in addition to the time-stamped voice and image annotation data being exchanged. Playback processes retrieves this information and uses the time stamps to reply the session maintaining the spatial characteristics of the image annotation and the time occurrence of the voice packets.

4.3.4 DCE RPC in Global PACS Environment

The RCD system follows the client/server model. It uses the RPC mechanism over a connection oriented protocol (TCP/IP) to handle participant's requests to the service providers. However, interactive data (video, voice) are exchanged between participants using a connection oriented protocol transfer.

Global PACS applications using RPC as a communication mechanism need not be rewritten when ported to different architectures, operating systems, communication protocols, or programming languages.

Depending on the relative data volume, two types of data are transmitted in a Global PACS environment: small and large. Small volume data (1-10 KByte) such as patient information, image annotations, and control commands during a remote consultation session. These data can be exchanged among Global PACS nodes (clients, servers) using a



API: Applications Programming Interface

TPI: Time Provider Interface

Time Service API for timestamps

utc_gettime ()	/* get the current system time as an opaque binary timestamp*/
utc_gmtime ()	/* converts binary timestamp into a tm structure that expresses GMT or the equivalent UTC */
utc_subtime ()	/* compute the difference between two binary timestamps */

Figure 4.18: Global PACS DCE Time Services

straightforward DCE RPC mechanism. However, large size data (1-10 MByte), such as medical images, can be transferred using a *pipe* abstraction supported by Interface Definition Language (IDL). DCE pipes are a mechanism on top of DCE RPC for transferring large quantities of data. In a Global PACS application, a pipe can be established between a Global PACS server and a client to supply the requested image.

4.3.4.1 Setting up a Global PACS DCE server

The steps leading to the establishment of a Global PACS DCE server are depicted in Figure 4.19. First, the server will set the type of RPC object with the RPC runtime. It registers the interfaces and selects the communication protocol sequences. Then, it obtains the binding handles and registers the end points. At this point, the server exports the binding information to the name space and waits listening for incoming calls.

4.3.4.2 Using DCE RPC for the Exchange of Image Annotation

This design illustrates the use of DCE RPC for the exchange of image annotation commands. Utilizing a *remote consultation manager* is part of the scalable approach to Global PACS distributed system design. The remote consultation manager coordinates the exchange of framing information (image annotation) among consultation session members, as shown in Figure 4.20. In the design, two clients were assumed. However, it can be extended to a number of participants (up to 10) without the n need to rewrite the application. The remote consultation manager was designed as an application server. When a server program starts up, it registers itself with the CDS server, so client applications only need to know the name of the service, not the name of the server. When a remote consultation

```
rpc_object_set_type ();            /* setting the type of RPC OBJECT
                                      with the RPC runtime */

rpc_server_register_if ();         /* Register RPC interfaces */

rpc_usr_all_protseqs ();           /* Selecting RPC protocol sequences */

rpc_server_inq_bindings ();        /* Obtaining server binding handles */

rpc_ep_register ();                /* Registering end points */

rpc_ns_binding_export ();          /* Exporting binding information to
                                      namespace */

rpc_server_listen ();              /* Listening for calls */

/* Now waiting for remote procedure calls */

/* shutdown the server */
rpc_epc_unregister ();
```

Figure 4.19: DCE Service Calls to Setup a Global PACS Server

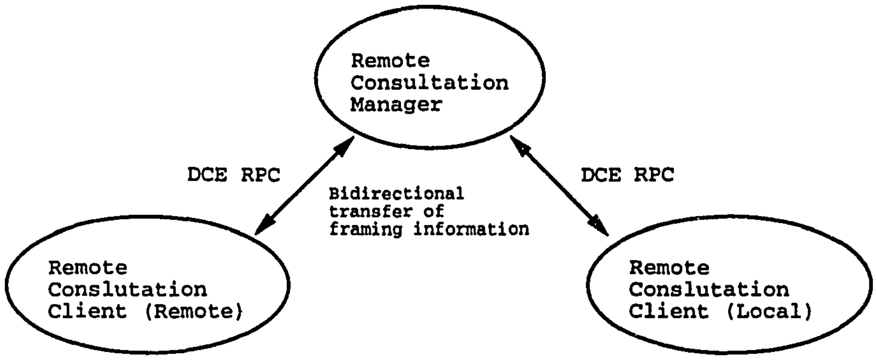


Figure 4.20: Using DCE RPC for the Exchange of Image Annotation

client starts up, it asks the CDS server for the location of the server providing the servers' end point mapping address.

4.3.4.3 Using DCE RPC PIPEs for Image Transfer

As previously indicated, a pipe abstraction is an efficient way for sending large amounts of data. Here is a description of a program designed to transfer images using DCE RPC PIPE abstraction [65]. Figure 4.21, depicts the code structure and algorithms for this design. This interface consists of three main components:

1. *allocate:* The *allocate* procedure allocates a buffer for incoming data at the client side, or points to a buffer at the server side.

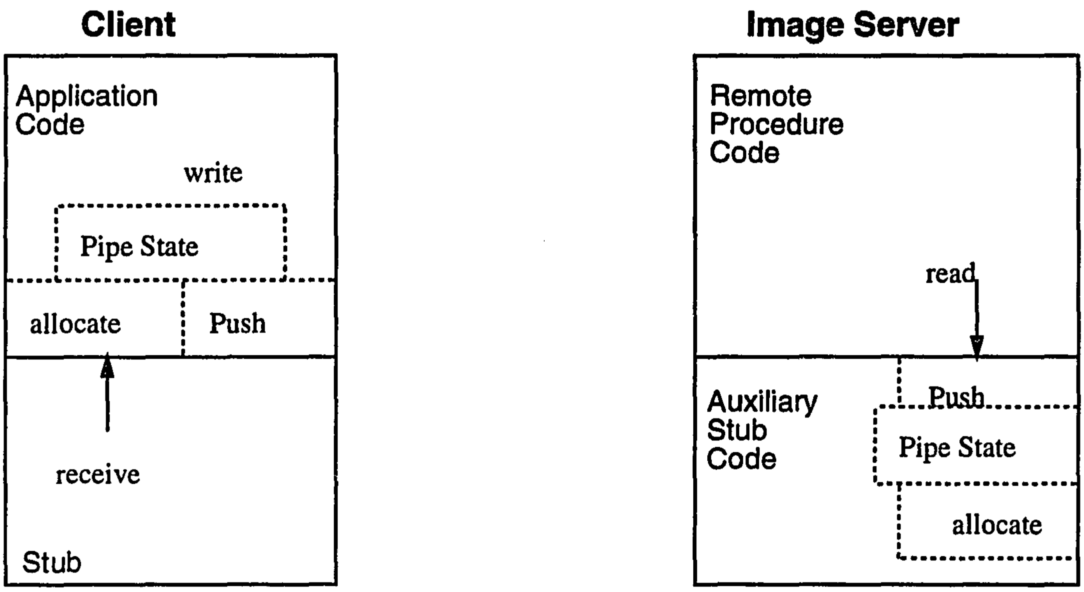
2. *push:* The *push* procedure is used for for the output pipe. At the client side it writes the data to a file, and at the server side it transfers the data from the server to the client.

3. *Pipe State:* The pipe state is application specific and is used to communicate between the application code and the stub code. This structure keeps track of where to find or place the data.

```
typedef struct pipe_state
{
   int filehandle;  /* client data file handle */
   char *filename;      /* client data file name */
}
 Pipe_State;
```

A pipe to receive images from a file on the server is defined in IDL as:

```
/* This describes transfer_images.idl */
interface transfer_images /* image transfer from a remote system */
{
   typedef pipe char pipe_type;  /* define the base type as char */
   void receive_images( /* get pipe from server */
        [out] pipe_type *data; /* output pipe of char data */
   );
}
```



Client Algorithm

```
repeat
{
 receive data;

 allocate buffer;

 push to write buffer;
}
until (data of zero size);
```

Server Algorithm

```
repeat
{
 read data from file;

 push to transfer data;
}
until (end of file);
```

Figure 4.21: Algorithms for Image Transfer Using DCE RPC Pipe

4.3.5 DCE Security Service in Global PACS Environment

Setting a high *protection level* for a Global PACS application means authenticating every message or packet and encrypting all user data. Setting a lower protection level means performing authentication at the time of establishing sessions only. The highly secure Global PACS DCE application will pay a high price in terms of performance due to excessive CPU utilization by the security mechanisms (especially encryption, decryption). More reasonably, the Global PACS DCE application would perform authentication at the beginning of sessions. It would protect the transfer of sensitive data, such as patient information and diagnosis, and send "in the clear" other data such as image annotation and session control commands. In some cases medical images need to be protected. However, more research is needed to further identify these cases.

An application using authenticated RPC may specify one of the following *protection levels*:

1. Connect Level: Performs authentication only when a client and a server establish a relationship. The data sent between the client and server is not encrypted.
2. Call Level: Attaches a verifier to each client call and server response. However, this level does not apply to RPC calls made over a connection-based protocol response.
3. Packet-Integrity Level: Ensures that none of the data transferred between the client and the server has been modified in transit. This is done by computing a checksum.
4. Packet-Privacy Level: Incorporates lesser protection levels and in addition encrypts all RPC argument values.

Intuitively, the more restrictive the protection level, the greater the impact on performance. This implementation of DCE Security for RCD uses Kerberos environment and the Data Encryption Standard (DES) for authentication and data privacy. The Connect Level protection is used when obtaining binding information about participants. However, to maintain patient's privacy, the Packet-Privacy Level is used when obtaining patient information.

