National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Canada

MULTIMEDIA WORKSTATION FOR CARDIAC IMAGE SEQUENCES

by
Dominic Richens

A THESIS
submitted to the School of Graduate Studies and Research
in partial fulfillment of the requirements
for the degree of
M.A.Sc.
in
Electrical Engineering.

Ottawa-Carleton Institute for Electrical Engineering
Department of Electrical Engineering
University of Ottawa
Ottawa, Ontario K1N 6N5

April, 1993

National Library of Canada

Acquisitions and Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale du Canada

Direction des acquisitions et des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

ISBN 0-315-89679-5

I hereby declare that I am the sole author of this thesis.

I authorize the University of Ottawa to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Dominic Richens

I further authorize the University of Ottawa to reproduce this thesis by photocopying or other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Dominic Richens

To my parents, all three of them.

Abstract

To diagnose patients with heart disease, cardiologists need to view the co-ordinated movements of various components of the heart, and check for narrowings and blockages in the coronary arteries. Echocardiography and Cardiac Angiography are two imaging modalities which permit visualization of these aspects of the heart. Currently, examinations performed with these modalities are stored on either 35mm ciné film or VHS video cassette, making their review tedious and time consuming for the cardiologists.

We propose a multimedia workstation to store the examinations and all related textual and graphical information in such a way as to permit convenient access. This information would be supported by a multimedia database which would allow for retrieval based on logical indices, such as patient name and examation date.

This thesis details the design, construction and evaluation of such a workstation. To ensure that the workstation is easy to use and that an appropriate set of functions are available to the cardiologist, we employed a *user centred design* methodology. Our workstation digitized the examinations from S-VHS video tape for storage onto disk. The examinations are displayed at 320 by 240 at 20 frames per second, by selecting the patient and the examination date through a graphical user interface.

Usability testing helped to fix both design and implementation errors, and revealed that our user interface was successfully designed. A one week trial in the Coronary Care Unit at the University of Ottawa Heart Institute proved that the workstation is useful for consultations and teaching between the cardiologists and residents on duty.

Acknowledgment

I wish to express my gratitude and appreciation to my thesis supervisors, Dr. Morris Goldberg and Dr. Sethuraman Panchanathan, who generously provided me with an opportunity to do further expand my knowledge in the area of image communications, and system design.

I also wish to thank the numerous cardiologists at the institute for their participation in the usability testing and for their many comments and constructive criticisms. I would particularly like to thank Dr. Brian Morton for this commitment to the project, and Mike Bleakley for his help during the clinical trial. This project would not have been possible without the technical assistance of Ross Jacobs and Joseph Briefman.

I am grateful to the University Research Incentive Fund (URIF) for financial support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In treating patients, cardiologists need to view the co-ordinated movements of various components of the heart. As much heart disease is caused by insufficient blood supply to the heart muscle, cardiologists also need to check for narrowings and blockages in the coronary arteries.

Echocardiography and Cardiac Angiography are two imaging modalities which permit visualization of these aspects of the heart. Currently, examinations performed with these modalities are stored on either 35mm ciné film or VHS video cassette. As a result, locating and viewing a specific examination is tedious and time consuming for the cardiologists, and takes them away from their patients. Also, it is impossible for them to make comparisons between different segments of an examination, when they are on the same film or video tape.

An important task for computer systems in the medical world has always been one of information management. Though the information has been textual, advances during the last decade have made the management of images and voice possible[16]. Development of workstations for the display and manipulation of medical images have focused on workstations for analyzing the image data[42] or as access points on a PACS network[13]. For the two modalities mentioned above, we need a workstation that is able to store and display video data.

To facilitate the cardiologists access to the echocardiographic and angiographic examinations, we propose a computer based workstation to store these examinations for random access by some convenient means. This includes indexing by such data items as patient name and exam date. This workstation, located in the patient ward, would allow the cardiologists to do comparisons between studies, even of different modalities.

In designing this workstation, our aim is not to just investigate the concept of a computer-based viewing of cardiac image sequences, nor just to provide a usable system, but to provide the cardiologist with a system they will *want* to use. To improve our chances of success, we have followed a user centred design methodology[3], a methodology that first designs the system from the user's perspective (the user interface essentially) and then worries about how to make such a design technologically feasible. This methodology tends to be highly iterative, as often some ideas turn out to be too difficult to implement, and hence have to be rethought. When the working prototype is tested, certain assumptions made about the users may turn out to be false, and hence the design must be reformulated. To make such revisions easier, we use the User Interface Development Systems (UIDS) facilities of Khoros (University of New Mexico)[41] so that we could quickly make changes to the user interface.

The result of this thesis is a workstation which is able to store all of the cardiac image sequence, for all patients in the Coronary Care Unit (CCU) at the University of Ottawa Heart Institute. Through a graphical user interface, cardiologists can, for instance, view a patient's examination by selecting the patient name and the examination date. The image sequences are displayed at a resolution of 320 by 240 at 20 frame per second, on a high quality monochrome CRT display. The workstation was built on an Intel 486 based EISA machine, running the UNIX operating system, and the X Window System.

Through usability testing, the workstation software was revised many times, to produce a system that was rated by the cardiologists as being between "easy" and "very easy" to use. To test the usefulness of the workstation in a real environment, a one week test trial was conducted in the CCU. Having the cardiac angiography and the echocardiography examinations available in the ward, the cardiologists and residents were able to consult them during the morning rounds. Also, the residents used the examinations to illustrate their teachings, during times of the day when they would normally not have access to them.

This thesis is organized as follows. In chapter 2, we present a summary of cardiac image sequences and their use as clinical tools. Knowing that our workstation must handle image sequences, chapter 3 examines computer systems designed to handle video information. To ensure that our workstation is easy to use, we familiarize ourselves with user interface design issues. Hence, a review of user interfaces and their design is presented in chapter 4. Next, the actual design process, leading up to the preliminary prototype, is presented in chapter 5. Chapter 6 describes our evaluation of the workstation, through usability testing and through a clinical trial in the CCU at the University of Ottawa Heart Institute.

Chapter 2

Cardiac image sequences

Crucial to the development of any system is a thorough understanding of both the problem and its context. Hence, as a preliminary design step for the cardiologist's workstation, we present a brief summary of cardiac image sequences and their use as clinical tools. We begin with a review of the human heart, its anatomy and function. Next the basic principles of cardiac angiography and cardiac ultrasound are presented. Lastly, their use as clinical tools is discussed, and from this we determine the cardiologist's requirements.

2.1 The human heart – a review

The heart is the organ responsible for pumping blood throughout the body, keeping the various tissues alive with nutrients and oxygen. An average healthy heart in a body at rest goes through its pumping cycle (i.e. 'beats') roughly once per second. As the tissues of the body, including the heart muscles, need a constant supply of oxygen and nutrients, the heart can never stop pumping. This makes heart diseases particularly problematic, as they must be examined either by indirect symptoms, or by observing the heart while it is still working. Cardiac angiography and Echocardiography are two techniques that 'look inside' the body, at the heart. Before we discuss these techniques, however, we give a brief overview of the anatomy of the heart, and its function, for later reference.

2.1.1 Anatomy

The human heart, shown in figure 2.1, is the muscular, roughly cone-shaped, organ lying behind the *sternum*, between the two lungs. About the size of a closed fist, the heart fits into

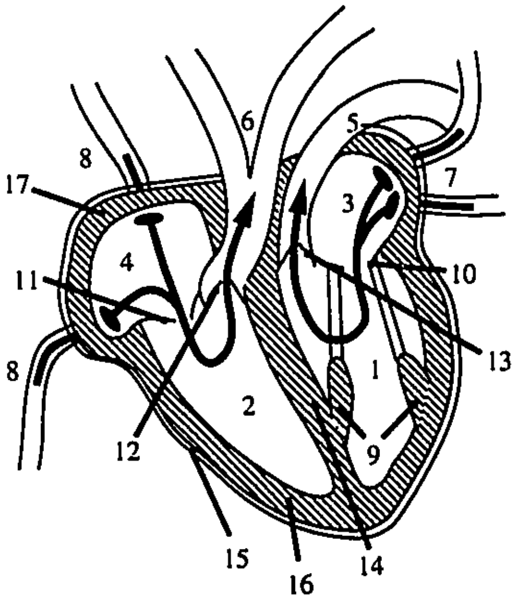


Figure 2.1: Major parts of the heart: 1 Left Ventricle (LV), 2 Right Ventricle (RV), 3 Left Atrium (LA), 4 Right Atrium (RA), 5 Aorta, 6 Pulmonary Arteries, 7 Pulmonary Veins, 8 Vena Cava (superior and inferior), 9 papillary muscles, 10 mitral valve, which is connected to the papillary muscles via chordae tendinae, 11 tricuspid valve, also controlled by papillary muscles (not shown), 12 pulmonary valve, 13 aortic valve, 14 interventricular septum (IVS), 15 pericardium, 16 myocardium, and 17 the sinoatrial node[49].

the chest cavity, and is held in place and surrounded by the *pericardium* membranes. Under the inner pericardium is the *epicardium* membrane, which surrounds the heart wall – the *myocardium* – which consists of cardiac muscle. The heart is divided into left and right sides by a *septum.* Each side has an upper *atrium* and a lower *ventricle.* Through coordinated nerve impulses and muscular contractions, initiated in the *sinoatrial node*, the heart pumps blood throughout the body.

Deoxygenated blood arrives at the heart via the *vena cava.* The superior and inferior vena cava deliver blood from parts of the body above and below the diaphram, respectively. This blood flows into the right atrium and into the right ventricle, which contracts to pump the blood through the *pulmonary arteries* up into the lungs. Once re-oxygenated, the blood enters the left atrium via the *pulmonary veins.* It then passes down to the left ventricle, which then pumps it back into the body, via the *aorta*[49].

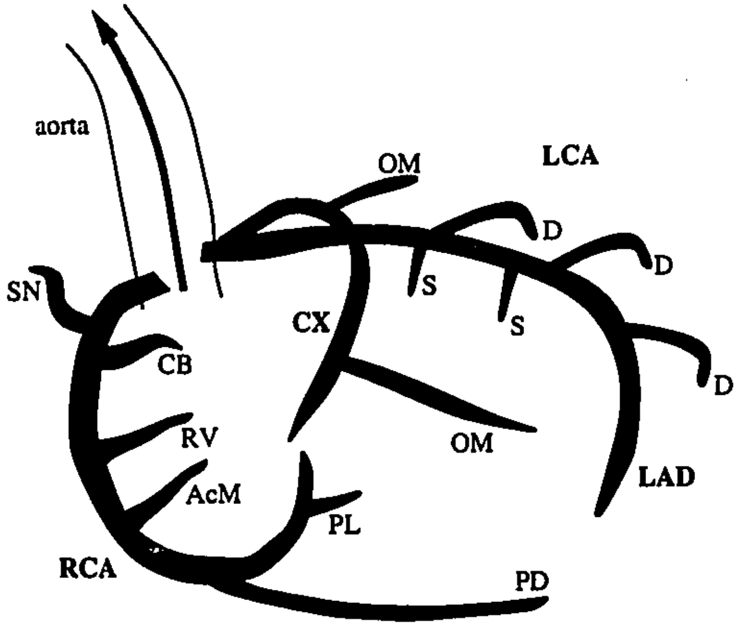


Figure 2.2: A left angular oblique view of the coronary anatomy. The Right Coronary Artery (RCA) delivers oxygen rich blood from the aorta to the right side of the heart. The Left Coronary Artery, which does the same for the left side of heart, is made up of the Circumflex (CX) artery and the Left Anterior Descending (LAD) artery. Each of these arteries has branches which feed the capillaries in the heart muscle[15].

The cardiac muscle, which drives the heart, requires a constant supply of oxygen and nutrient rich blood. There is an extensive network of arteries which deliver blood from the base of the aorta to the numerous capillaries in the heart muscle. Figure 2.2 shows a map of the major arteries and their branches, as they appear in most people.

2.1.2 Function

The heart cycle, while very complex, is usually divided, from a clinical point of view into *systole* and *diastole.*

Systole is the period of the heart cycle where the walls of the ventricles contract to force the blood into the arteries. It begins with the closing of the mitral and tricuspid valves and ends with the closing of the aortic and pulmonary valves.

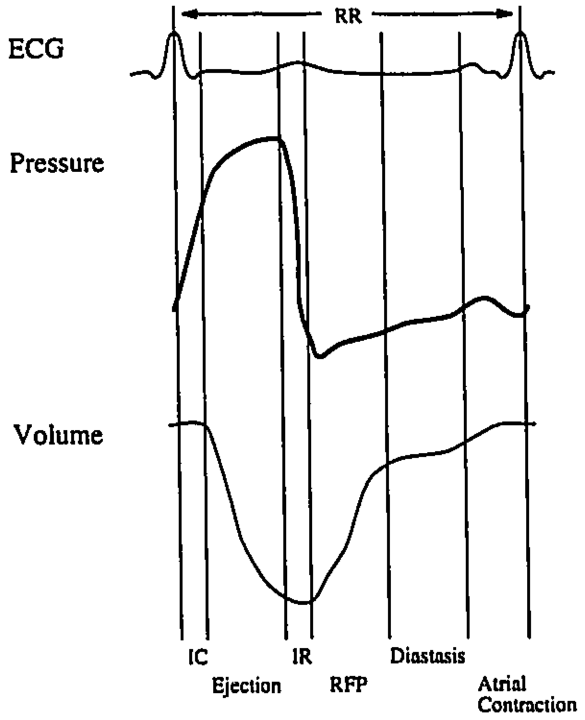


Figure 2.3: The Cardiac Cycle of the left ventricle divided into six stages. IC = isovolumic contraction; IR = isovolumic relaxation; RFP = rapid filling period. From a clinical point of view, systole includes the first two stages and diastole the last four[9].

Diastole is the part of the heart cycle where the ventricles are filled from the atria. It begins with the closing of the aortic and pulmonary valves and ends with the closing of the mitral and tricuspid valves.

Systole and Diastole in a normal health human heart is further divided into six stages[9] as detailed below and shown in figure 2.3:

1. Isovolumic Contraction (IC): After the closing of the mitral valve, there is a period where the volume of the left ventricle remains constant, and there is a sharp pressure increase due to contraction of the heart muscle, prior to the opening of the aortic valve.
2. Ejection: Once the aortic valve opens, the left ventricle continues to contract, causing pressure in the ventricle to remain relatively constant even as the blood is being ejected. This stage ends with the closing of the aortic valve.

3. Isovolumic Relaxation (IR): After the closing of the aortic valve, there is a period where the left ventricular volume remains constant, prior to the opening of the mitral valve. Pressure in the left ventricle drops sharply during this period, due to *active relaxation* of the left ventricle, causing a large pressure gradient across the mitral valve.

4. Rapid ventricular Filling Period (RFP): Once the mitral valve opens, blood rushes into the ventricle, pushed by the difference in pressure between the atrium and the ventricle, which is kept relatively constant by the continued suction of the left ventricle. Evidence to this is the continuing pressure drop in the left ventricle, even after 40 to 60 percent of the blood has entered the ventricle.

5. Diastasis: This stage, known as the *passive filling* phase, starts sometime after the time of peak filling, as the pressure gradient across the mitral valve drops to near zero, due to the end of *active relaxation* of the ventricle. Although blood is not being actively pumped through the mitral valve during this stage, about half of the ventricular volume enters the left ventricle during this time.

6. Atrial Contraction (AC): Just prior to Isovolumic Contraction, the atrial contraction imparts a final push, causing a short yet sharp increase in blood flow just prior to the mitral valve closing.

When the heart becomes afflicted with some illness, one or more of the above stages will change in duration, with respect to the other stages. By identifying which stages have changed and to what degree, a cardiologist can narrow down the number of possible illnesses.

2.2 Principles of ciné angiography

To examine a live human heart, a cardiologist needs to be able to visualize three aspects of the heart; namely the overall shape and position of the heart, the shape and position of its interior anatomy, and its movement throughout the cardiac cycle. Traditional chest radiographs (particularly the posterior-anterior view) are effective in visualizing the first aspect, where the heart casts a shadow that gives some indication of its size, and shifting of the mediastinum at the base is often caused by displacement of the heart. However, on their own, radiographs cannot illustrate the other two aspects. To view the interior anatomy of the heart an angiogram is used. An angiograms is essentially a radiograph taken as a

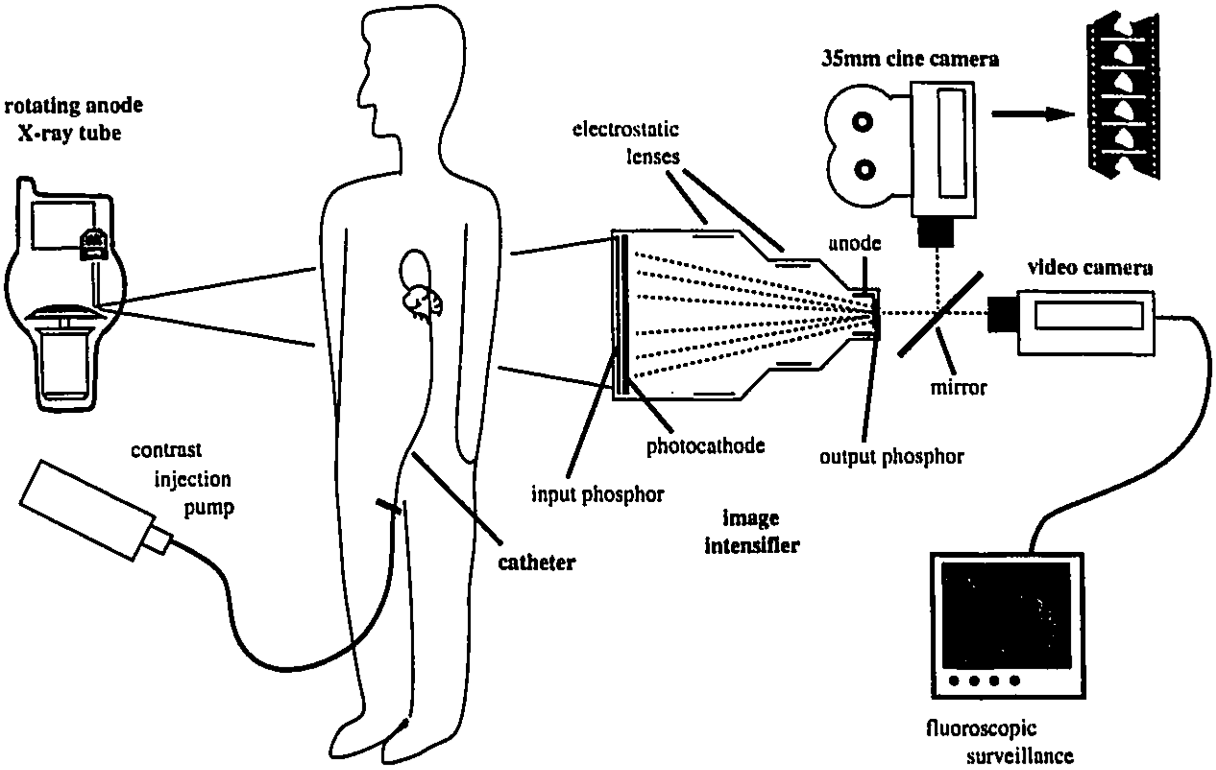


Figure 2.4: A typical cardiac angiography set up, comprising of an X-ray source, an image intensifier, a 35mm ciné camera to record the examination and a video camera for real time fluoroscopic guidance of the catheter. [15].

radio-opaque dye is injected into the vessels of the organ of interest. The resulting image shows the vessels as lighter than the surrounding tissue.

To visualize the movement of these vessels, several angiograms can be taken per second, by using a ciné camera, instead of a film holder with a single sheet of film. Figure 2.4 shows a typical set up for performing ciné angiography.

Cardiac angiogram are recorded at 30 frames per second onto stock 35mm ciné film. The amount of exposure that the film receives is dependent on three parameters of the x-ray source; the duration of the x-ray flash, the amount of current delivered to the x-ray tube during that time, and the voltage at which the current is applied. Given that some small arterial branches can move as fast as 200 millimeters per second, an exposure time of 3 to 6 milliseconds is desirable. The amount of current is directly proportional to the photonic flux that is given off the by x-ray tube. X-ray tubes, however, are only about 0.5% efficient, and hence the amount of current that can be supplied to a tube depends on how fast the tube can dissipate its waste heat. In order to spread the heat over a larger surface, yet still keep the focus spot of the x-ray source small, rotating anodes, spinning at 20 000 rpm are used. The voltage at which the current is applied, while indirectly affecting exposure, directly influences the contrast of the resulting radiograph. The higher the voltage, the higher the frequency of the x-ray photons, and the higher their penetrating force through tissue, resulting in a lower contrast radiograph. For iodine contrast dyes, the voltage should ideally be 70 kV. However, for adequate exposure of the film (even through an image intensifier) this requires a current of 1000 mA, which is not possible with existing technology, nor is it very safe (too much radiation exposure to the patient). As a result, most angiogram are performed with tube voltage from 90 to 110 kV.

While conventional films will expose under x-rays, this exposure is not enough to form a usable image. As a result, image intensifiers are place before the cameras, mainly to convert the x-ray photons into visible light, but also to accelerate the photons over 2000 times, which further reduces the patient's radiation exposure, and permits shorter exposure times. Image intensifiers for ciné angiography are usually about 20 centimeters in diameter, and have a resolution of about 3 line pairs per millimeter[15]. Taking framing into account, the resolution of the resulting images on the 35mm film (18 × 24 mm) is about 25 line pairs per millimeter.

To aid the cardiologist in guiding the catheter to its destination, a small portion of the output of the image intensifier is directed to a video camera for real time fluoroscopy. The result is displayed on a monitor, placed where the cardiologist can see it. Typically, the activation of their the 35mm ciné camera and/or the video camera is done by foot switches located just by the cardiologist.

In modern angiography equipment, the whole assembly, the x-ray tube, the image intensifier, and the two cameras are mounted on a C arm, with the patient in the middle. This way the angiogram can be acquired from many different angles, with respect to the patient, and the source and cameras will always be in alignment.

2.3 Cardiac Angiography

In cardiac angiography, the dyes are injected into the coronary arteries, or into the ventricles, while an x-ray 'movie' is recorded onto 35mm ciné film. The former, called a *coronary angiogram*, is used to visualize the supply of blood to the cardiac muscle. The latter, called a *ventriculogram*, shows the functioning of the atrioventricular valves (the mitral and the tricuspid) and the movement of the ventricular walls.

2.3.1 Coronary angiogram

The principle use of coronary angiogram as a diagnostic tools, is to check the coronary arteries for stenosis or blockages. Chest pain or *angina*, is usually caused by an infarct in part of the cardiac muscle. Infarcts are essentially dead muscles, sometimes the result of infection, but, in the case of the heart, more often by lack of blood supply. When a *stenosis* (narrowing) or blockage occurs in one of the arteries supplying the heart muscle with blood (i.e. the coronary arteries), the muscle is usually damaged, leading to a heart attack. To find the problem, cardiac angiography is used to perform an arteriogram of the heart, which allows them to visualize the major arteries of the heart.

To deliver the contrast agent (the radio-opaque dye) to the coronary arteries, a catheter is inserted through a needle into the femoral artery just below the groin. The catheter is then guided up the aorta, right up to the aortic valve in the heart. At this point there are two branches off the aorta, the right coronary artery (RCA) and the left coronary artery (LCA). The catheter is inserted part way into one of these, and then the contrast dye is injected and its flow is recorded on film for two to four cardiac cycles. As some arteries will be fore-shortened because of their orientation with respect to the camera, this procedure is repeated several times, to record the arteries from several different angles. Most centres that perform angiograms of the heart have a standard set of angles, or *views* which are used for each examination. These views are labelled based on the position of the camera with respect to the patient. Usually, angiograms are recorded for both the RCA and the LCA (see figure 2.2).

2.3.2 Ventriculogram

In order to visualize the interior of one of the ventricles and its valves, a contrast agent can be injected, via a catheter, directly into the ventricles while filming. For the left ventricle, the same approach is used as for an arteriogram, except that the catheter is pushed right through the aortic valve, into the ventricule.

For the right ventricule, the catheter is inserted into the femoral vein, instead of the artery. The catheter is guided up the inferior vena cava, into the right atrium, and then through the tricuspid valve, into the right ventricle.

2.4 Principles of Ultrasound Imaging

The audible range of the human ear is from 20 Hz to 20 KHz. Ultrasound is a sound wave above 20 KHz, that like audible sound, transports energy through a medium. For ultrasound imaging, the sound is formed into a beam, and hence has the properties of both sound and a beam, such as attenuation, reflection, refraction, characteristic velocity in a medium, Doppler effect and straight propagation though homogenous materials.

2.4.1 2-D ultrasound

The velocity of sound through a medium is given by $c = f \times \lambda$, where f is the frequency of the sound, and λ is the wavelength. The speed, is dependent on the acoustic impedance, and the density of the material through which the sound travels, according to $Z = \rho_o c$. Like light passing between materials with different refractive indices, sound passing between materials with different acoustic impedances is reflected and refracted at the boundary. Ultrasonic imaging is based on this principle. If, for example, the ultrasonic beam is incident with the boundary at zero degrees, then the fraction of the energy which is reflected is given by:

$$R = \frac{Z_2 - Z_1}{Z_2 + Z_1}$$

where Z_1 and Z_2 are the acoustic impedance of the two materials. The intensity of this reflection, or echo, is measured along with the amount of time between the emission of the beam and the reception of the echo. From $d = \frac{tc}{2}$, where t is the time to echo, the distance of the boundary can be calculated.

When the intensity of the echo is plotted along a line at the corresponding distance, the type of ultrasound imaging is termed B-mode ultrasound. To get a two dimensional image, a sweep is made of the area, with about one hundred B-mode scans, and an image appears in the shape of a fan. If B-mode scans are taken of the same path, several time per second, this

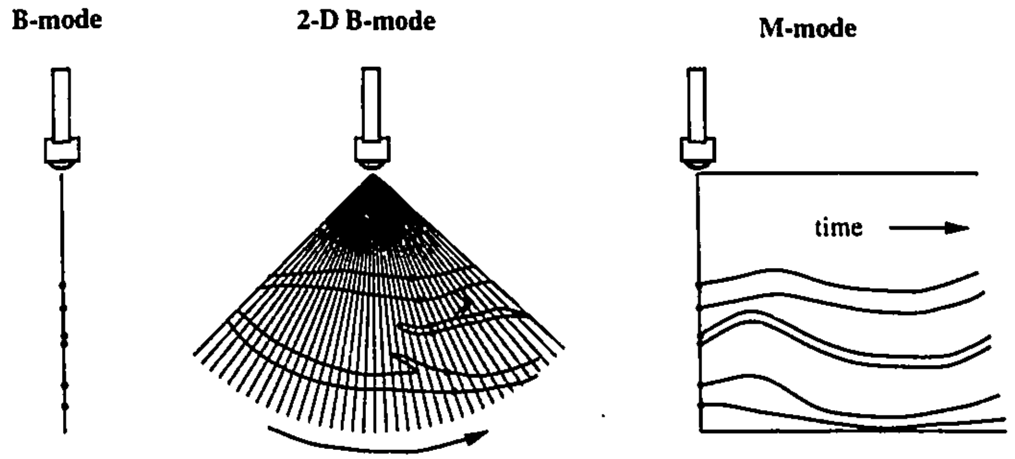


Figure 2.5: Illustration of 2-D and M-mode ultrasound imaging techniques.

sweep is called an M-mode scan. See figure 2.5 for illustration. This permits the visualization of the movement of a boundary[20].

In clinical ultrasound, the spatial resolution of the image presented to the viewer varies according to a number of parameters. This resolution, however is always less than the resolution of the NTSC video monitor that the images are displayed on.

2.4.2 Doppler ultrasound

The Doppler effect is a well know effect, where the whistle of a train, or the siren of a fire truck rises in pitch as it moves towards the listener, and then drops in pitch as it moves away. In ultrasound, this effect is used when a sound of known frequency is emitted by a transducer, and is then reflected by moving objects, such as red blood cells. When the echo is received by the transducer, its frequency will have changed. Assuming the angle of incidence to be zero, the change in frequency, or Doppler shift f_d, will be equal to:

$$f_d = \frac{2v}{c} f_o$$

where v is the velocity of the object, c the propagation velocity of sound through the material, and f_o the emitted frequency[43].

Typically, the Doppler shift is displayed by performing a Fourier transform on the echo,

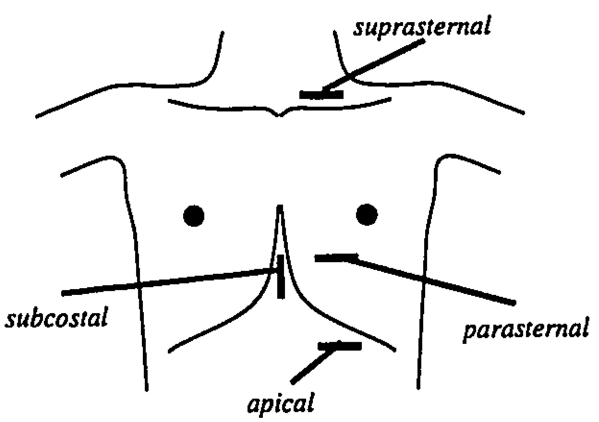


Figure 2.6: The four standard placements for transducers for an echocardiographic examination.

subtracting the base frequency (f_o) and displaying the results like an M-mode scan, except that the intensity of each frequency is plotted along the frequency shift, rather than the echo intensity along the distance to the boundary.

2.5 Echocardiography

Transducer placement

One of the characteristics of sound energy, is that it does not travel too well through gaseous or calcified media (e.g. air or bone). Hence to image the body's internal structures, the transducer must be placed over an 'acoustic window', so that the path of the ultrasound does not cross any such media. Named by the American Society for Echocardiography, there are four such windows, parasternal, apical, subcostal and suprasternal, as illustrated in figure 2.6.

2-D echocardiography

For 2-D echocardiography of the heart, there are three main *views*, or tomographic slides, that are used. The following is a list of them[20], and descriptions of their typical use.

Long axis: is usually taken with the transducer in the parasternal placement, in one of the spaces between the second and sixth rib, though also possible from the apical position. The image is a slice through the left ventricle, along the major axis of the heart. The parasternal long axis view is commonly used to view the left ventricle, the left atrium, the aortic root and aortic and mitral valves. M-mode tracings are used to examine the function of the mitral and aortic valves.

Short axis: when taken through the parasternal position, this view gives a minor axis slice through the heart. Typically used to examine the function of the left ventricle, the transducer can be angled to sweep from the sinuses, through the mitral valve, down to the apex of the left ventricle. At the sinus level, the opening and closing of the aortic valve is visible. Images at the mitral level show the movements of the leaflets of the mitral valve, and the mitral orifice. At the mid-ventricular level, the contraction and relaxation of the left ventricle is observed.

4 chamber: is taken by positioning the transducer at either the apical or subcostal placement, this view slices perpendicular to the IVS, presenting a view of both ventricles, atria and atrioventricular valves (mitral and tricuspid), and the IVS. In the apical position, rotating the transducer 90 degrees gives the apical long axis view.

Doppler echocardiography

There are fundamentally two types of cardiac Doppler techniques. They are: pulsed wave (PW) and continuous wave (CW). For PW, a transducer transmits a short burst of ultrasound and then after a delay, operates as a receiver for a short period, or 'window' of time. The delay is proportional to the depth of the object whose speed is to be measured. For CW, one transducer continuously transmits an ultrasound wave, while the a second transducer receives the echos. What is displayed, hence, is the sum intensities of each Doppler shift produced by all object along the path of the wave, hence the operator must ensure that the path of the wave crosses only on flow[20].

2.6 Other imaging modalities

Although still experimental, there are two digitally based imaging modalities that have been applied to produce cardiac image sequences. In the early 1980's, the Mayo Clinic built a high speed CT machine, capable of acquiring an entire volume at once. Through ECG gating, it was possible to acquire a time sequence of volumes. This machines, however, was used for research only, as it was very expensive to build and operate, and exposed its subjects (usually dogs) to unacceptably high levels of radiation.

Similar techniques have also been used on MRI equipment, which has none of the radiation problems. Using *gradient-echo* techniques with TR times of about 30 milliseconds and ECG gating, time sequences can be acquired of a live human heart. Again, it has been used mainly for research purposes, such as for ventricular analysis[45]. Its major problem is that the patient must be completely immobilized for up to six minutes while the study is acquired.

2.7 Clinical use of cardiac image sequences

Understanding how cardiac image sequences are produced is important to determine how they should be acquired and archived. In order to decide how they should be retrieved and viewed is more dependent on how the image sequences are used by the cardiologists to make decisions. In this section we discuss what general features the cardiologists look for when viewing cardiac angiograms and echocardiograms.

2.7.1 Angiography

A normal artery will gradually taper in size from source to its end, getting narrower as branches take away portions of its blood flow. When examining an arteriogram of the coronary arteries, a stenosis is characterized by sudden narrowing of an artery for a short section, after which it goes back to normal size. It is important to view the suspected stenosis over the whole heart cycle as the degree of stenosis will sometimes vary over the heart cycle, due to variations in blood pressure and artery position. Blocked arteries usually manifest themselves as either arteries that simply appear to end, rather than gradually taper, or as fading of the artery, due to the contrast agent not being carried up to the blockage.

Ventriculograms are useful for diagnosing a number of defects, such as deformations in ventricular function (e.g. infarct), atrioventricular valvular incompetence (i.e. valve regurgitation), septal defects (to the IVS) and hypertrophic myopathy (i.e. abnormally large heart). When the contrast agent is injected into the ventricle, the movement of the ventricular walls that are perpendicular to the plane of view are evident, and give clues as to the volume of the ventricle during the six stages of the cardiac cycle. It is also possible for a cardiologist to determine the approximate location of a myocardial defect, by seeing what part of the walls are not moving properly. Normally, during systole, the contrast agent goes out the aortic valve. If any is seen leaking into the atrium, then this is evidence of atrioventricular valvular incompetence. In the case of septal defects, both ventricles become visible as the contrast agent is allowed to flow into the other ventricle.

To view these studies, the cardiologist uses a special 35mm film projector. The film (on its spool) is place on the projector and the film is passed through a film gate, containing the shutter, on to the take up spool. The image is rear-projected on to a piece of frosted glass in front of the cardiologist. The film can be moved back and forth through the projector at a number of different frame speeds, usually between freeze frame and 60 frames/second. Usually, the cardiologist will run the film forward at 30 frames/second, scanning the various views one after another, until some artifact is seen. Then she will usually examine stills from that view, perhaps running the film back and forth a few times, while she builds a mental model of the problem. While the cardiologists are generally only interested in seeing the films running forwards, some will purposely run ventriculograms backwards, to better visualize the extent of mitral regurgitation.

As mentioned before, the resolution of 35mm ciné cardiac angiograms is about 600 by 450 line pairs per frame, which requires a digital representation of 1200 by 900, following a Nyquist sampling rate. Regardless of the spatial resolution, the temporal resolution must be variable, from freeze frame to 30 frames per second. Generally, all the information the cardiologist needs from a view comes from a sequences of frames covering one heart cycle. In some cases, not enough contrast agent is used, and so some arteries are visible during one heart cycle, and then the rest are visible during the next heart cycle.

2.7.2 Echocardiography

2-D echocardiograms can be used to diagnose many of the same pathologies for which ventriculograms are used. In addition, echocardiograms provide a direct visualization of the valves, and therefore permit a more detailed diagnosis. For instance, whereas a ventriculogram will show mitral valvular incompetence, a parasternal long axis echocardiogram will actually show the ruptured chordae tendon that is causing the valve to close improperly. Echocardiograms are also able to visualize obstructions in the ventricles or the atria, such as thrombi (large blood clots), myxomas (masses of connective tissue), or other emboli. Ventricular or atrial emboli can sometimes be seen in ventriculograms, but not always due to the fact that they may be hidden during the two or three cardiac cycles where there is contrast agent in the ventricle. An echocardiographic view can run for as long as the technician performing the examination deems necessary, and hence emboli are usually visualized. Echocardiograms can only provide indirect evidence of problems with the coronary arteries, by detecting infarcts, but cannot locate the stenosis as the arteries are not visible.

Doppler is often used in an apical long axis view in order to evaluate the mitral valve's performance. Regurgitation of blood back into the atrium is visible as blood flows away from the transducer.

Echocardiograms are recoded and stored on video tape (VHS or S-VHS). As a result, the cardiologist views them on a video monitor, using a conventional industrial grade VCR. The viewing process is very similar to that used for viewing angiograms. The cardiologist will usually scan the section of the tape containing the examination, looking at view after view, until something abnormal is spotted. The only difference is that video tape runs only at 30 frames per second, or freeze frame. Also echocardiograms are never viewed backwards, although this is partly due to the fact that VCRs generally cannot play backwards.

The spatial resolution of a 2-D echocardiogram is that of NTSC S-VHS video tape, which is about 640 by 480 pixels. The same is true for the temporal resolution, which is 30 frames per second. Except in cases of arrhythmia, which are better diagnosed through ECG recordings, all the information the cardiologist needs can be seen by continuously looping the images of just one heart cycle. The ultrasound scanners can differentiation 64 levels of gray (i.e. 6 bits per pixel).

2.8 Summary

Cardiac image sequences are clearly superior to still imaging modalities in diagnosing diseases of the heart. It is also evident that the current methods for viewing studies could be improved by first allowing cardiologists to view angiograms and echocardiograms through the same interface. Also, the physical storage requirements of ciné films and video cassette presents many problems and is very costly. Digital storage of these modalities would both reduce the space requirements, but mostly important, would improve the access time.

From the analysis of the resolution requirements, it appears that today's computer storage and display hardware are more than capable of meeting the needs of digital cardiac image sequences. Also, their flexibility would make access and analysis of the images much easier for the cardiologists.

In the next chapter, we will look at existing computer systems for storage and display of digital video, to see if they may meet our requirements.

Chapter 3

Computer workstations for image sequences

This chapter presents an review of computer systems that can handle video or short image sequences. Most commercial systems that can handle digital video are either for running interactive multimedia applications, or for video post-production. In the last chapter, we identified the technical requirements for displaying cardiac images sequences. The purpose of this review is to see if any of the current computer video systems can meet our requirements, or possibly give us clues on how we might do it ourselves.

To begin, a brief description of analog and digital video is presented. Several commercial systems for 'multimedia computing' and video post-production are described, followed by a description of two PC based systems developed specifically for handling short high quality image sequences.

3.1 Video

Thanks to the marvels of the human visual system, the illusion of continuous motion can be created by rapidly displaying a time series of pictures. This is the basis for ciné film and video. While ciné films are physically stored as a set of 2-D still images, that are visualized by mechanical means, traditional video is stored as a set of 1-D magnetic fields on some magnetic medium, such as magnetic tape. To get a 1-D signal from a 2-D image, the image is *rasterized*, that is, sliced from top to bottom, and the strips laid end to end. To display the image, the reverse is done.

3.1.1 Analog video

Traditionally, video capture, display, recording and playback relies on analog technology, where some measurable physical quantity, such as a voltage, indicates the brightness level of some point on an image at some point in time. To ensure that video transmitted at one place is viewable somewhere else, standards, such as NTSC for North America, exist. These standards were developed to fit within the technological limitations of the time, and yet meet the physiological and psychological requirements of the human visual system needed to provide the illusion of a continuous image with continuous motion.

According to the NTSC standard, a video image is divided into 525 lines for transmission. These images are divided into two fields, one field containing the even lines, the other the odd lines. All the lines in the even field are transmitted in 1/60 of a second, followed by the lines in the odd field. Therefore, in 1/30 of a second, the whole image is transmitted, but with natural scenes, it appears to the human views as if the images were being updated at a rate of 60 per second.

So that the receiver can recreate the transmitted picture, the NTSC standard defines a set of synchronization pulses to be mixed in with the image information. Figure 3.1 shows the a) horizontal sync pulse, which tells the receiver when to start a new line and b) the vertical sync pulses, which tell the receiver when to start a new field.

The spatial resolution of the individual frames is about 480 lines in the vertical direction. This is because 45 of the 525 lines in each frame are used for the vertical sync and the equalizing pulses. In the horizontal direction, the resolution depends on the bandwidth of the video signal. NTSC specifies a bandwidth $B_w = 4.2$ MHz for the lumenance of the image, which gives us $R_h = 340$ lines of resolution per frame in the horizontal direction, according to the following:

$$R_h = \frac{2B_wC_h}{A_rN_fF_s}$$

where $N_l = 525$ lines per frame, $F_s = 30$ frames per second, $C_h = 0.85$ the fraction of each line that contains line information, and $A_r = 4/3$ the aspect ratio of the screen (width/height)[19].

Sources of video are numerous. A video camera produces a video signal of the scene at which it is pointed. The camera can also be used to image a screen displaying a ciné film, provided that the film projector's shutter is synchronized with the video camera's 'electronic

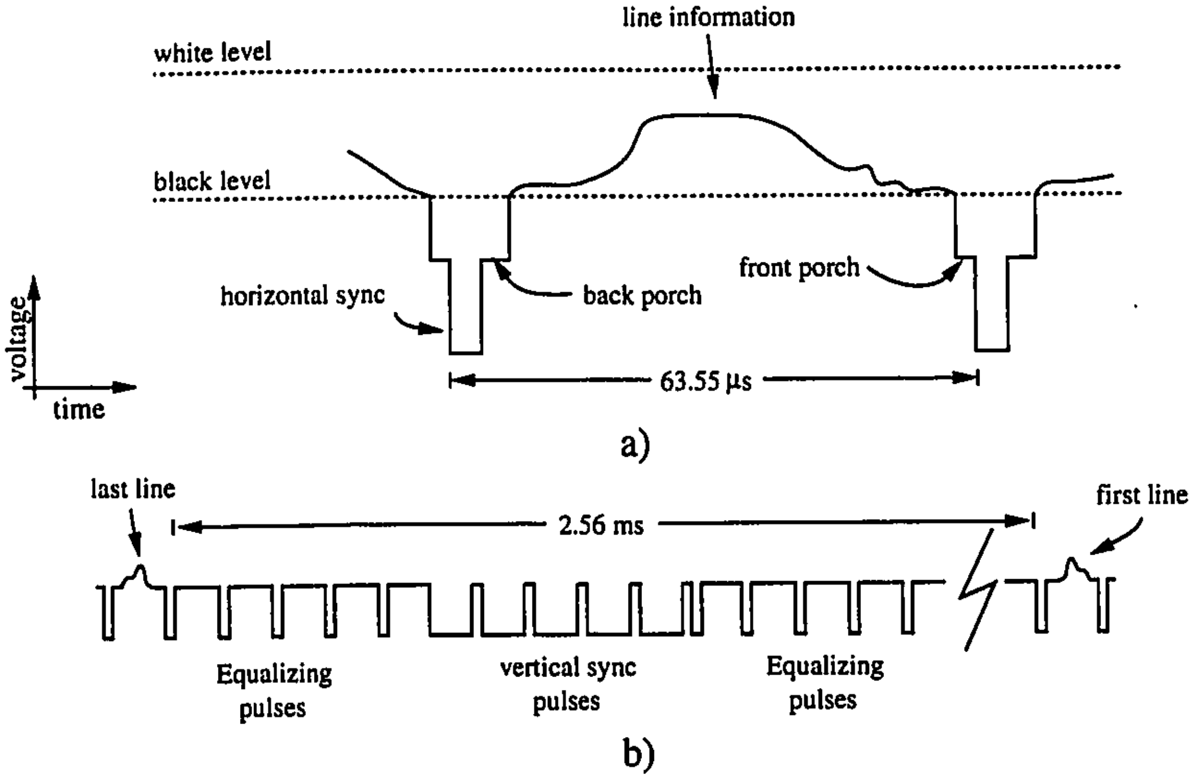


Figure 3.1: Synchronization pulse inserted into analog video signal to allow for drift in both the transmitter's and the receiver's oscillators[2].

shutter'. Another common source is a Video Tape Recorder or a Video Cassette Recorder, playing back a video signal that was previously recorded on magnetic tape.

3.1.2 Digital video

As mentioned before, in analog video, the brightness of a point in a scene is represented by measurable physical quantity. With digital video, that brightness is represented by a number. The advantage with numbers is that their storage and access is more flexible. For instance, digital numbers representing a sequence of video can be stored on magnetic disk for random access. Also, digital numbers stored on magnetic disk can be protected against change over time with error correction. There is no similar way to ensure that analog video stored on tape does not degrade from tape wear.

Digital video usually starts out as analog video, which is then *digitized*. If the source provides a steady video signal, such as one from a camera, the digitizing process is relatively simple. Given that NTSC specifies a horizontal resolution of 340 lines, we simply measure the brightness *at least* 340 times per line and store the numerical representation of these measurements. For example, the difference between a black signal and a white signal for NTSC video is about one volt. As the human visual system cannot differentiate more than 200 levels of gray, this one volt difference is usually divided into 256 levels to give an 8 bit number for each sample. This is done for each of the 480 lines that contain information, and stored as a frame, of 480 lines of 340 pixels each (i.e. $480 \times 340 = 163.2$ Kbytes).

In practice, 640 samples are usually taken per line[55]. This *oversampling* is done primarily so that, given the 4/3 aspect ration and the 480 lines per frame, the resulting pixels are square. In terms of the Nyquist limit, which states that a signal should be sampled at *more than* twice the signal bandwidth, given a bandwidth of 4.2 MHz, 640 samples per line works out to sampling at 3.76 times the signal bandwidth.

When digitizing video from a VCR, the mechanical recording and playback system can introduce errors in the timing relationship between the various synchronizing pulses. If, for example, the horizontal sync signal comes $10\mu sec$ late (i.e. about $\frac{1}{4}$ of a line), then the analog to digital converter will digitize the sync pulse instead of the line information. This Time Base Error (TBE) is correct by first passing the video signal through a Time Base Corrector. A TBC digitizes a few lines, using the sync pulses to trigger the digitizing, rather than relying on timing. These lines are temporarily stored, and then output at the proper time.

3.2 Multimedia computer workstations

Human beings can see in colour, focus on motion and hear keenly. As a result, we tend to comprehend information better if it is presented to us in colour, in motion, with sound. Information that is static and monochrome is seen as merely an abstraction of something real. This is the why broadcast television is so successful[59]. However, to be effectively used as an education tool, video has to be easier to create and to control.

For this reason, industry and academia have recently put a lot of effort into merging video and computers, to provide what has been coined *multimedia* computer systems. The aim has been to allow computers to used video and audio information as easily as they can currently handle text and static images.

3.2.1 Apple's QuickTime[57]

To give their Macintosh computers a standard way in which to display, compress, cut, copy and paste dynamic data, Apple developed the QuickTime extensions to their System 7 operating system. These extensions add two new file formats and three new managers to the operating system.

The new file formats are 'Movie', for storing dynamic data, and 'extended PICT', for still images and graphics. A 'Movie' file consists of *tracks*, which can be either video tracks or audio tracks. These tracks contain information such as time scales, screen co-ordinates, and pointers to raw video or audio data. QuickTime uses this information to display the temporal data at the correct time, and to synchronize the data to other information which may be displayed at the same time. The extended PICT file format, adds compression and browsing capabilities to the existing Macintosh PICT file format. To help a user browse a list of PICT files, each file contains a *thumb-nail* representation of the picture, which may be displayed next to the file name.

QuickTime uses three managers to handle the information. They are:

Movie Manager is responsible for loading and displaying the appropriate data at the right time. This includes ensuring that synchronization between different data *streams* is maintained.

Compression Manager selects the best compression scheme when compressing data, to ensure the best trade-off between quality, bandwidth and compression time.

Component Manager acts as an interface between applications and the underlying hardware. This allows applications to request resources in a descriptive manner, rather than a procedural manner. For example, an application might ask it to decompress some data. The Component Manager will either make use of some special decompression hardware, or do it in software if necessary.

To date, Apple supplies three compression schemes. The basic one is JPEG, a lossy, non-real time (without special hardware), high quality image compression algorithm. The other two are for real time video compression/decompression, one being for natural scenes (video), the other for 'noise-free' computer generated animation. These two algorithms use both inter and intra-frame coding to achieve a typical compression ratio of 6 to 1. This permits the use of CD-ROMs as storage, as 240x180 by 15 frames per second will fit into the 150 Kbytes/second bandwidth of CD-ROM, as long as the compression ration is at least 5 to 1.

There are numerous *Authoring* tools available for QuickTime on the MacIntosh, which combined with special digitizing hardware, allow users to capture analog video, store it on disk, and then edit it into a multimedia presentation, complete with text, graphics and sound[58].

3.2.2 Intel DVI[25]

Starting in the early 1980's as a concept at Princeton, and later implemented for the IBM PS/2 and IBM AT by Intel, Digital Video Interactive is a marriage of the control and flexibility of computers and the realism of television images and sound.

There are four unique elements to DVI:

- The custom VLSI chip set, which is the heart of the video system.
- A specification for a runtime software interface.
- Audio and video file formats
- Compression and decompression algorithms.

At the time that DVI was first implemented, general purpose CPU's found in personal computers were not powerful enough to handle digital video and audio. Hence Intel developed the i750 DVI Video/Graphics processor chip set to handle the digital video, graphics and audio. There are two special peripheral cards needed to use DVI, the ActionMedia Delivery board and the ActionMedia Capture board. The former contains the i750 chips, a TMS320C10 DSP chip for audio, video RAM, a CD-ROM interface, and the audio and video RGB outputs. The i750 chips perform all the necessary graphics and video processing, such as real time compression/decompression. The Capture board is for digitizing analog video; it passes the resulting digital video to the Delivery card for compression and writing to disk.

The Application Programming Interface (API) is composed of different modules which essentially provide a typical software interface to the ActionMedia boards and the CD-ROM drive. A script language, called MEDIAscript allows applications to be developed without writing any C language code.

3.2.3 Intel/IBM audio/visual kernel[10]

The Audio/Visual Kernel (AVK) was developed as a real time kernel to go with the second generation of the ActionMedia cards. Its architecture is divided into four layers. The lowest layer is the *microcode engine*, which is the interface to the microcode routines of the i750 chips on the ActionMedia II cards. The next layer is the Audio/Video Driver (AVD) layer, which controls the ActionMedia hardware and the microcode engine to provide a hardware independent interface. The third layer up is an Audio/Video Library (AVL) which models the system as a digital production studio. In the AVL, audio and video data are generalized into *streams*, which are collected into *groups*. A group collects streams that must be controlled synchronously. The AVL includes a set of functions that are gathered into *command lists* for execution. The last layer is the API, which integrates AVK into the hosts file system and windowing environment. For example, AVK can be integrated into QuickTime as a set of facilities.

3.2.4 VEX: Video Extensions to X[4]

For UNIX workstations using the X window system, Tektronix Inc. developed the Video Extensions to X. Their aim was to allow applications to merge graphics and video, using some of the newer display hardware that could also display live video in a window.

In particular, VEX provides the following facilities:

- Display an analog video signal (for example, from a camera or VCR) in a window, and allow the user or the application to move, resize or iconify the window, with the video still in it.
- Digitize a frame of video.
- Blending of live (analog) video and graphics displayed in a window. A special *blendmap* permits the application to specify the blending for an entire window, rather than pixel by pixel (e.g. 80% video and 20% graphics).
- Put the contents of a window out to analog video.
- Provide a framework for controlling the video hardware, including external devices, such as a VCR. The framework does not specify semantics, however, as there are so many different kinds of hardware with many different functions.

The availability of these functions are dependent on the hardware. VEX provides a facility to query what functions are available. Hence with these extensions and the proper hardware, it is possible to build multimedia applications in the UNIX workstation environment, similar to AVK applications.

3.3 Computer workstations for video post-production

While multimedia computer workstations have integrated video and audio into the computer environment, the video production industry has been more interested in adding the control of computers to the video editing environment.

To this end, special computers were developed to interface with VTRs. These computers record the time codes of start and end points of scenes on the unedited tape. These start and end time codes are arranged in proper order into an *edit decision list.* Once the edit decision list has been assembled, the computer takes over and automatically controls the source and destination VTR to assemble the final edit[19].

To reduce the expense and increase the flexibility of these systems, effort has been made to implement these on standard general purpose computers.

3.3.1 AT&T StudioMaster[60]

Using a combination of special hardware and software on the Macintosh platform, StudioMaster is essentially an automated video editing system. Using special interface cards, the software allows the Macintosh to control two Super-VHS decks.

Starting with the rushes on the source tape, the user identifies the start and end points of sequences they wish to form into the final edit. As the sequences are marked, the display card in the Macintosh (a Truevision NuVista+) digitizes the first frame of each and displays iconified versions of them on the Macintosh's screen. Once all the sequences have been marked, their order can be specified by moving the icons around. Once satisfied, the user merely has to push one button, and the software automatically controls the decks to produce the final edit on tape.

3.3.2 Fluency[61]

Fluency is very similar to StudioMaster, but for the IBM AT compatible platform, running Microsoft Windows. The hardware side is more comprehensive, providing hardware compression with a C-Cube JPEG chip, as well as analog or digital video in a window. The software is mainly a developer's kit with several example applications that are useful for video editing.

The heart of Fluency is FluentStreams, whose API is based on Digital Video Objects (DVO), which make the windows under Microsoft Windows aware of analog and digital video. There are three flavours of DVO's: one for displaying analog video from a live video source in a window, another for digitizing and playing back audio, and an audio/video object for manipulating digitized audio and video.

To edit video, Fluency operates in a similar manner to StudioMaster. The difference is that Fluency can digitize the entire source tape, and hence the user can select the scenes from the digital rushes. This is much faster as the user does not have to wait for the VCR to wind to the appropriate tape position when searching for a scene. Once the edit decision list has been compiled, the user can preview the final edit immediately. Producing the final tape is still done in the usual manner, as quality of the digital video is very low. Typically, the images are at one-quarter resolution (360x240), 15 frames per second with a JPEG compression ratio of about 20:1.

3.4 Medical workstations for sequential images

Systems, such as those sold by Nova Microsonics and Freeland[30], are aimed at providing quantitative analysis tools for echocardiographic studies. These PC based systems use proprietary video capture and display cards, which have enough memory to hold the sequences during capture and display. For stress testing, only systole need be captures which lasts only 350 milliseconds. Also, only a 320x240 window of the scanner output is captures. These two factors simplify the problem of high quality video loop capture, display and storage.

These systems can be integrated with some manufacturers ultrasound scanners, which allows the workstation to capture the image sequences directly from the scanner while the examination is being performed. Hence they provide facilities for the technician to operate both the scanner and the workstation. For other makes of ultrasound scanners, the image sequences can be acquired from video tape.

While the hardware for these workstation could be adapted to work with digital cardiac image sequences, as described in the last chapter, these workstations are sold as complete systems, and hence can not be used for anything, other than their intended application.

3.5 Summary

What we have presented in this chapter is the state of the art of computer based video systems. As for most video applications, these systems are designed to handle video sequences that last for a few minutes, up to a few hours. In order to cope with hugh memory demands of digital video, these systems decimate, drop frames and perform lossy compression on the video in order to get it down to a size that is manageable by current general purpose computer systems. For higher quality, these systems simply revert to using analog video.

As stated in the last chapter, one of the most necessary characteristics of cardiac image sequences is their high quality. Hence it is obvious that current general purpose computer based video systems are not applicable to a cardiac image sequence application.

Given that the average duration of a cardiac image sequences is one second, an approach similar to that of either the Nova Microsonics or Freeland is needed. The cardiac image sequences are first loaded into system memory from disk and then transferred to the display via a high speed bus (of the order of 10 Mbytes/second is idea). Aspects of this design will be discussed in more detail in chapter 5.

Chapter 4

The user interface and its design

Before reviewing the topic of user interfaces, it is best to get an idea as to where the term comes from. The term interface refers to "the place at which independent systems meet and act on or communicate with each other"[28]. Hence, the term user interface simply defines one of the independent systems as potentially being a human being, wanting to control the other systems. Also, the "place" of meeting is some part of the latter system, designed expressly so that the user can control the system. An example would be the handle of an axe; it is the axe head which does the cutting, but it is of little use without the handle through which it is controlled.

While this example might seem a little trifling, it is to make the point that user interfaces have been around forever, that people are very familiar with them, and that without a usable user interface, most machines are all but useless (e.g. an axe with a handle that is too short). However, for the rest of this chapter, we will use the term "user interface" to mean the user interface to a general purpose computer (e.g. not including embedded computers such as the ones commonly found in household appliances).

This chapter is divided into four main parts. The first deals with user interfaces from the point of view of the computer hardware design to receive and relay information to the user. In the second section we discuss some general user interface concepts, followed by a section describing various software to implement these concepts. Also, some examples of software systems to aid in the development of Graphical User Interfaces (GUI) are presented afterwards. The final section discusses user centred design techniques that have been developed to facilitate good user interface design, and consequently, good system design.

	data type	direct	continuous	absolute
keyboard	text	-	no	-
mouse	2-D	no	yes	no
trackball	2-D	no	yes	no
touch screen	2-D	yes	yes	yes
light pen	2-D	yes	yes	yes
stylus & tablet	2-D	no	yes	yes
joy stick	2-D	no	yes	no
voice input	text	-	no	-

Table 4.1: Classification of some basic input devices according to the data they convey to the computer.

4.1 Interactive devices

Today we think of computers as interactive systems, designed to help us do work. From the last chapter, it should be clear that their interactive nature is very important. In order to bridge the physical gap between the user and the computer, many different hardware devices have been invented and developed for the transfer of information from the user to the computer and from the computer back to the user.

4.1.1 Input devices

Since interactive computing began, hundreds of different human input devices have been invented. Here we will review only the more popular forms of data input, textual and location. Table 4.1 shows a summary of some more common and successful devices, classified according to the type of data and the way in which the data is conveyed to the computer. For a similar classification of some more esoteric devices, see [5].

Foley *et al.*[12] classify location indicating devices according to three independent characteristics: direct vs. indirect means, discrete vs. continuous domain, and absolute vs. relative location. With direct input devices, the feedback appears to be a direct consequence of the user's action, whereas indirect input devices require the user to learn some additional hand-eye co-ordination. Continuous devices allow the user to specify curves or gestures with the continuous motion of the hand, whereas discontinuous devices require the user to build it out of discrete actions. Finally, absolute devices allow the user to specify an absolute point in space with one action, whereas with relative devices, the user's action indicates a change from the present location.

The most common input device used for general purpose computers is the QWERTY keyboard. All potential computer users are able to use this, to varying degrees, to communicate alphanumeric characters, terse commands, and natural languages to a computer. While it is a good means for entering text into a computer, it does not do to well for communicating spatial information.

The most popular location input device is the mouse. It is both cheap and accurate. By using a variable mapping between the relative movement data provided by the mouse, and the final screen co-ordinates, the mouse can both cover a wide physical area, and be precisely positioned, down to the pixel. Trackballs, or thumballs are becoming popular for portable computers with Graphical User Interfaces (GUI). They are essentially upside-down mice.

Over they last few years, practical *voice inputs* systems have begun to find applications. Such systems can be classified according to whether they are speaker dependent or not (i.e. do they need to be trained to recognize a user), and whether they can recognize continuous speech or if they require the speaker to insert pauses between words. Speaker independent, continuous speech systems can recognize up to 200 words. Speaker adaptive, discrete word systems, on the other hand, can recognize up to 20 000 words[8].

Pausch and Leatherby [35] have successfully used such a system to supplement mouse inputs for a graphics editor. The voice commands are used in much the same way as 'hot-keys' are used under applications such as the Macintosh MacDraw graphics editor.

4.1.2 Output devices

For displaying text or graphics interactively from a computer, the most popular device, unquestionably, is the Cathode Ray Tube (CRT). There are a wide range of commercially available CRTs, with a wide range in capabilities, at a wide range in price. The most basic CRT is little more than a square oscilliscope screen, driven in a raster fashion, to present text and medium resolution graphics in green. Coupled with different graphics hardware, the resolution of such displays can go as high as 2000 lines, and can display as many as 16 million different colours. Accordingly, the physical size of the display area can go as high as a meter diagonal, though typically for most work situations, 50 cm is as large as is needed.

As portable computers becoming more powerful, their display requirements have increased as well. Thin, low power, light weight Flat Panel Displays (FPD) are slowly taking over where once only CRTs were capable. Active Matrix Liquid Crystal Displays (AMLCD) are now able to provide full VGA resolution (640x480, with 4096 colours), yet weigh very little, and are sufficiently power efficient to be put into portable computers[52].

Another form of computer output is sound. Almost every computer system, both general purpose and embedded, use "beeps" to alert the user to important information, or to indicate errors. In noisy environments, however, beeps can get lost in the background noise, hence digitally recorded voice messages are more effective. Not only do they convey more information than a beep (i.e. a specific message such as "Core temperature has reach the critical level") but they also stand out among the background noise as people subconsciously register human voice.

Speech synthesis, including text-to-speech systems, have had some success in the past as output devices for the blind. Simple lexically based systems, which break down words into their constituent phonemes, are able to produce recognizable speech at a low cost. More sophisticated language processing is needed, however, to correctly produce phrases such as "I read a lot", where the pronunciation of "read" is either "red" for past tense, or "reed" for present tense. Intonation control can also be added, to change the pitch and add pauses based on the word context. Future applications for speech synthesis include remote access to electronic mail or FAXs over the telephone[23].

4.2 User interface concepts

To develop a computer system for a given application, it is necessary to select the most appropriate hardware I/O devices. Determining the appropriateness of a piece of hardware, however, is not a simple task. In most businesses, cost is always an important factor, hence the makes of computer hardware have naturally strived to produce cheap, yet widely applicable I/O devices. The most popular interactive computer input and output devices over the last twenty years have been the keyboard and the video display. Within the last ten years, technology has made the handling of spatial information economical as well. In particular, mice, which are very flexible at communicating 2-D spatial information, can be had for as little as 20 dollars. Also, medium resolution monochrome graphics displays cost around two hundred dollars (though most users want, and are willing to pay for, colour and higher resolution).

As a result, there has been a lot of focus in developing software to make the keyboard,

	command language	menu selection	natural language	fill-in form	direct manipulation
user control	high	medium	high	low	high
learning time	high	medium	low	low	low
speed of use	high	medium	medium	high	medium
error-proneness	high	low	high	low	low
extensibility	high	medium	high	medium	medium
typing skill	high	none	high†	high	none

Table 4.2: User interface dialog style characteristics, according to Foley *et al.*[12] and Booth[3]. For natural language (†), no typing skill is required if voice is used as the mode of input.

mouse, and graphics display as easy to use as possible.

4.2.1 User interface dialog styles

The exchange of information between a user and a computer is often characterized as a *dialog*, where the dialog defines the exchange of symbols between the user and the computer, and the meaning that has been assigned to these symbols[3].

All user interfaces for systems with a 2-D display, a keyboard and a mouse can be classified into one of five user interface dialog *styles*. The oldest style is the *command line* interface style, where the user types in commands, using a specified vocabulary according to a strict syntax (e.g. the UNIX command line). Instead of requiring the user to remember a command vocabulary and syntax, a *menu* interface style allows the user to access the commands through a set of hierarchical menus (as in Lotus 1-2-3). Another way to avoid having the user learn command syntax, is to simply let them type commands in a *natural language* (e.g. English or French). The *fill in form* style of interface presents the user with a screen or window with a set of blank fields into which the user can type information (e.g. entering customer address on an order form). An interface with a *direct manipulation* style allows the user to manipulate graphical abstractions of the computer's resources directly, rather than syntactically (such as the Apple Macintosh file system). Table 4.2 compares these five styles according to six characteristics that should be considered when selecting a user interface style for an application's interface.

An important thing to note about user interface dialog styles is that they are not mutually exclusive. It is common practice to combine two of more different styles in a user interface, such as the print control panel on the Apple Macintosh, which is essentially a fill in form style interface within a direct manipulation environment.

4.2.2 Look and Feel

So far we have presented different types of user interfaces in the very general classification of dialog style. At a finer level of detail we also have to consider the specifics of the user's *interactions* with the interface, and the set of *constraints* that define the interface behavior. