Distributed health care imaging information systems

Mary Thompson[a], William Johnston[a], Jin Guojun[a], Jason Lee[a], Brian Tierney[a], and Joseph F. Terdiman[b]

[a]Lawrence Berkeley National Laboratory, Berkeley CA 94720
[b]Kaiser Permanente, Division of Research, Oakland CA 94611

ABSTRACT

We have developed an ATM network-based system to collect and catalogue cardio-angiogram videos from the source at a Kaiser central facility and make them available for viewing by doctors at primary care Kaiser facilities. This an example of the general problem of diagnostic data being generated at tertiary (centralized) facilities, while the images, or other large data objects they produce, need to be used from a variety of other locations such as doctor's offices or local hospitals. We describe the use of a highly distributed computing and storage architecture to provide all aspects of collecting, storing, analyzing, and accessing such large data-objects in a metropolitan area ATM network. Our Large Data-object Management system provides network interface between the object sources, the data management system and the user of the data. As the data is being stored, a cataloguing system automatically creates and stores condensed versions of the data, textual metadata and pointers to the original data. The catalogue system provides a Web-based graphical interface to the data. The user is able the view the low-resolution data with a standard Internet connection and Web browser. If high-resolution is required, a high-speed connection and special application programs can be used to view the high-resolution original data.

Keywords: large data-objects, distributed architectures, digital images, medical imaging, Web-based libraries

1 Introduction

Health care information, especially high-volume image data used for diagnostic purposes - e.g. X-ray CT, MRI, and cardio-angiography - is increasingly collected at tertiary (centralized) facilities, and may now be routinely stored and used at locations other than the point of collection. The importance of distributed storage is that a hospital (or any other instrumentation environment) may not be the best environment in which to maintain a large-scale digital storage system [1], and an affordable, easily accessible, high-bandwidth network can provide location independence for such storage. The importance of remote end-user access is that the health care professionals at the referring facility (frequently remote from the tertiary imaging facility) will have ready access to not only the image analyst's reports, but the original image data itself.

We are evolving a strategy for using high-speed wide area networks and high capacity, high performance network storage systems for the management of data from on-line instruments and imaging systems.[2] The high-level goal is to dramatically increase our ability to organize, search, and provide high performance and location independent access to "large data-objects" (LDO). These objects - typically the result of a single operational cycle of a scientific instrument, run in size from tens of MBytes to tens of GBytes. It is frequently necessary to keeps sets of such objects logically grouped together along with some identifying textual information, for example a set of MRI images, or all the video clips from a single cardio-angiogram procedure on a patient.

In any scenario where data is generated in large volumes, and especially in a distributed environment where the people generating the data are geographically separated from the people cataloguing and using the data, there are several important issues:

- automatic generation of at least minimal metadata;
- cataloguing of the data and the metadata as the data is received (or as close to real time as possible);
- transparent management of cache and tertiary storage systems;
- facilitation of co-operative research by allowing specified users at local and remote sites immediate access to the data;
- incorporation of the data into other databases or documents.

We have several examples of such a system currently in use at Lawrence Berkeley National Laboratory [3]. This paper describes the architecture of an LDO system that collects the daily output from the Kaiser Permanente Cardio-Angiogram facility in San Francisco, stores the raw data and generated metadata on a high-speed memory cache and a tertiary tape storage system in Berkeley, and allows physicians in Oakland and Vallejo to view the captured videos from a local work-station.

The general architecture of the systems we have built consists of several separate components: a data generation component that is normally specific to the source of data; a distributed, high-speed parallel storage system (the DPSS [4]) that can rapidly store and retrieve large data objects; a data-object cataloguing component that automatically generates derived data, organize collections of objects and presents this information via a WWW user interface (the ImgLib Web server [5]); a mass storage system (MSS) where data is automatically migrated between disk and tape depending on its use; a data viewing component that may consist of just standard Web clients, or can be special purpose applications, and; a network linking all the distributed components into a single system.

2 Kaiser Cardio-angiography System

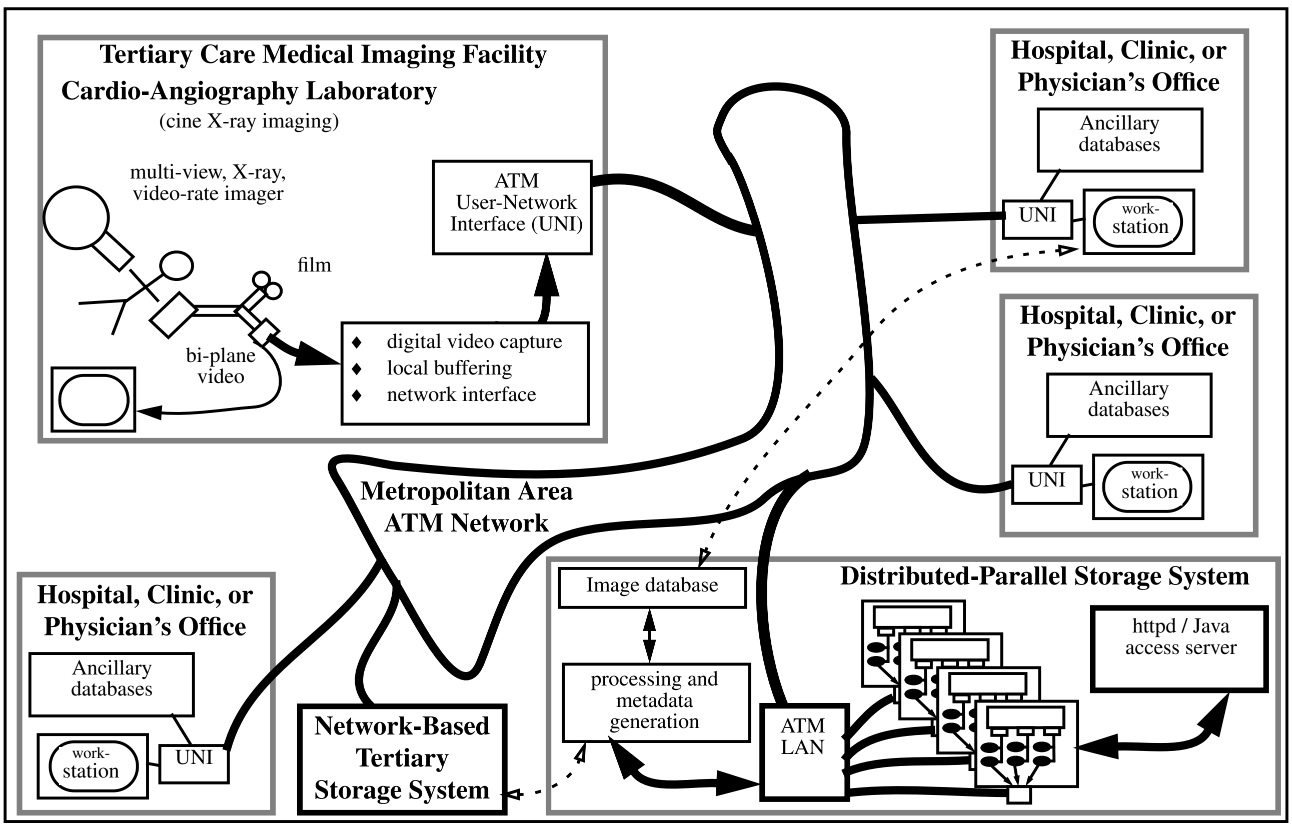

Figure 1 **Distributed Large Data-Object Health Care Imaging Application**

The Imaging and Distributed Computing group at Lawrence Berkeley National Laboratory has been working on a joint project with Kaiser Permanente, Philips Research (Palo Alto), the Pacific Bell CalREN ATM program and the National Transparent Optical Network Consortium [6], that uses a distributed large data-object architecture to provide for collection, storage, cataloguing, and playback of video-angiography images using a metropolitan area ATM network.

> *Cardio-angiography is used to monitor and restore coronary blood flow, and though clinically effective, the required imaging systems and associated facilities are expensive. To minimize the cost of such procedures, health care providers are beginning to concentrate these services in a few high-volume tertiary care centers. Patients are typically referred to these centers by cardiologists operating at clinics or other hospitals; the centers then must communicate the results back to the local cardiologists as soon as possible after the*

procedure. The advantages of providing specialized services at distant tertiary centers are significantly reduced if the medical information obtained during the procedure is not delivered rapidly and accurately to the referring physician at the patient's home facility. The delivery systems currently used to transfer patient information between facilities include interoffice mail, U.S. Mail, fax machine, telephone, and courier. Often these systems are inadequate and potentially could introduce delays in patient care[7].

Using a shared, metropolitan area, ATM network, and a high-speed distributed data handling system, video sequences are collected from the video-angiography imaging systems, stored, and accessed by a remote user. The image data are sent through the network to storage and analysis systems, and from there can be delivered to users at multiple clinic sites. Thus, data can be stored and catalogued for later use, data can be delivered from the imaging device to remote clinics in near real-time, or these data flows can all be done simultaneously. Whether the storage servers are local or distributed around the network is entirely a function of the optimal logistics: There are arguments in regional health care information systems for centralized storage facilities away from the hospital environment, even though the architecture is that of a distributed system.[1]

Figure 1 illustrates the configuration of the Kaiser application. Angiography data is being collected directly from a Philips scanner by a computer system in the San Francisco Kaiser hospital Cardiac Catheterization Laboratory. This system is, in turn, attached to an OC-3, metropolitan area ATM network. When the data collection for a patient is complete the data is sent across the ATM network to LBNL (in Berkeley) and stored on an experimental, high-speed, network-based, distributed-media storage system (the DPSS). Compressed versions of the data are created and then the whole set of data is indexed in a Web database and archived to one or more mass storage systems. This process goes on 8-10 hours a day. Independent of the data gathering process, doctors from several Kaiser facilities are able to find patient's angiograms through Kaiser patient-database Web interfaces that link to LBNL's ImgLib Web server, and then launch a special application to display either the compressed or raw image data.

2.1 Data object generation

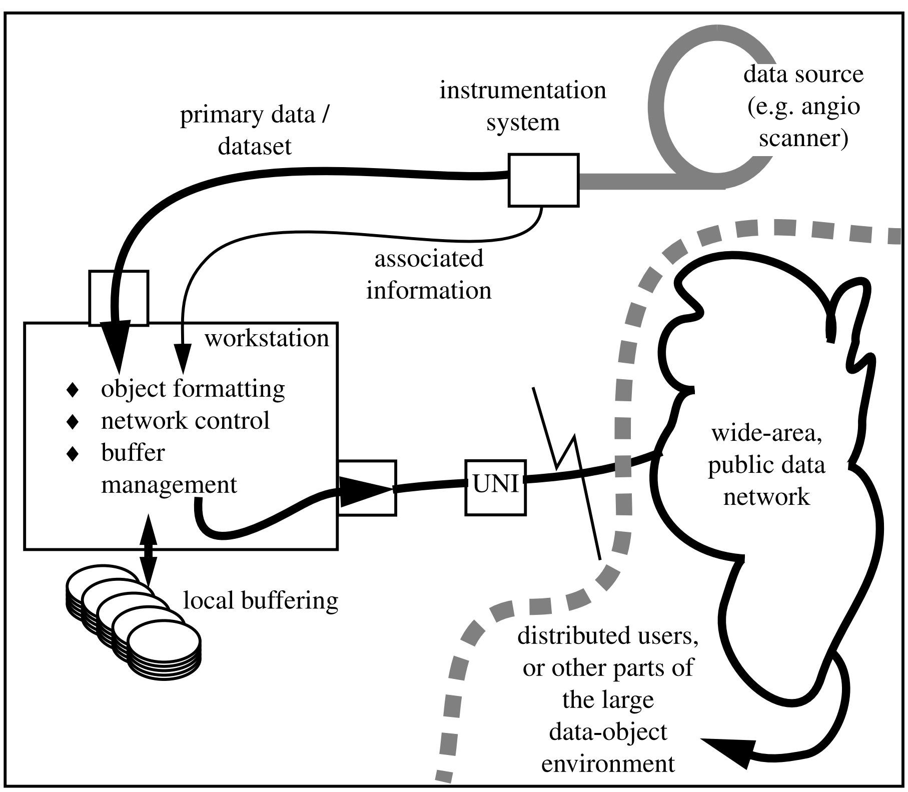

Figure 2 Architecture of the Large Data-Object Generation Environment

Data generation involves capturing the output of the cardio-angiography system. Cardio-angiography imaging involves a two plane, X-ray video imaging system that produces from several to tens of minutes of digital video sequences for each patient session. The digital video is organized as tens of data-objects, each of which are of the order of 50-100 MBytes.

The architecture of the capture system is illustrated in Figure 2. A workstation is interfaced to the Philips scanner to capture digital bi-plane video that is then buffered on a local disk (both to change the data rate from the 125 Mbits/s of the video capture system to the 30 Mbits/s used in the ATM network, and to provide temporary storage of the data in the event of network interruptions). When a patient enters the Cath lab, a technician assigns an identification number which is entered into the capture system. This is followed by a series short (10 seconds or so) X-ray video sequences. These are the “data objects” for this application. This is the same data that is being displayed on video monitors for the presiding doctor and being recorded on film for a permanent record and use by on-site physicians. The capture software sends the patient identification and time the procedure started to the Web data object manager. Whenever the machine is not actively storing data onto its local disk array, it will attempt to send previously saved data to the DPSS at Berkeley. A complete patient session consists of about 7 to 20 such objects, may take about an hour to complete and consists of 500 - 1000 MBytes of digital video data. A small amount of metadata (patient name, study number, date, etc.) is also provided from the angiography system.

This experimental system operates in a non-interfering way with the normal Cath Lab collection and cataloguing of film recordings of the data.

2.2 Data cataloguing

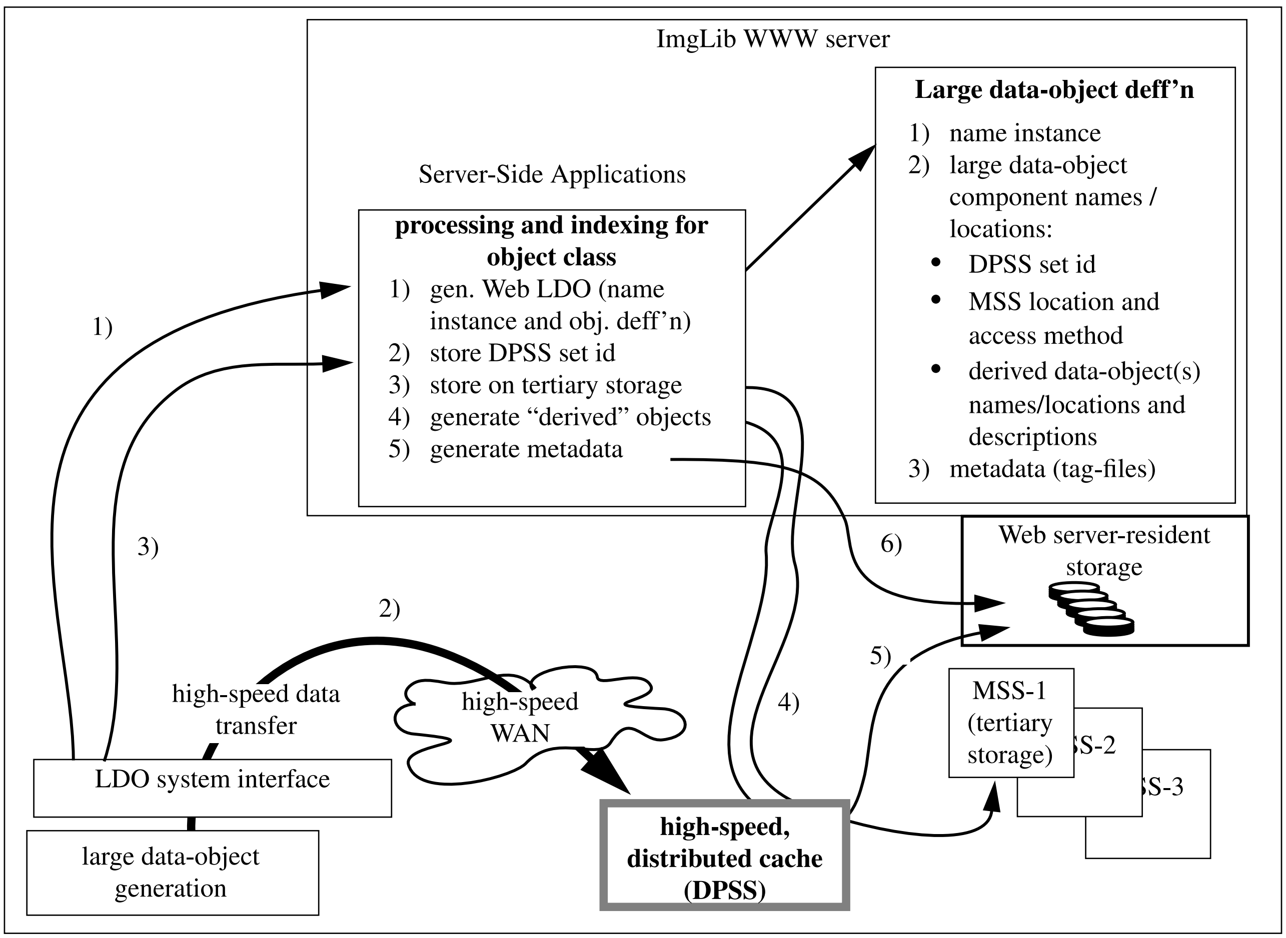

Figure 3 **Large Data-Object Environment: Producer Side**

Figure 3 illustrates the detailed data flow as large data-objects are being created in this system:

1) The angiography system metadata is used to create a name instance and initial object definition in the form of an html document.

2) The data is written to the DPSS which serves several purposes: The DPSS acts as a time “window” for on-line access to the angiograms (that is, the DPSS fills with some quantity of data - say two weeks worth - after which each new data set causes the oldest one to be removed). Another use of the DPSS is as an intermediate cache for any processing that is done to the data. (Each instance of the DPSS has a fixed quantity of disk storage assigned during its initial configuration, and several DPSS systems might run on the same set of distributed disk servers.)

3) Upon completion of the dataset transfer, the DPSS set id (“file name”) is added to the LDO definition.

4) Derived data objects are generated by accessing the original data set on the DPSS and writing the derived data either to the DPSS or to the Web server disks. The derived data is typically something that is useful for creating browsable Web catalogues of the data, or, as is the case for the angiography data, alternate forms of the data that are useful for specific applications. (For the Kaiser angiography system, the original data is converted to motion JPEG[1] sequences and to MPEG[2] sequences. Additionally a “characteristic” frame is chosen and used as the icon for LDO browsing).

5) The data and derived data is archived to tertiary storage. The file location on the tertiary storage system and access method are incorporated into the object definition. (The access method is typically represented as a pointer to a Web server cgi-bin script.)

6) Finally, all of the metadata is put into a standard format “tagged field” html document.

2.3 Data use

The users of these angiograms are the physicians at two East Bay Kaiser facilities, one in Oakland and one in Vallejo. To ensure patient privacy, all on-line patient data is kept on machines that are only accessible via the Kaiser internal network. This patient database contains the same Cath id number that was assigned to the patient when the procedure was done, and that was sent to the ImgLib object server. Kaiser provides an internal Web interface to its patient records and this system can provide a URL to the video data objects on the ImgLib server. Once the doctor has found the data objects for a specific patient procedure, he can view them using any standard MPEG video viewer or download (also via a Web interface) a special application video viewer that will show either the compressed JPEG videos or the uncompressed video frames. So while the patient information comes from a Kaiser Web server (interfacing to internal databases), and the data object information comes from the LBNL ImgLib server and the data comes from the LBNL Distributed-Parallel Storage Server, referencing a angiogram video appears as a seamless Web access to the user.

The DPSS that we are using is sized to hold about two weeks worth of data. Angiograms are usually looked at as soon as they are taken if they are to be used as a basis for a decision on patient care. They may be viewed again within a few weeks if needed as part of follow-up care. But after that they are probably useful only a a part of the patient record or for comparison to a new study if the patient continues to have problems. So after two weeks or so the data will be removed from the DPSS to make room for new sets. The data does remain on the mass storage system and the ImgLib data base keeps track of its location. Thus when a user tries to view an angiogram via the ImgLib interface, the ImgLib can immediately return a pointer to the data on the DPSS if the data is there, or start the reload procedure if the data only exists on the MSS. It may take 10 minutes to an hour or so to reload the data, depending on the speed of the MSS. The user will be informed by the ImgLib that a data reload is in progress and to try to view the data at a later time.

2.4 System evaluation

The system has been collecting data since June 1996 and has been used by the clinicians since about September 1996. The user interface had to be considerably streamlined to get the non-research doctors to use it. Typically the practicing doctors have short periods of time free and are not normally sitting at a computer screen. They want to be able to walk up to a workstation that is already logged on, authenticated and more or else looking at the right place. They are willing to type in a patient’s name navigate a few links before viewing data, but that is about all. Also most primary care physicians will not use such a system very often, so needing to know some specialized interface is unreasonable. The standard Web interface, with clearly labeled links to click on, seems to meet the need and is probably an interface with which most doctors are familiar.

It is also important that the system be reliably available. Our current system, being built from several experimental and rapidly evolving components, including the network, is not as consistently available as it should be. The data collection path is quite good, due to the local data buffering at the Cath lab site and a very robust retry procedure on the data transmission. It survives

---

1. JPEG is a standard single frame image compression that may also be used to encode video (“motion JPEG”). This coding is of higher quality at the single frame level than is MPEG coding, but as a video representation it is not standardized.

2. MPEG is a standard compression encoding of video that is frequently used to deliver short video programs via the Web.

network outages and server down times and loads the stored data when everything is back up. If the network link between San Francisco and LBNL is down for more that 24 hours or the DPSS server is down for a similar time, the data for that time is not captured. If the workstation (a Sun Sparc 20) in the Cath lab fails, that also causes us to fail to capture data. Failures of this sort have been relatively rare. The more common cases of short network outages or DPSS failures are handled without data loss.

The more frequent case of data loss results from the fact that the data on the DPSS has not always been backed up immediately to the MSS. In theory and design this should be done automatically as soon as the data arrives on the DPSS, but because the Lab's Mass Storage system is seriously overused, and this is an experimental project where the data is not vital to the Lab's mission, the backup has been done on a periodic basis. Thus disk failures on the DPSS or errors in the server code have occasionally caused major data loss. The current DPSS design assumes that it is being used as a high-speed data-cache and thus does not recover well from disk errors. The design is currently being expanded to provide some data redundancy, but for the moment any permanent data stored there should also be saved in some more archival data system.

3 Distributed Large Data-Object Management

Our experience with the Kaiser angiogram system, and a couple of ImgLib applications (see [5]) that consist of larger collections of smaller objects (e.g. 2-30Mbyte microscope and photographic images) led to a generalized concept of Large Data-Object Management. The basic goal of such a system is to provide a convenient and quick way to manipulate large collections of these unwieldy objects. The functions provided include storing, cataloguing, searching and analyzing the objects.

Our model of a large data-object includes flexibility of definition, location independent access, high performance access, and persistence in naming, location of reference, and in time. These characteristics are provided through an architecture that combines the addressing, searching, association, and "method" based access features of the Web, (the ImgLib system) together with a high speed network cache system for receiving high-speed data streams, and making large data sets readily available (the DPSS) and a archival storage system (MSS) to ensure data persistence

The general structure of one of these "objects" includes persistent pointers (URLs), a generalized object definition, and search mechanisms. The object definition is manifest as a Web document that contains pointers to metadata, data component locations, access methods, derived information and representations, and search methods.

3.1 Basic elements

The Kaiser system demonstrates all the basic elements of our distributed, large data-object architecture, except for the general security architecture[8], which is currently being implemented:

- data collection and the instrument-network interface
- on-line cache storage that is distributed throughout the network and transparently managed by the system
- near-line tertiary storage for high volume archives, transparently managed by the system
- processing elements - also distributed throughout the network - for various sorts of data analysis and derived data generation
- data management that provides for the automatic cataloguing and metadata generation that produces the large data-object
- data access interfaces, including application-oriented interfaces
- user access to all relevant aspects of the data (application, data, metadata, data management)
- flexible mechanisms for providing various searching strategies
- transparent security that provides access control for the data and all of the system components based on the resource / data - owner's policies

These elements all need to be provided with flexible, location-independent interfaces so that they can be freely (transparently) moved around the network as required for operational or other logistical convenience.

The two areas under most development currently are the security aspects and increasing the storage redundancy of the DPSS. In our current systems either security is not important, i.e.,we want the information to be publicly available, or it can be handled by the current Web domain and password functionality, or, as in the Kaiser case, all critical data is kept off of the public net.

3.2 Entering an object into a large-data object library

The basic steps are:

- The data-objects are entered in the high-speed cache storage and subsequently archived in a tertiary storage system.
- Metadata is generated by analyzing the object, encoding information forwarded by the object generator, or by associating other information with the object. Typically there is a "tagged-field" text file for description, small "thumbnail" images for image data, and compressed versions of video data.
- All of the information related to the data-object is combined into (or otherwise made available as) a Web document-based description that defines the large data-object: this includes an index of all related data and metadata for the data-object.

The first step must be done by the object generation process which tends to be unique to the type of object. We provide library code to talk to the DPSS server that an application program can use. It is also planned to add a DICOM interface to the DPSS so that it can accommodate medical instrumentation systems more easily.

Metadata generation also tends to depend somewhat on the type of object that is being entered. There are some items that are done for all objects, like the "tagged" text files and the thumbnail images that the ImgLib browsing procedure use. In the Kaiser case we added the JPEG and MPEG video objects and, while we generate the tagfiles and thumbnails, they are not used much since the location of objects is generally done from the patient database.

Once the data object has been entered, the original dataset has a comprehensive "description" (object definition) with a persistent URL assigned to it, a permanent instance in tertiary storage, and (perhaps) a temporary instance in the network cache.

3.3 User interface to the data-objects

The generic user interface consists of:

- A persistent URL for accessing the data-object.
- A Web interface for searching, browsing, and accessing the metadata, the derived data components, and the primary data component.
- A web interface for adding to or modifying the metadata.
- A Web interface for managing the migration of the data-object into and out of the network cache (or any other cache), as needed by the applications.
- Specific applications that access the primary data component can be launched directly from the Web interface, or can use the Web interface to migrate these components to cache, and access them there.

In the Kaiser application only some of these are used. The persistent URL is used by the Kaiser Web server to provide links to the data object, the migration from tertiary storage may be used, and a specific video viewing application is launched by the client Web browser to view the data. Several other examples of this interface may be seen at [5].

4 The Resulting Environment

Figure 4, Figure 5, and Figure 6 show the Large Data Object environment that results from the model just described.

Figure 4 illustrates the top-level of the angiography collection. An organizational hierarchy based on "year" or "hospital-of-patient-origin", or both, can be added above these individual study pointers as part of the collection curation

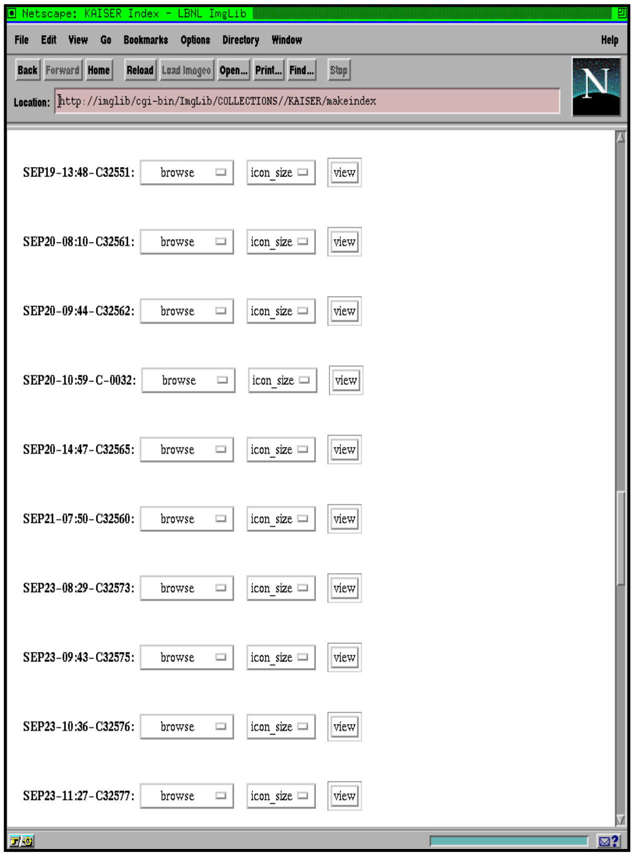

Figure 4 **Index Built in Real-Time From Incoming Video Data**

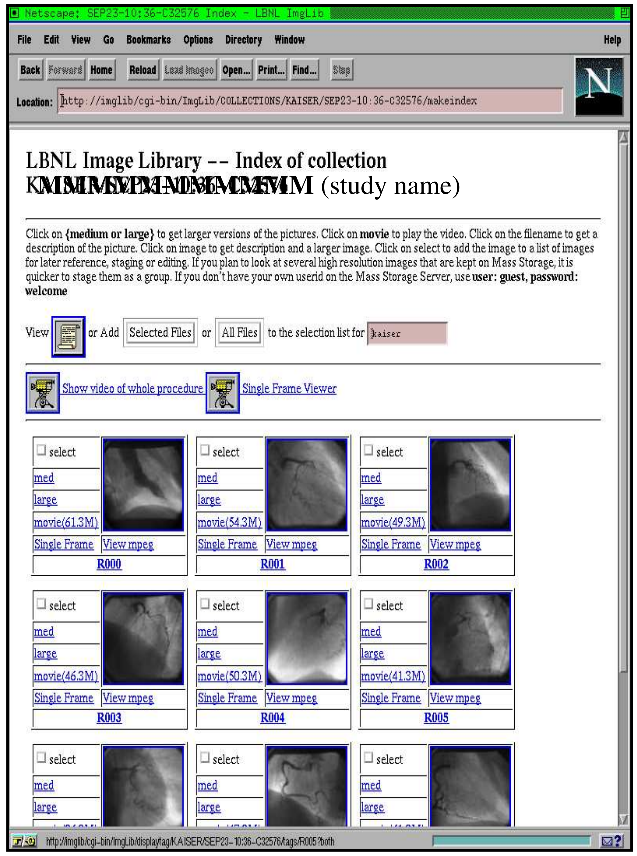

Figure 5 **Browsing the Sub-Collection that Represents One Patient Study (Multiple Video Clips)**

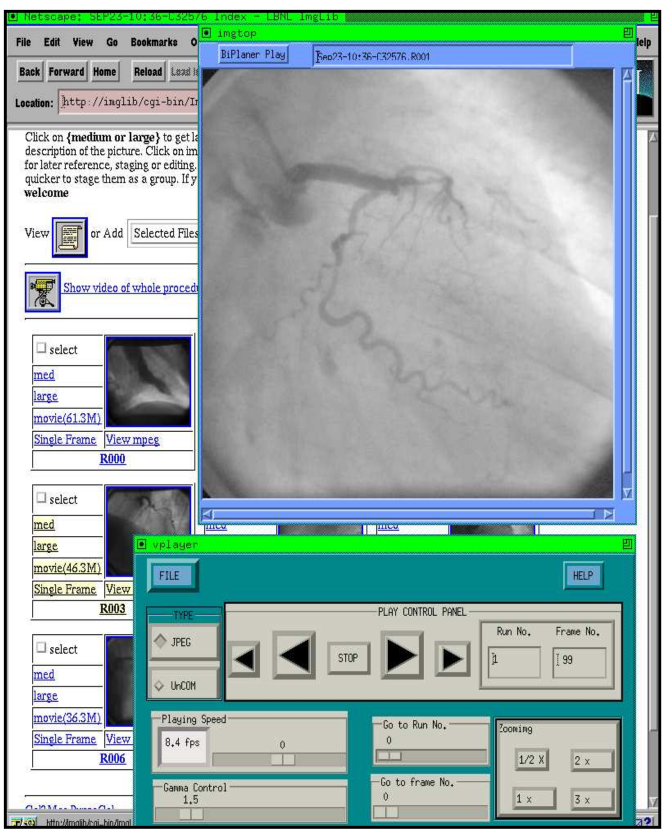

Figure 6 **The User Interface to the Special Application that Displays the Video-Objects from the DPSS**

process. If this is done, then several naming persistence mechanisms can be provided so that links created while the collection has its current organization will remain valid. (See [9] for details.)

Figure 5 illustrates browsing a collection of LDOs (in this case, related by the fact that they are all for a single patient study). At the top of the page is a brief description of the collection (the study identifiers in this case), some standard instructions, and the tertiary storage migration interface. This is followed by the two special applications that understand the bi-plane video format. These are invoked here rather than as a browser "plug-in" because the normal mode of operation is to view all of the segments of a study in sequence. When these viewers are invoked that are also given all of the pointers to the original dataset components of the LDO.

Figure 6 illustrates the result of invoking the bi-plane video player that reads data from the DPSS and displays it as video.

5 Conclusions

A system such as ours to collect, store and provide distributed access to large medical images can be divided into four semi-independent pieces: digital data capture, high-speed network-accessible storage, archival storage and a client graphical user interface.

In our case the digital data capture from the Philips scanner required extensive coding by a member of the Philips Research Lab with support from the hardware developers. This was because the instrument that Kaiser owns is a older one that was not designed to do digital capture. If a newer digital scanner is used, most of the problem would be solved. The only work left would be uploading the data from the capture site to a network-based storage system. If the instrument's software proved a DICOM server interface, that should be a simple problem.

When we started the Kaiser system, we had a working version of distributed storage system. The DPSS was developed first as a general purpose high-speed, distributed, network-based store cache. This is a complicated problem to solve well, and the system is still under intensive development to implement a key-based security system, to increase the level of distribution and number of servers and to add data redundancy. A distributed object management system which did not require high-speed access to the objects, e.g. uncompressed video display, might be able to do without such a system, and substitute standard Web file access instead.

A Mass Storage System that provides mostly transparent migration of data between disk and an archival media such as tape or CD storage also requires a sophisticated piece of software, but one which is available as variety of commercial products. We simply used one that was available to us, in spite of its sometimes inadequate performance. Most medical image systems must have some large scale archival storage. In the past this function may have been provided by film storage archives. But with the advent of digital images, digital archival storage seems necessary. If this system is integrated with the object management system, automatic searching and retrieval from the archive can be easily provided.

The first user interface we provided was a special application video player. The most inherently difficult part of a client interface is to provide a graphic interface that can run on different client workstations. At the time we coded this program, the only workstations that Kaiser had that were fast enough to display the uncompressed video were Sun SPARCstations. Thus we built our program as an X-window application and utilized some specific Sun display code for increased speed. We have since worked on more standard, e.g. cross-platform, ways of displaying the data, such as an MPEG viewer and a couple of Java applets. As we have discovered with our other Image library applications, it is a tremendous advantage to use standard Web browser technology whenever possible for client access to the objects. This interface has the advantages of running on all the major workstation/PC platforms, supporting graphics, and being familiar to many people. We are hopeful that as Java continues to develop, it can meet the need for application specific client interfaces.

The glue that holds these four components together, the Data Object Manager, was very quick to develop as it was leveraged off the Web client-server protocols and applications. It consists of a set of short special-purpose scripts that are run by a standard Web server. The client side is provided by whatever Web browser is available. The Web protocols and applications have proved to be a very powerful way of integrating existing special purpose servers and applications into a coherent, easily accessible and extensible object management system.

6 Acknowledgments

The work described in this paper is supported by the U. S. Dept. of Energy, Office of Energy Research, Office of Computational and Technology Research, ERLTT Program and Mathematical, Information, and Computational Sciences Division (http://www.er.doe.gov/production/octr/mics), under contract DE-AC03-76SF00098 with the University of California. Contacts are: wejohnston@lbl.gov, MRThompson@lbl.gov, Lawrence Berkeley National Laboratory, mail stop: B50B-2239, Berkeley, CA, 94720, ph: 510-486-5014, fax: 510-486-6363, http://www-itg.lbl.gov), and Joe Terdiman, jft@dor.kaiser.org. This is report no. LBNL-39904.

7 References

[1] W. Johnston and A. Allen, "Regional Health Care Information Systems: Motivation, Architecture, and Implementation," LBL Technical Report 34770 (draft), Dec. 1993

[2] Johnston, W., Jin Guojun, Gary Hoo, Case Larsen, Jason Lee, Brian Tierney, Mary Thompson, "Distributed Environments for Large Data-Objects: The Use of Public ATM Networks for Health Care Imaging Information Systems", (http://www-itg.lbl.gov/~johnston/APII.1.1.fm.html)

[3] Johnston,W.,Jin Guojun, Jason Lee, Mary Thompson, Brian Tierney, "Distributed Large Data-Object Management Architecture", Proceedings of Storage and Retrieval for Image and Video Databases V, EI97 San Jose, Feb. 1997, published by SPIE

[4] Imaging and Distributed Computing Group, Lawrence Berkeley National Laboratory "The Distributed-Parallel Storage System (DPSS) Home Page". See http://www-itg.lbl.gov/DPSS.

[5] Imaging and Distributed Computing Group, Lawrence Berkeley National Laboratory, "ImgLib: The LBNL Image Library System". See http://imglib.lbl.gov.

[6] "National Transparent Optical Network Consortium". See http://www.ntonc.org. (NTONC is a program of collaborative research, deployment and demonstration of an all-optical open testable communications network.)

[7] Kaiser - LBNL - Philips CalREN project. See http://www-itg.lbl.gov/Kaiser/LKP/homepage.html.

[8] William Johnston and Case Larsen, "A Use-Condition Centered Approach to Authenticated Global Capabilities: Security Architectures for Large-Scale Distributed Collaboratory Environments". Available at http://www-itg.lbl.gov/~johnston/Security.Arch.Global.Cap.html

[9] William Johnston, Jin Guojun, Case Larsen, Jason Lee, Gary Hoo, Mary Thompson, and Brian Tierney, "Real-Time Generation and Cataloguing of Large Data-Objects in Widely Distributed Environments" (DRAFT -- Submitted to "Research and Technology Advances in Digital Libraries", May, 1997.) Available at http://www-itg.lbl.gov/~johnston/LargeDataObj-Arch/LargeDataObj-Arch.fm.html