The Access Control List (ACL) specifies the set of permissions granted to a system user, that is, the system services and data sets the user is authorized to access. Current design allows two group of users:

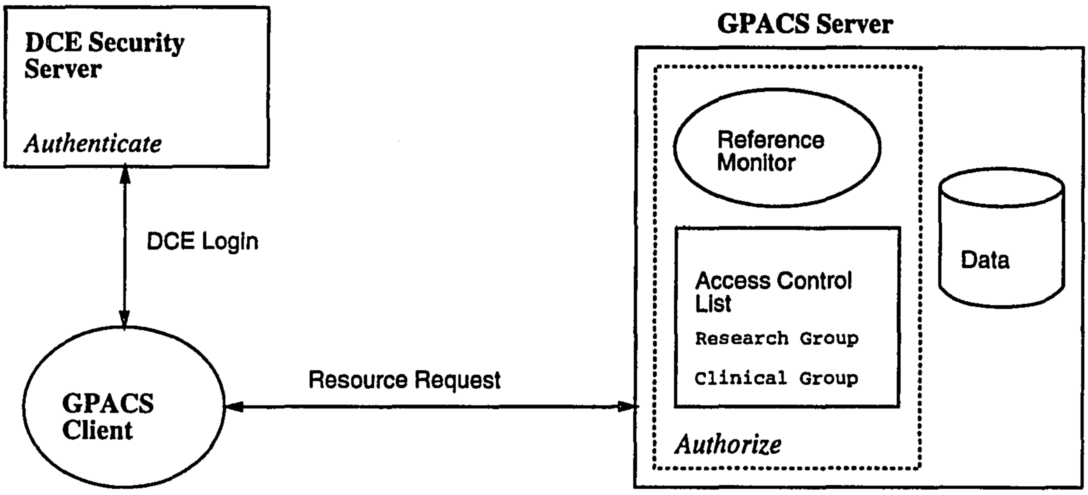
1. Research Group: has access only to pseudo-patient records, and phantom images offered by RCD Service Providers (e.g., database).

2. Clinical Group: Has access to real patient records and clinical images available in the hospital information system or radiology information system.

Further definition of different groups is possible through the DCE ACL interface.

The steps leading to the use of Global PACS resources are depicted in Figure 4.22. A Global PACS user must perform a DCE login before using the system resources. In DCE login the user communicates with the DCE Security Server to prove his identity. Then, to access a resource such as a Global PACS Server the user contacts the DCE Security Server and get a *certificate* of the user identity. The client passes this certificate to the resource server as part of the DCE RPC call. The Global PACS Server passes the certificate to the Reference Monitor which examines the certificate against its access control list and determines the user's authorization.

It is worth noting that there is a strong relationship between DTS and the security service. The security service depends on a relatively close synchronization of network




```
rpc_binding_set_auth_info ();            /* annotate binding handle for
                                            authentication */

rpc_binding_inq_auth_client ();          /* Check the authentication
                                            parameters of this client */

Reference_Monitor
{
  search_acl_entry ();                   /* traverse the entry list */
  locate_matching_acl_entries ();        /* locate entries that matches the
                                            client's security attributes */
  perform_group_check ();                /* check group in order to grant
                                            permission */
}
```

Figure 4.22: Using DCE Security Services for Accessing Global PACS resources

```
typedef struct
  {
    sec_acl_id_t        default_realm;  /* The default cell */
    uuid_t              sec_acl_manager_type;
    unsigned32          num_entries;
    [ptr, size_is(num_entries)] sec_acl_entry_t  *sec_acl_entries;
  } sec_acl_t;


typedef struct
  {
    sec_acl_permset_t  perms;
    union sec_acl_entry_u
          switch (sec_acl_entry_type_t entry_type) tagged_union
              {
                  case sec_acl_e_type_mask_obj:
                  case sec_acl_e_type_user_obj:
                  case sec_acl_e_type_group_obj:
                  case sec_acl_e_type_other_obj:
                  case sec_acl_e_type_unauthenticated:
                  case sec_acl_e_type_any_other:

                  case sec_acl_e_type_user:
                  case sec_acl_e_type_group:
                  case sec_acl_e_type_foreign_other:
                                                   sec_id_t    id;

                  case sec_acl_e_type_foreign_user:
                  case sec_acl_e_type_foreign_group:
                                     sec_id_foreign_t  foreign_id;
                  case sec_acl_e_type_extended:
                  default:
                        [ptr] sec_acl_extend_info_t * extended_info;
              } entry_info;
    } sec_acl_entry_t;
```

Figure 4.23: DCE ACL Entry

clocks, a service provided by the DTS. When network clocks become too skewed, unexpired tickets (permissions) to services may be regarded as invalid. This will lead to the denial of services to legitimate requests.

4.3.6 DCE Threads in Global PACS Environment

Global PACS servers and clients in a DCE environment lend themselves to being structured as multiple flows of control. A Global PACS server can benefit from DCE Threads to satisfy multiple RPC requests from different clients concurrently. Additionally, Global PACS servers perform mainly input/output operations to archival and storage devices. Some threads will be blocked waiting for an input or output operation while other threads can continue working. A single threaded Global PACS client making an RPC call will block until a response is returned from the Global PACS server. With the availability of multiple threads of control, the Global PACS client can work on another thread while the RPC thread is being blocked. This will lead to better performance on Global PACS clients and servers.

The RCD Service Providers make use of the Threads Facility. This allows a service provider to respond concurrently to RPC requests from participants engaged in a number of RCD sessions. This leads to a performance advantage since these RPC share access to the RCD service (e.g., a handle for an open database) in a single address space. During initialization phase, the server determines the number of threads it needs to process remote procedure calls.

```
rpc_server_listen ();     /* sets the number of threads for rpc calls */
```

Clients wishing to make multithreaded procedure calls should use *explicit binding* methods to bind to the desired servers. Using this approach, the client can control the distribution of load among server.

4.3.7 Global PACS DCE Environment and International Character sets

As global PACS becomes truly international, it will uses a variety of character sets. If data of a certain type is to be treated as international characters (I-chars), then that type must be given the attribute cs_char in the ACF. This attribute takes as an argument the local type to be used for this data. For example:

```
/* The IDL file includes the declaration
   typedef byte my_byte                    */
typedef [cs_char(l_type)] my_byte;
```

When data of this type is marshaled or unmarshalled, codeset conversion routines are invoked. ltype is the local type used for the data by the application.

CHAPTER 5

Experiments and Results

5.1 Global PACS DCE Testbed

The Global PACS testbed using OSF DCE demonstrates the utility of DCE services and facilities. These services are coupled with the HL7, DICOM, Scalable Network Video interfaces to support Global PACS applications. The following subsections describe the Global PACS DCE platform, the Comprehensive Chart testbed, and the Remote Consultation and Diagnosis system testbed.

5.1.1 Global PACS DCE Platform

We have developed an OSF DCE testbed for application to Global PACS. This testbed currently operates on the University of Arizona campus Ethernet. Eventually, the testbed will be expanded to Internet nodes. The prototype uses two DEC 3000/300 AXP 300 workstation running DEC OSF/1 AXP Version 1.1 and Digital DCE Version 1.1. These machines are located at different campus locations and are connected via the University of Arizona campus network. Digital DCE consists of a subset of the full DCE as defined by the OSF. We ran the minimum DCE configuration of three servers: A Cell Directory Service server, a security server and a Distributed Time Service server. These servers ran together on the same machine. All of the platforms had the DCE executive running which is the fundamental part of DCE that is required to participate in a DCE environment. All of the images used in the testbed were compressed in JPEG encoding. All of the dynamic sequences were compressed in MPEG-1 encoding.

5.1.2 Comprehensive Chart Testbed

This testbed illustrates the mapping of DCE services and the standardized interfaces to support the CompChart system. The CompChart testbed machines and communication protocols are depicted in Figure 5.2. The CompChart makes extensive use of DCE Services, as shown in Figure 5.3. First, the client is authenticated using DCE Security to check of its true identity. This client binding is held by the DCE Directory Service using the Implicit Method. Second, the client gets a view of the patient record from the CompChart server formatted as an HTML document. The client uses a *Mosaic* browser for this combind view. Let us examine what happens when the client requests an item from the combined view.

1. Request a HIS report: The client requests a HIS report by clicking on a Text Report Icon. The DCE RPC at the Packet Privacy Level (that is the highest security level) is the transport mechanism. The HIS Server endpoint bindings are held by the DCE Directory Service using the Automatic Method. The request is checked for authorization by using DCE Security Services. If the Access Control List indicates that the client is authorized to receive that peice of information, the response is generated. The DCE RPC at the Packet Privacy Level brings back the response. The CompChart client spawns a text player to display the requested HIS report.

2. Request an image: The client requests an image by clicking on a Modality Icon, such as XRAY. To retrieve the image the client goes through similar steps to retrieving a

Table 5.1: Average Time for Image Transfer (seconds)

	Image Size (Mbyte)				
Mechanism	**1**	**2**	**4**	**8**	**10**
TCP socket	2.33	4.67	7.91	17.77	20.72
DCE PIPE	4.93	9.91	19.45	37.61	47.43

HIS report. That is, the client's request is checked for authorization at the server. However, in this case, the client contacts the Global PACS Image Server. It uses the DCE RPC Pipe abstraction to deliver the images at the Connect Level security. The image player at the client is the public domain *xv* program which subsequently displays the image. The average measure time for the transfer of an image using regular TCP connection versus the DCE PIPE mechanism is presented in Table 5.1.

3. Request a dynamic sequence: The client requests a dynamic sequence by clicking on a dynamic Modality Icon, such as ECHO (for an Echo Cardiogram). To retrieve the desired dynamic sequence the client's request is checked for authorization at the Video Server. If the client is authorized, the Video Server sends the dynamic sequences as explained earlier in the Scalable Network Video delivery section. The *Nmpeg-play* plays these sequences at the client's workstation.

The DCE Threads package is part of the minimum cell configuration since OSF's DCE technology components assume the availability of threads support. The servers use the DCE Threads package to handle simultaneously multiple requests.

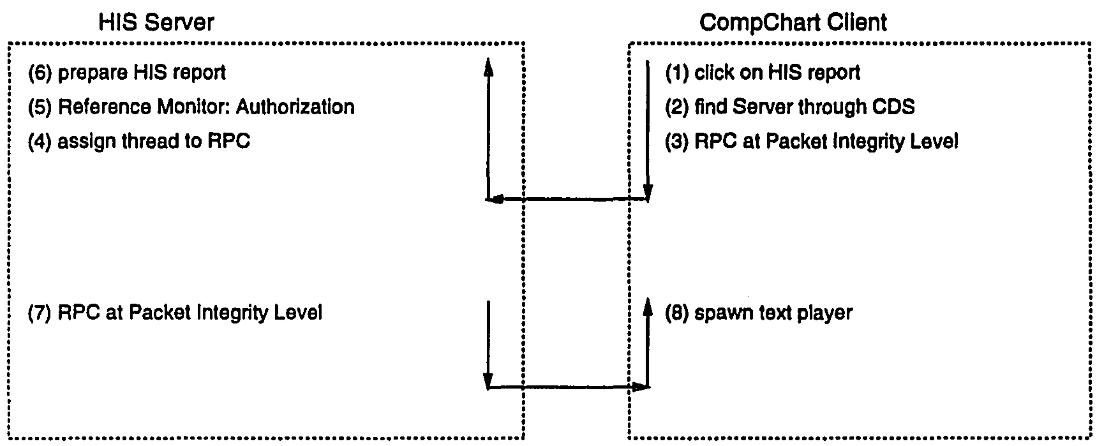


Figure 5.1: Requesting a HIS Report in Global PACS CompChart

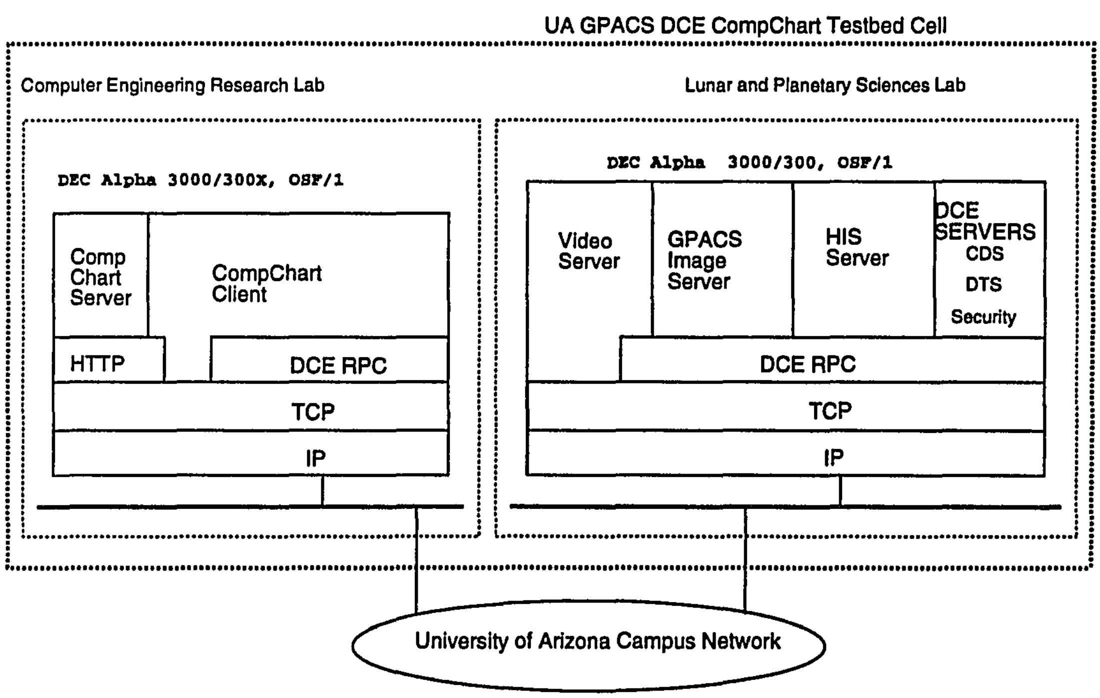


Figure 5.2: Prototype Global PACS CompChart Testbed Using DCE Architecture

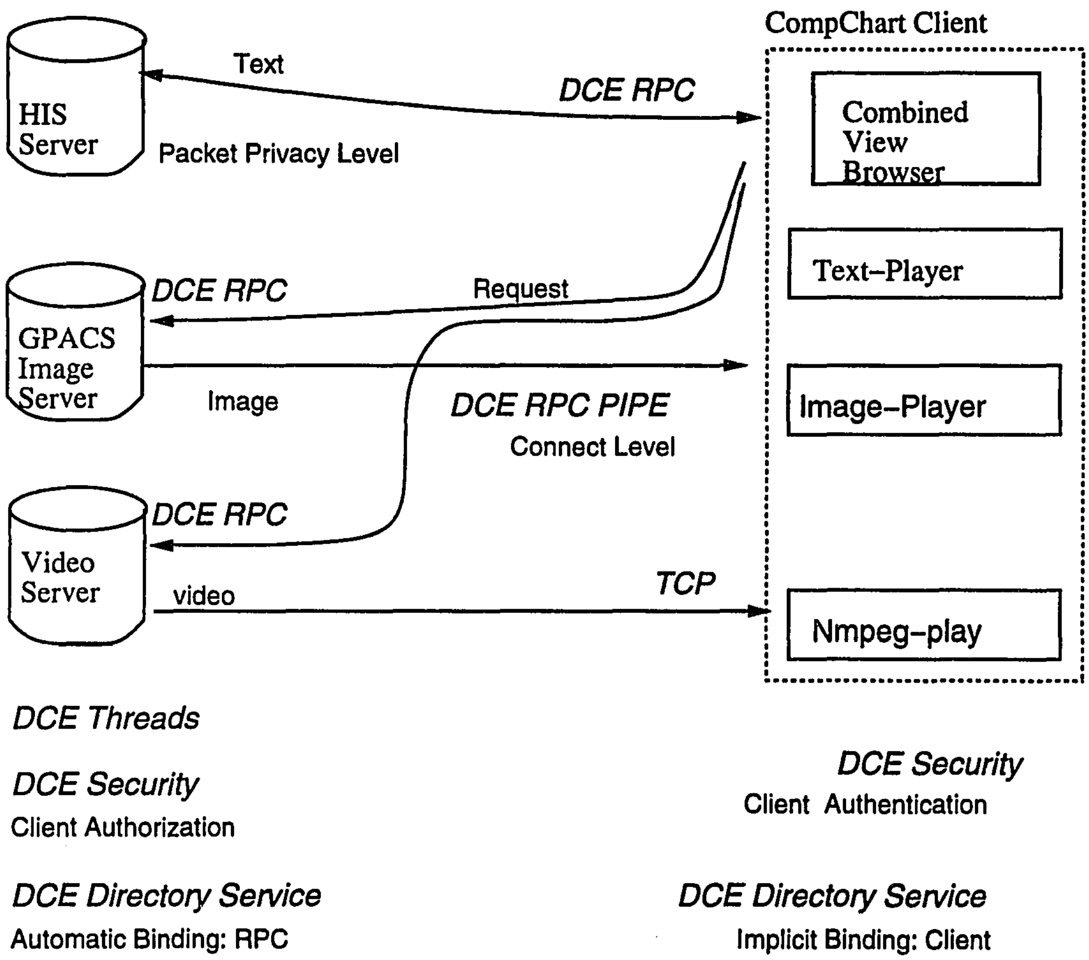


Figure 5.3: Mapping DCE Services to Comprehensive Chart Testbed

5.1.3 Remote Consultation and Diagnosis Testbed

A portion of the Global PACS RCD developed by Martinez [2] has been mapped to a DCE environement. This prototype illustrates the mapping of DCE services and the standardized interfaces to support the RCD system. The RCD testbed machines and communication protocols are depicted in Figure 5.5. The RCD system makes extensive use of DCE Services, as shown in Figure 5.3.

First, the participant is authenticated using DCE Security to check of its true identity. This participant binding is held by the DCE Directory Service using the Implicit Method. Then the participant requests the bindings of the other participant in the RCD session. Both participants request an image from the Global PACS Image Server. Each establishes an association with the server. The request is checked for authorization by using DCE Security Services. If Access Control List indicates that the participant is authorized the server sends the image to both. It uses the DCE RPC Pipe abstraction to deliver the images at the Connect Level security. Now, both participant can exchange image annotations over the DCE RPC at the Connect Level security.

The average measured time for the exchange of an image annotation between the two systems using UDP is $3.5ms$. The average measured time using DCE RPC is $24ms$. This is understandable since remote procedure calls involve data conversion and possibly a contact with the directory service during RPC runtime to determine end-points of consultation manager.

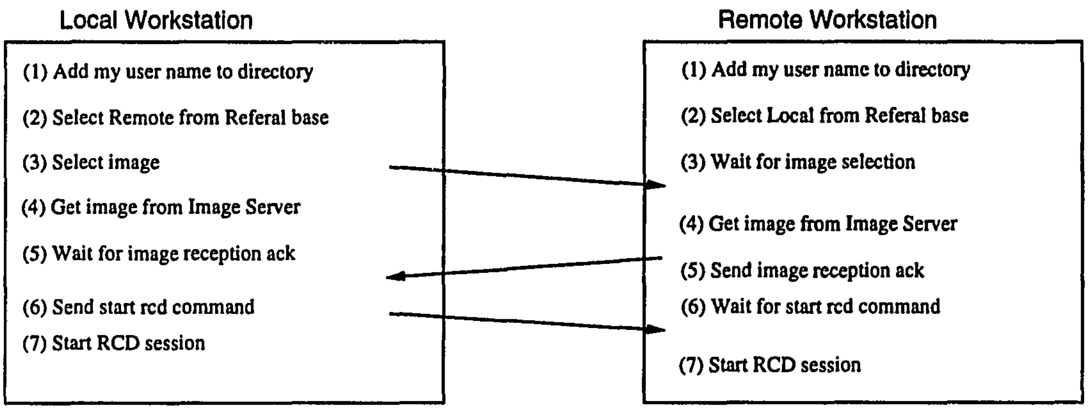


Figure 5.4: Steps for a Session in Global PACS RCD Using DCE Architecture

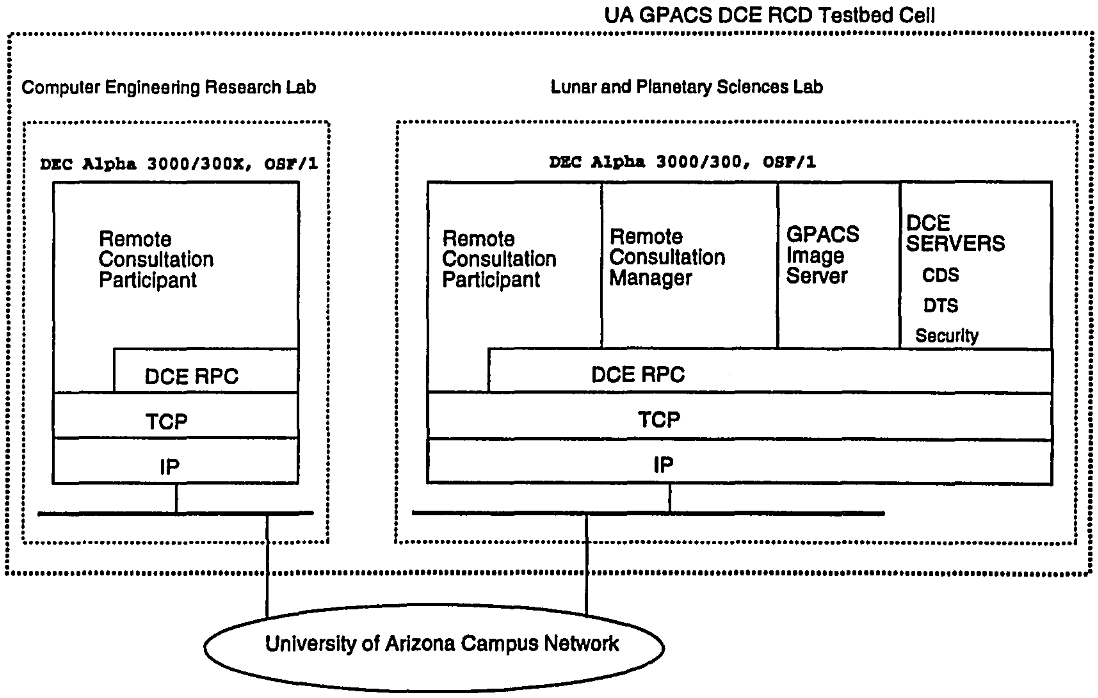


Figure 5.5: Prototype Global PACS RCD Testbed Using DCE Architecture

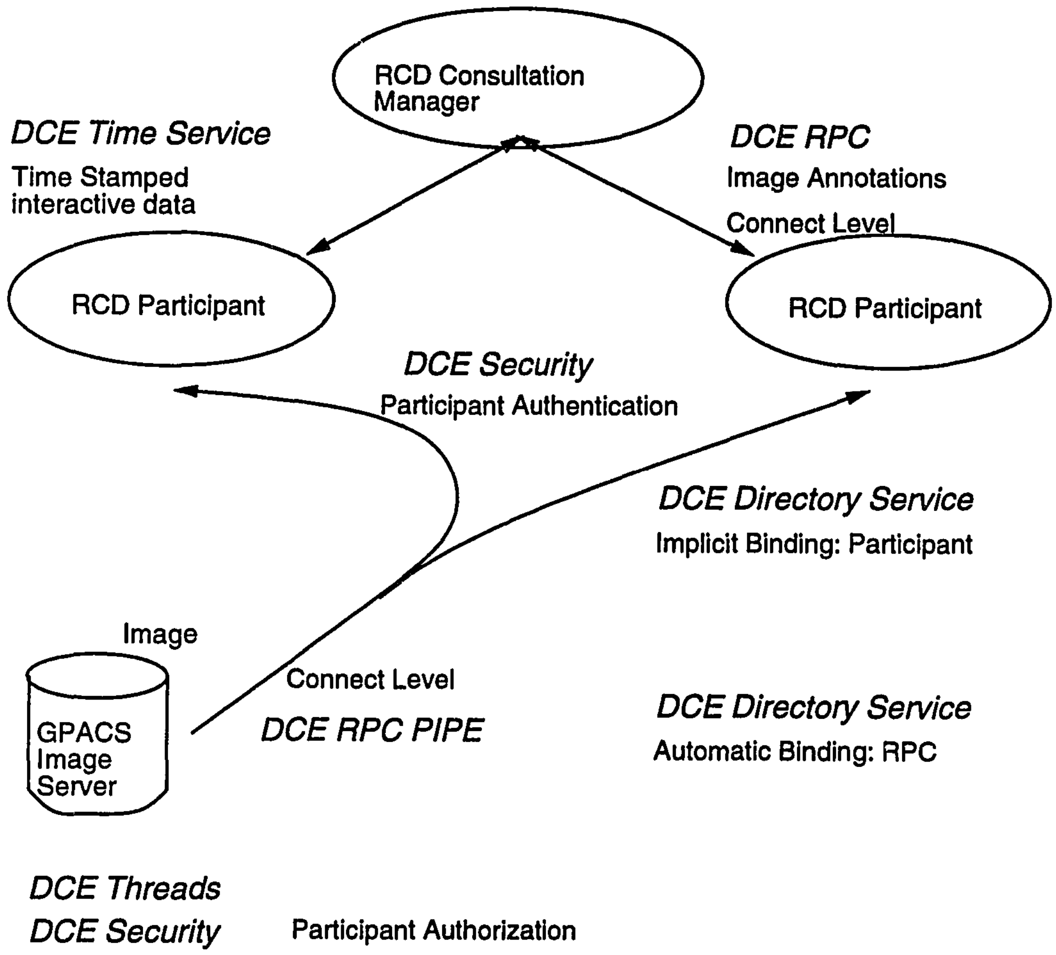


Figure 5.6: Prototype Global PACS RCD Testbed Using DCE Services

5.2 SNV Experiments

5.2.1 Experiment Platforms

The experiments used these platforms:

a. Client: Sun SPARCstation IPC running SunOS 4.1.1 and 8bit display, 25 Mhz clock cycle and 24 Mbyte RAM (flyer.cs.arizona.edu).

b. Client: Silicon Graphics IRIS 4D / 340 VGX running IRIX 4.0.2 and 24bit display, 33 Mhz clock cycle and 64 Mbytes RAM (vochelle.cs.arizona.edu).

c. Video Server: Sun SPARCstation 10 running Solaris 2.3 and 8bit display, 150 Mhz clock cycle and 32 Mbyte RAM (gpacs1.ece.arizona.edu).

d. Client: DEC Alpha 3000/300X running OSF/1 and 8bit display, 175 Mhz clock cycle and 32 Mbyte RAM (gpacs2.ece.arizona.edu).

e. Video Server: DEC Alpha 3000/300X running OSF/1 and 175 Mhz clock cycle and 32 Mbyte RAM (raptor.lpl.arizona.edu).

The test videos were carefully selected to represent a wide variety of video scenarios. The tests used to determine the effect of video delivery mechanism on the playing frame rate consisted of eight files, as in Table 5.2: Bear1, Bear2, Bird, Flower1, Flower2, Simpsons, Still1, and Still2. The tests used to determine the time to start viewing a video file consist of relatively large Ultrasound video files (1-60 Mbyte), as in Table 5.4.

The video player *Nmpeg-play* is based on *mpeg-play* program version 2.0 from the University of California in Berkeley. It was modified in several ways to make the network connection possible, while maintaining all the options as before. With the additional option *-host* the user specifies that the data should be read from the network and provides the server address. All programs were compiled using gcc, version 2.5.7 with optimization flag -O2, and the code was not stripped. An Imakefile was used to determine which libraries to link dynamically to make the program portable to all experiment platforms. The addition of video support does not modify the Mosaic client and does not require recompilation of the Mosaic program. Simple editing of configuration files suffices, such as adding to file .mailcap the following line:

application/Nmpeg; Nmpeg_play %s

and to file .mimetype the following line:

application/Nmpeg Nmpeg

5.2.2 Experiment Design

The experiments were conducted to the compare current Mosaic mechanism with the SNV mechanism. The first experiment measured the video playing rates using different video delivery mechanisms. The second experiment measured the response time to start viewing video. The following sections explain these two experiments.

I. Frame Rate Experiment

Three experiments were conducted to measure the frame playing rates using the eight benchmark files previously described, as shown in Figure 5.7:

(i) **Local:** This experiment is depicted in Figure 5.8, and tests the Mosaic mechanism. The *Nmpeg-play* program and the video benchmark files are in the /tmp directory of the same workstation. The *Nmpeg-play* reads the files, decodes the MPEG sequences, and displays them on the local display (i.e., the three operations are performed on the same machine).

(ii) **Remote**: This experiment is depicted in Figure 5.9, and tests an alternative mechanism to deliver video data. The *Nmpeg-play* program and the video benchmark files are in the /tmp directory of the video server workstation (Sun 10). We login to this workstation remotely using the rlogin command and set the display to be to the remote workstation.

Specifically, in an X Windows environment [86] the function XOpenDisplay connects an Xlib program to an Xserver [87]. The XOpenDisplay connects to the server specified by setting the environment variable 'DISPLAY' to *'host:server'*, where *host* is the machine name, and *server* is the Xserver number on that machine. The *server* number is always zero on a single-user workstation. The needed command line is , for example:

```
setenv DISPLAY lyra.cs.arizona.edu:0
```

In this experiment, the *Nmpeg-play* reads the local files, decodes the MPEG sequences, and the X Windows system displays them on the remote display. Thus, read and decode are performed on the video server, and the display operation is performed on the remote workstation. As illustrated in Figure 5.7, four variations were carried out: the video server Sun 10 to the client workstation Sun IPC, Sun 10 to the client workstation SGI IRIS, Sun 10 to itself, and DEC Alpha to DEC Alpha.

(iii) **Network:** This experiment is depicted in Figure 5.10, and tests our proposed mechanism. The video benchmark files are in the /tmp directory of the video server workstation (Sun 10), and the *Nmpeg-play* program is in the /tmp directory of a client workstation. The video file transfer program on the video server reads the benchmark file and sends the MPEG stream to the client workstation over a TCP/IP connection. The *Nmpeg-play* gets the sequences from the network connection, decodes them, and display them. In this experiment the reading operation is performed on the video server and the decode and display operations on the client machine. Another setup for this experiment is when the movie server sends live video, and the client decodes and displays it, as depicted in Figure 5.11. The *Nmpeg-play* plays the network MPEG stream as fast as possible. This means that in the case of a slow network connection it would consume the buffers as fast it could and then block waiting for the new data to arrive. So the user may notice a decrease in the frame rate. As in the previous experiment four variations were carried out: the video server Sun 10 to the client workstation Sun IPC, Sun 10 to the client workstation SGI IRIS, Sun 10 to itself, and DEC Alpha to DEC Alpha.

II. Response Time

One experiment was conducted to determine the response time, i.e., the time it takes to start viewing video after making a selection. The experiment used the Mosaic mechanism depicted in Figure 4.2, and the SNV mechanism as depicted in Figure

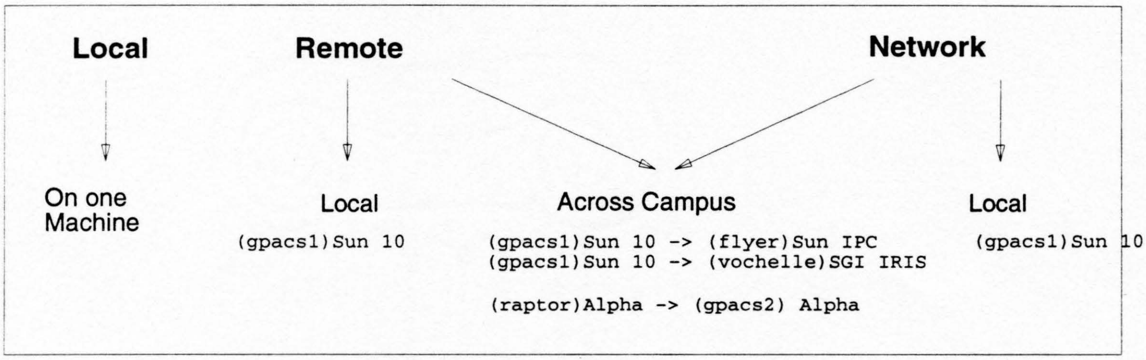


Figure 5.7: Video Delivery Experiments

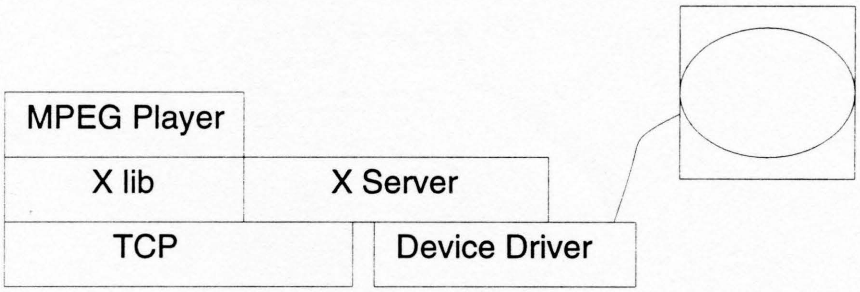


Figure 5.8: Local Experiment: Play The Movie and Display it on The Same Machine

5.10. This experiment used the Sun 10 workstation as a video server and mosaic server, and the Sun IPC workstation as the client.

5.2.3 Measurement Results and Analysis

The following sections present the frame rate experiment and the response time experiment. The video frame rate is: $Frame\ Rate = \frac{Number\ of\ Frames}{Video\ Delay}$. Video delay is given by:

$$Video\ Delay = T_I + T_P + T_N + T_m$$

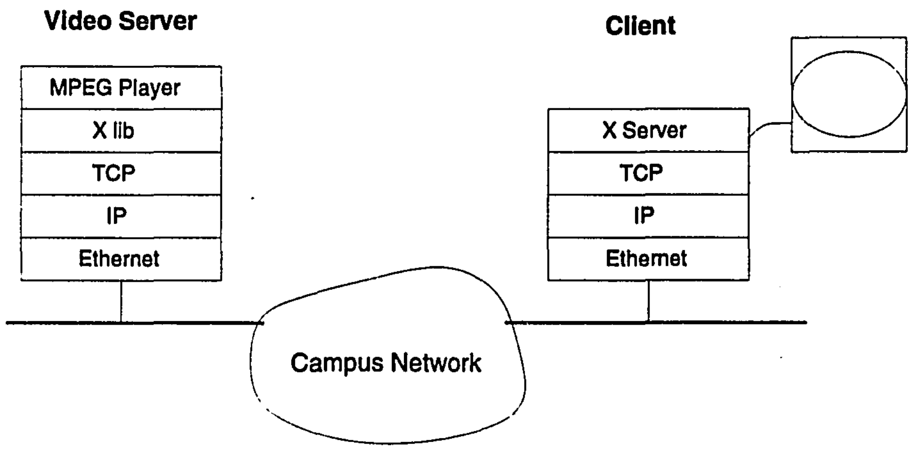


Figure 5.9: Remote Experiment: The Server Plays The Movie, The Client Displays it

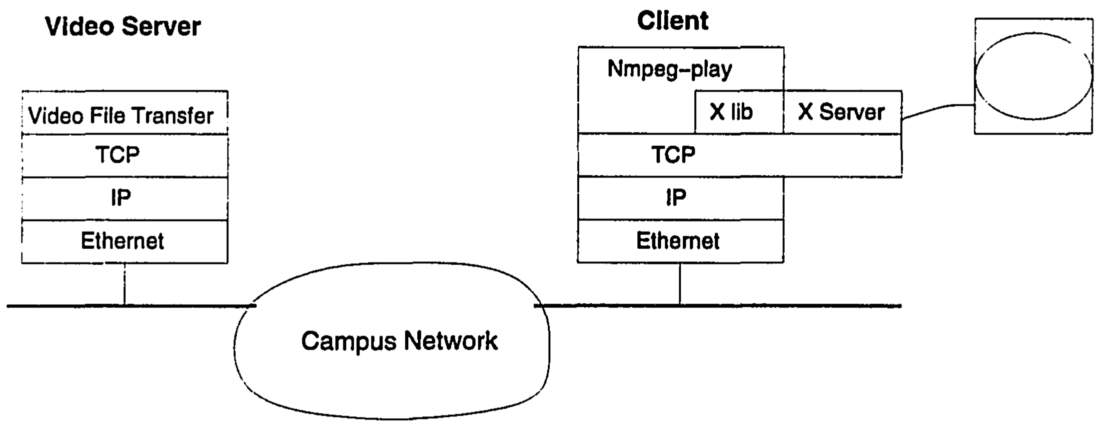


Figure 5.10: Network Experiment: The Server Transfers the Movie, The Client Plays and Displays it

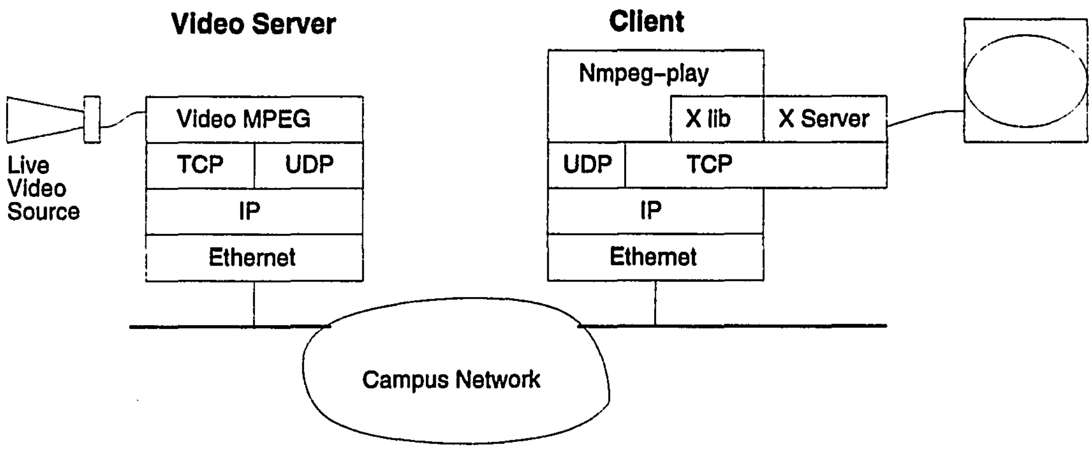


Figure 5.11: The Server Sends Compressed Live Video, The Client Plays and Displays it

where: T_I is the disk I/O, T_P is the communication protocol processing, T_N is network latency, and T_m is the time to perform MPEG decoding.

I. Frame Rate Experiment The measurement results for the playing frame rate are shown in Table 5.2. The values in this table show the mean number of frames per second achieved when playing the benchmarks. The results are as follows:

(i) **Local:** The frame rates reported in Table 5.2 Local section are the Mosaic mechanism rates. In this experiment the computation speed and memory bandwidth on the workstation are the limiting factors [88]. No experiment workstation could play the MPEG streams as fast as their encoding frame rate.

(ii) **Remote:** This approach consumes network bandwidth [89],[90] as it does not benefit from the compression features of MPEG. With MPEG there is usually a compression factor of 50 as compared to raw video data. This approach loses all this compression and unnecessarily increases the network traffic. Another important limitation of this approach is that it places a high demand on the video server to decode the MPEG streams for many clients. Obviously, this approach does not scale well in terms of network bandwidth, and video server CPU utilization. However, performing most of the processing on the server benefit relatively slow workstations such as the Sun IPC. Using this approach we notice that the Sun IPC in the Table 5.2 achieves a better frame rate than in the Local experiment. In the Remote experiment of Sun 10 sending to itself, the frame rate was much lower than in the Local experiment. The reason behind this decrease was due to the video bandwidth sent. The video maps of X Windows had to go down the communication protocol stack and come back to the Xserver for display.

(iii) **Network:** This experiment achieved a frame playing rate comparable to the frame playing rate of the Local experiment for the Sun IPC and Sun 10 clients. This approach keeps the video stream compressed in MPEG format as long as possible, thus preserving the network bandwidth (unlike the Remote experiment). Additionally, decoding the MPEG stream is performed on the client's machine relieving the video server from this burden, which is computationally expensive. Actually, the only burden on the video server, in this experiment, is in reading the benchmark file from the video repository (hard disk). Notice that in the case of the SGI machine, we achieved a slightly higher frame rate in the Network experiment than in the Local experiment since the video server Sun 10 hard disk access was a little faster. This shows that a sophisticated video server that overcomes the disk I/O bottleneck [91] may improve video playing rates on clients. Also notice that in the Network experiment of Sun 10 to itself the frame rate achieved was higher than the Remote experiment. This is expected since a compressed video stream had to go down the protocol stack and come back to the *Nmpeg-play*. Moreover, since the frame rates in this experiment are comparable to the Local rates, this demonstrates that the campus network was not a limiting factor.

From these measurements note that all machines are quite slow if one remembers that no other demanding processes were running. The DEC Alpha performed similar to the Sun 10. The SPARCstation IPC is practically unusable. The SGI workstation performs bad if its raw speed is compared to the Sun IPC. That is computation speed and the memory bandwidth on client machines were not adequate to support the playing of compressed video. Given that, the SNV mechanism achieved frame playing rates comparable to the Mosaic mechanism rates.

II. Response Time Experiment: Figure 5.12 shows a timing comparison between the Mosaic mechanism and SNV. The response time for SNV is given by:

$$SNV\ Response\ Time\ =\ T_i\ +\ T_p\ +\ T_n\ +\ T_m$$

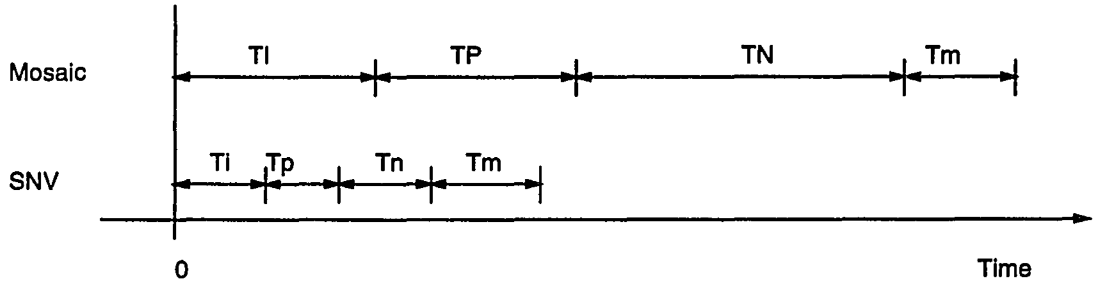
Table 5.4 presents the results of the experiment used to determine the time to start viewing video. The time achieved by the Mosaic mechanism is linearly proportional to the video file size. The time achieved by the SNV mechanism was consistently 2-3 seconds, and has no relationship to the video file size.

Table 5.2: Frame Rate Video Benchmarks and Results in frames/s

Name / Frame Size (bits) / Number of Frames		Bear 1 320x240 201	Bear 2 320x240 134	Bird 160x128 241	Flower1 320x240 148	Flower2 320x240 150	Simpsons 192x144 960	Still 1 320x240 161	Still 2 320x240 173
Client WS	Server WS	Local Experiment: on the same machine							
Sun IPC	Local	2.95	3.89	8.74	2.22	1.27	5.39	2.15	3.80
SGI IRIS 4D	Local	3.91	5.80	12.38	3.79	2.24	9.15	3.64	6.11
Sun 10	Local	9.93	11.93	26.95	8.56	5.19	19.93	7.65	12.27
		Remote Experiment: Sun 10 (gpacs1) is the player							
Sun IPC	Sun 10	4.25	4.78	17.04	2.36	2.53	9.30	3.72	4.30
SGI IRIS 4D	Sun 10	4.20	4.50	16.70	3.00	2.12	9.76	3.33	4.40
Sun 10	Sun 10	4.82	5.22	18.05	4.28	3.22	12.47	4.34	4.91
		Network Experiment: Sun 10 (gpacs1) is the server							
Sun IPC	Sun 10	2.84	3.65	7.70	2.40	1.30	5.32	2.25	3.53
SGI IRIS 4D	Sun 10	5.04	6.53	14.16	4.65	2.87	10.33	4.11	6.30
Sun 10	Sun 10	9.87	12.25	26.95	8.75	5.08	19.15	7.82	12.29

Table 5.3: DEC Alpha Frame Rate Video Experiment in frames/s

	Local Experiment on the same machine (gpacs2)	Remote Experiment (raptor) is the player	Network Experiment (raptor) is the server
201 frames 320x240 pix	9.11	4.30	7.28



Tl: Time for disk I/O for the whole video
Ti: Time for disk I/O for SNV segment

TP: Time for protocol processing for the whole video
Tp: Time for protocol processing for SNV segment

TN: Time for network latency for the whole video
Tn: Time for network latency for SNV segment

Tm: Time for decoding first MPEG frame

Figure 5.12: Comparison of the Time to Start Viewing Video

Table 5.4: Response Time (seconds) to Start Viewing Video

	Video File Size (Mbyte)						
Mechanism	**1**	**10**	**20**	**30**	**40**	**50**	**60**
Mosaic	10.71	106.99	213.76	321.55	428.42	536.63	644.61
SNV	2.23	2.23	2.23	2.23	2.23	2.23	2.23

CHAPTER 6

Conclusion

6.1 Summary

The use of OSF DCE services for Global PACS enables us to develop a robust distributed structure and new user services which feature reliability and scalability for Global PACS environments. This work presented:

a. System components of a healthcare delivery system are standard based.

b. OSF DCE provides the middleware to integration of system components

Benefits are heterogeneity, scalability, security in a complex distributed system

The main contributions of this work is

a. Design considerations of Global PACS in a DCE environment.

b. Description of the development of a Global PACS DCE-based prototype at the University of Arizona.

c. Specific implementation of Global PACS DCE servers using directory service, time service, RPC facility, security service, and threads facility.

6.2 System Constraints

There were some constraints that affected the design of the system:

a. The current vital signs interface is not standard based. An interface based on the Medical Information Bus standard can be used.

b. Better replication and caching of DCE resources can be achieved after monitoring the system in operation. The system adminstrator can observe the operational traffic and corresponding performance measurements and experiment with different system setups, and replication and caching allocations.

c. Currently, many developers provide DCE kits for different hardware platforms. However, the designer had access to the DEC DCE development kit only. This limited the testing to DEC machines.

6.3 Future Work

Eventually, a Global PACS environment for healthcare delivery might include 1000's of workstation nodes and 100's of database archive nodes. This opens the door for a wide area of research and development:

a. Add video interface developed for the RCD system to the Comprehensive Chart interface to allow the medical expert to visually examine the patient. This interface will bring healthcare to the patient avoiding unnecessary travel by the patient or by the medical expert.

b. Add control interfaces for remote procedures. such as the interface to control a remote microscope for a pathology session, or an interface for an endoscopy procedure. This addition would be easy due to CompChart design.

c. Perform clinical trials for CompChart to determine the usability of the interface and the further need for authoring tools.

d. The design of scheduling programs that would prefetch multimedia patient data to the physician's workstation. Such a scheduler will work at times of lower network activities, therefore reducing traffic at peak hours.

e. Development of expert systems that would generate non-linear relationship between events and data elements. This will help the medical expert recognize hidden relationships among the abundant data repository.

f. Integration of prefetching expert systems that may identify important events or data items within the record or reference similar cases in text books or study cases.

g. A monitor of the use of the CompChart system will identify patterns and would further tailor the interface to better suit the user. This monitor may keep a history of operations to identify usage patterns.

h. CompChart could include an administrative interface and not limit the interface to clinical data only.

Appendix A

Setting DCE User Directory Interface

```
/*-------------------------------------------------------------*/
/* Distributed Computing Environment (DCE)                     */
/*                                                             */
/* DCE User Directory interface                                */
/*-------------------------------------------------------------*/
```

After the build is complete, start the server with the following command:

```
% dce_login cell_admin
Enter Password:
% usrdird
```

Once the server is running, you can run the client on the same host, or on any other host in the network that is configured to run in the same cell as the server host. Before running the client, you must define an environment variable on the client system that can be used to locate the server binding information in the directory service during the auto-handle process:

```
% setenv RPC_DEFAULT_ENTRY .:/usrdir_server
```

After you define the environment variable, run the client with the following command:

```
% usrdir
```

The client imports server binding information from the directory service.

If you want to clean up the directory for this program so that you can build it again, enter the following command:

```
% make  clean
```

Appendix B

Setting Stub Auxiliary Files for DCE RPC Pipes

Look at the dxbook Chapter 8. IDL Compiler Enhancements.

8.2 Stub Auxiliary Files

By default, The OSF DCE IDL V1.0.3 compiler does not normally generate the -c a u x and -sau x files that would have been generated when an IDL file was compiled with the V1.0.2 version of the IDL compiler. However, if you want to use build procedures that were designed to work with the V1.0.2 IDL compiler, you can cause the V1.0.3 IDL compiler to generate empty auxiliary files. To do this, define the environment variable IDL_GEN_AUX_FILES with the following command:

```
% setenv IDL_GEN_AUX_FILES "1"
```

REFERENCES

[1] W.J. Dallas et al., "A prototype totally digital radiology department: conception and initiation," in *Proc. SPIE 1987, PACS V, Vol. 767*, February 1987.

[2] R. Martinez, J. Kim, J. Nam, B. Sutaria, "Remote consultation and diagnosis in a global PACS environment," in Proc. of SPIE medical imaging conference, February 1993.

[3] R. martinez, Y. Alsafadi, J. Kim, "OSF Distributed Computing Environment for multimedia telemedicine services in global PACS," in *HICSS-28 Jan 3-6* 1995.

[4] T. Ozeki, M. Suzuki, H. Umino, R. Martinez, C. Archwamety, M. Nemat, "Investigation and analysis of the requirements for a picture archiving and communication system (PACS) at the University of Arizona Health Science Center," in *Proc. SPIE 1987, Medical Imaging, Vol.767*, February 1987.

[5] "Fuji computed radiography AC-1," Refrence Number XB-901E (SK.90.06.SA.MW), Fuji photo film Co. LTD.

[6] M. M. Goodsitt, R. A. Bausman, and G. S. Lodwick, "Digital workload in a large radiology department," in *Proc. SPIE, Medicine XIV/PACS IV*, Newport Beach, CA. Feb. 1986, vol. 626, pp. 710-716.

[7] W. J. Chimiak, "The digital radiology environment," in *IEEE Journal on selected areas in Communications, Vol. 10, No.7*, September 1992.

[8] R.Martinez, W. Dallas, K. Komatsu, H. Shinoda, "Concepts and system requirements for a Global PACS," presented at *SPIE Medical Imaging Conference*, 23-38 February 1992, Newport Beach, CA.

[9] J.Steffens, C. Gayda, H. Oswald and Eckart Fleck, "Realtime Requirements and technical support for image communication in medicine," in *SCAR'92*, Baltimore, MD, June 14-17, 1992.

[10] B.K. Stewart, S.L. Lou, A. Wong, K.K. Chen and H.K. Huang, "Performance characteristics of an ultrfast network for PACS," in *Proc. Medical Imaging V: PACS design and evaluation*, 26 February - 1 March 1991, San Jose, CA.

[11] G. K. Wallace, "The JPEG still picture compression standard," in *IEEE Transactions on consumer electronics*, V. 38, No. 1, Feb. 1992.

[12] "JPEG still image data compression standard," Van Nostrand Reinhold, N.Y., 1993.

[13] "MPEG: A video compression standard," in *Communications of the ACM, Vol 34(4), pp.30-44* 1991.

[14] M. Breeuwer, R. Heusdeus, P. Zwart, "Overlapped transform coding of medical x-ray images," in *SPIE Vol. 2164, pp. 264-275.*

[15] P.M. Ratib, N.J. Mankovich, R.K. Taira, P.S. Cho, and H.K. Huang, "Integration of an MR image network into a clinical PACS", in *Proc. of SPIE Medical Imaging II,* 1988.

[16] E.B. Diethrich, "Voice recognition and voice synthesis in the cardiac intensive care unit," in *Speech Technology,* Sept./Oct. 1988.

[17] M.L. Gordon, "A voice entry radiologic database system," in *Speech Technology,* Sept./Oct. 1988.

[18] J. McGee, "Voice-to-text radiology report systems: today and the future," in *Speech Technology,* Sept./Oct. 1988.

[19] Y.Alsafadi, W.H. Sanders, and R. Martinez, "Evaluation of a picture archiving and communication system with combined voice and data traffic," in *Proc. SPIE Medical Imaging IV*, Newport Beach, February 1990.

[20] Simonic, K.-M.; Gell, G. "A general approach to the strategy of integrating healthcare applications using HL7", MEDINFO 92. Proceedings of the Seventh World Congress on Medical Informatics p.1432-6 vol.2, 6-10 Sept. 1992

[21] Sideli, R.; Johnson, S.; Weschler, M.; Clark, A.; Simpson, R.; Chen, C. "Adopting HL7 as a standard for the exchange of clinical text reports," Fourteenth Annual Symposium on Computer Applications in Medical Care. Standards in Medical Informatics. A Conference of the American Medical Informatics Association p.226-9 Washington, DC, USA 4-7 Nov. 1990

[22] B.P. Cahill, J.R. Holmen, P.L. Bartleson, "Mayo foundation electronic results inquiry, the HL7 connection," in *Sixteenth Annual Symposium on Computer Applications in Medical Care. Standards in Medical Informatics. A Conference of the American Medical Informatics Association p.516-20,* 1992.

[23] M.J. Shafarman, et. al., "Implementing a record-oriented clinical lab interface using HL7 version 2.1 at Indiana University Hospital," in *Sixteenth Annual Symposium on*

Computer Applications in Medical Care. Standards in Medical Informatics. A Conference of the American Medical Informatics Association p.511-15, 1992.

[24] "Health Level Seven: An application protocol for electronic data exchange in healthcare environments," Version 2.1, Available from: Health Level Seven, P.O.Box 66111, Chicago, Illinois 60666-9998

[25] Schultz, S.; Abraham, I. "HL7: an emerging standard for health information systems," MEDINFO 89. Proceedings of the Sixth Conference on Medical Informatics p.1032-5 Editor(s): Barber, B.; Cao, D.; Qin, D.; Wagner, G.

[26] Rishel, W. "Pragmatic considerations in the design of the HL7 protocol (medical computing)" in *The Thirteenth Annual Symposium on Computer Applications in Medical Care (Cat. No.89TH0286-5) p.687-91 Washington, DC,* 5-8 Nov. 1989.

[27] Rishel, W; "Standards seen as a key to managing healthcare costs," in *Computers in Healthcare,* November 1992.

[28] Hammond, W.E. "Health Level 7: An application standard for electronic medical data exchange," in *Topics in Health Record Management, 11(4), pp. 59-66*, 1991.

[29] S.W. Smith, M.E. Ebert, "Health level seven for radiology applications," in *Computer assisted radiology, CAR'91,* 1991.

[30] S. Schultz, "Integrating information systems with HL7," in *Hospitals, FB60-FB62, July 5, 1989.*

[31] W.E. Hammond, "Health level 7 a protocol for the interchange of healthcare data," in *Progress in standardization in health care informatics,* G.J.E. De Moor et al. (Eds.) IOS Press 1993.

[32] Robison, D.R. "Implementing the health care industry's HL7 communication protocol," Proceedings of the Digital Equipment Computer Users Society. Fall, 1990 USA p.217-22

[33] G. Kuperman, R.M. Gardner, T.A. Pryor, "HELP: A dynamic hospital information system," Spinger-Verlag, 1991.

[34] K. Inamura, "Tutorial: basic IMAC concepts, technology overview," in *The First International Conference on Image Management and Communication,* Washington, D.C., June 4-8. 1989.

[35] S. C. Horii, "An introduction to the ACR-NEMA standards," in *2nd international conference on image management and communication,* Kyoto, Japan, April 10-13, 1991.

[36] Y. Wang, et. al., "A summary of compatibility of the ACR-NEMA imaging and communication standard, " in *SPIE Vol. 767, Medical Imaging,* 1987.

[37] O. L. Nelson, C. L. Rennolet, "Implementing the ACR-NEMA protocol on a network," in *SPIE Vol. 626 Medicine XIV/PACS IV,* 1986.

[38] D. E. Best, S. C. Horii, W. Bennett, C. Parisot, "Update of the ACR-NEMA digital imaging and communications in medicine standard," in *SPIE Vol. 1654, Medical Imaging VI* 1992.

[39] R. Martinez, C. Archiwamety and M. Nemat, "Simulation and design of ACR-NEMA standard for fiber optics network image transfer," in *Proc. SPIE 1987, Medical Imaging, Vol.767,* February 1987.

[40] B. M. Hemminger, B. G. Thompson, P. E. Stancil, "Implimentation of the ACR-NEMA standard and building towards an ACR-NEMA test facility at the university of north carolina," in *SPIE Vol. 767, Medical Imaging,* 1987.

[41] K. M. McNeill, et. al. , "Evaluation of the ACR-NEMA standard for communications in digital radiology," in IEEE Transactions on medical imaging, Vol. 9, No. 3, Sept. 1990.

[42] U. M. Maydell, H. S. Hassanein, S. Deng, "On enhancing the performance of the ACR-Nema protocol," in *SPIE Vol. 1446,* 1991.

[43] American college of radiology, national electrical manufacturers association, "ACR-NEMA digital imaging and communications standard: ACR-NEMA standards publication No. 300-1985." Washington, D.C., 1985.

[44] K. McNeill, M. P. Capp, W. J. Dallas, R. Martinez, "Standardization of network addressing in picture archiving and communications systems utilizing the ISO OSI protocols," in *Computer methods and programs in biomedicine, Vol. 37, no. 4, pp. 311-318,* May 1992.

[45] E. Alzner, L. Murphy, "The ACR-NEMA digital imaging and communications standard: evolution, overview and implementation considerations," in *SPIE Vol. 626, Medicine XIV/PACS IV,* 1986.

[46] H. Oosterwijk, F. W. Gutzwiller, A. Giangrande, "Practical and strategic implications of the ACR-NEMA interface standards," in *SPIE Vol. 626, Medicine XIV/PACS IV*, 1986.

[47] S. C. Horii, et. al., "The ACR-NEMA magnetic tape standard," in *SPIE Vol. 626, Medicine XIV/PACS IV*, 1986.

[48] S. C. Horii, et. al., "The ACR-NEMA exchange media standards: progress and future directions," in *SPIE Vol. 767, Medical Imaging,* 1987.

[49] J.L. Lehr, "The ACR/NEMA standard data dictionary: aradiologist's perspective," in *SPIE Vol. 626, Medicine XIV/PACS IV*, 1986.

[50] W.F. Good, J.M. Herron, G.S. Maitz, D. Gur, "ACR-NEMA standard: the reality vs. the ideal," in *SPIE Vol. 914, Medical Imaging II,pp. 1081-86* 1988.

[51] Y. Wang, et.al., "Update of the ACR-NEMA standard committee," in *SPIE Vol. 914, Medical Imaging II, pp.1406-08* 1988.

[52] S.B. Seshadri, S. Khalsa, R.L. Arenson, I. Brikman, M.J. Davey, "An image archive with the ACR/NEMa message formats," in *SPIE Vol. 914, Medical Imaging II, pp.1409-15* 1988.

[53] C.L. Ringleben, W.R. Wessell, P. Wiess, Klaus Niggenaber, "An ACR-NEMA implementation and PACS architectural implication," in *SPIE Vol. 914, Medical Imaging II, pp.1469-80* 1988.

[54] R. Martinez, W. Dallas, K. Komatsu, "Evaluation and critique of the ACR-NEMA standard for picture archiving and communications systems," in *SPIE, Medical Imaging IV,* 1990.

[55] R. De Dominics, Y. Watanabe, "Interface Standardization," in *The Second international conference on image management and communication, pp.255-258,* Kyoto, Japan, April 10-13, 1991.

[56] W.J. Chimiak, R.C. Williams, "Using the ACR/NEMA standard with TCP/IP and Ethernet," in *SPIE Vol. 1446 Medical Imaging V, pp.93-99*, 1991.

[57] J.K. Udupa, H.M. Hung, D. Odhner, R. Goncalves, "Multidimensional data format specification: a generalization of the ACR-NEMA Standards," in *SPIE Vol. 1654, Medical Imaging VI, pp. 266-277,* 1992.

[58] W. D. Bidgood Jr., S.C. Horii, "The ACR-NEMA DICOM standard: information model, commands and network protocols," in *SCAR'92, pp. 566-580* Baltimore, MD, June 14-17, 1992.

[59] I. Obayashi, Y. Takahashi, Y. Tani, M. Suzuki, "Network analysis of picture archiving and communication system," in *ISMII Proc., pp 109-119,* 1984.

[60] U.M. Maydell, P.S. Gill and W.A.Davis, "Design criteria for an interactive image computer system," in *Proc. ELECTRONICOM '85, pp. 282-285,* Toronto, Oct. 1985.

[61] A.D.A. Massar, J.P.J. de Valk, G.L. Reijns, A.R. Bakker, "Simulation of an image network in a medical image information system," in *Proc. SPIE 1985, Medical Imaging Processing, Vol.593.*

[62] R.K. Taira, N.J. Mankovich, H.K. Huang, "Oneyear experience with a PACS module in pediatric radiology:. system viewpoi nt," in *Proc. SPIE 1988, Medical Imaging II, Vol.914*, February 1988.

[63] S.C. Horri, G. Isles, R.T. Bergeron, "Peak data rates for image generation ," in *Proc. SPIE 1987, PACS V, Vol. 767*, February 1987.

[64] S.K. Mun et al., "Total digital department: implementation strategy," in *Proc. SPIE 1989, Medical Imaging*, Newport Beach, CA, February 1989.

[65] J. Shirley, "Guide to writing DCE Applications," O'Reilly & Associates, Inc., 103 Morris Street, Suite A, Sebatopol, CA 95472 1992

[66] Open Software Foundation, "Introduction to OSF DCE," Revision 1.0, Prentice Hall, Englewood Cliffs, New Jersey 07632 1992

[67] W. Rosenberry, D. Kenney, G. Fisher, "Understanding DCE," O'Reilly & Associates, Inc., 103 Morris Street, Suite A, Sebatopol, CA 95472 1992

[68] Open Software Foundation, "OSF DCE: User's guide and reference," Revision 1.0, Prentice Hall, Englewood Cliffs, New Jersey 07632 1993

[69] Open Software Foundation, "OSF DCE: Application development reference," Revision 1.0, Prentice Hall, Englewood Cliffs, New Jersey 07632 1993

[70] Open Software Foundation, 'OSF DCE Administration Reference," Revision 1.0, Prentice Hall, Englewood Cliffs, New Jersey 07632 1993

[71] Open Software Foundation, "OSF/1 User's Guide," Revision 1.0, Prentice Hall, Englewood Cliffs, New Jersey 07632 1993

[72] Y. Alsafadi, K.M. McNeill, R. Martinez, "Application driven strategies for efficient transfer of medical images over very high-speed networks," in Proc. of SPIE medical imaging conference, February 1993.

[73] C.J. McDonald, "The search for national standards for medical data exchange," in *MD Computing, Vol. 1, No. 1*, 1984.

[74] Simonic, K.-M.; Gell, G. "A general approach to the strategy of integrating healthcare applications using HL7", MEDINFO 92. Proceedings of the Seventh World Congress on Medical Informatics p.1432-6 vol.2, 6-10 Sept. 1992

[75] Sideli, R.; Johnson, S.; Weschler, M.; Clark, A.; Simpson, R.; Chen, C. "Adopting HL7 as a standard for the exchange of clinical text reports," Fourteenth Annual Symposium on Computer Applications in Medical Care. Standards in Medical Informatics. A Conference of the American Medical Informatics Association p.226-9 Washington, DC, USA 4-7 Nov. 1990

[76] B.P. Cahill, J.R. Holmen, P.L. Bartleson, "Mayo foundation electronic results inquiry, the HL7 connection," in *Sixteenth Annual Symposium on Computer Applications in Medical Care. Standards in Medical Informatics. A Conference of the American Medical Informatics Association p.516-20*, 1992.

[77] M.J. Shafarman, et. al., "Implementing a record-oriented clinical lab interface using HL7 version 2.1 at Indiana University Hospital," in *Sixteenth Annual Symposium on Computer Applications in Medical Care. Standards in Medical Informatics. A Conference of the American Medical Informatics Association p.511-15*, 1992.

[78] Hammond, W.E. "Health Level 7: An application standard for electronic medical data exchange," in *Topics in Health Record Management, 11(4), pp. 59-66*, 1991.

[79] "Health Level Seven: An application protocol for electronic data exchange in healthcare environments," Version 2.1, Available from: Health Level Seven, P.O.Box 66111, Chicago, Illinois 60666-9998

[80] Robison, D.R. "Implementing the health care industry's HL7 communication protocol," Proceedings of the Digital Equipment Computer Users Society. Fall, 1990 USA p.217-22

[81] Rishel, W. "Pragmatic considerations in the design of the HL7 protocol (medical computing)" in *The Thirteenth Annual Symposium on Computer Applications in Medical Care (Cat. No.89TH0286-5) p.687-91 Washington, DC,* 5-8 Nov. 1989.

[82] E. Nishihara, et. al., "RIS/PACS Architecture to assis daily practical reading at the Toshiba Hospital," in *SPIE Medical Imaging'94, Vol. 2165, pp.172-178*, 15-18 February 1994.

[83] P.J. Toussaint, F.P. Ottes, A.R. Bakker, "Should allocation of functions between HIS/RIS and PACS be standardized," in *SPIE Medical Imaging'94, Vol. 2165, pp.795-801*, 15-18 February 1994.

[84] P.J. Haug, T.A. Pryor, P.R. Frederick, "Integrating radiology and hospital information systems: the advantage of shared data," in *Sixteenth annual symposium on computer applications in medicl care,* November 8-11, 1992.

[85] "Technical requirements for participation 1994 RSNA DICOM Demonstration," Radiological Society of North America, Mallinckrodt Institute of Radiology, Version 1.3, July 11, 1994

[86] R.W.Scheifler, J. Gettys, "The X window System," in *ACM Trans. on Graphics, Vol.5, No.2, pp.79-109,* April 1986.

[87] A. Nye, "Xlib Programming Manual for Version 11," Volume One, O'Reilly and Associates, Inc., 1990.

[88] Patel, et. al., "An MPEG player for the X windows system," 1993.

[89] R.Droms, W.R. Dyksen, "Performance measurements of the x window system communication protocol," in *Software-Practice and Experience, Vol. 20, No.s2, pp.119-136,* October 1990.

[90] N.Basil, C.L. Williamson, "Network traffic measurement of the x window system," in *13th IPCCC , pp.148-155,* April 1994.

[91] M. Buddhikot, G.M. Parulkar, J.R. Cox, "Design of a large scale multimedia storage server," in *Proc. INET 94*, Sept. 1994.

[92] T.S. Yum, M-S. Chen, Y-W. Leung, "Bandwidth allocation for multimedia teleconferences," in *ICC '91.*

[93] S.M. Mok, T.S.P. Yum, "Optimal scheduling of delayable videos in video distribution networks," in *Globecom '92.*

[94] Y.W. Leung, T.S. Yum, "Optimum connection paths for a class of videoconferences," in *ICC '91*.

[95] T.S. Yum, M. Chen, "Dynamic channel assignment in integrated-services cable networks," in *ICC '91*.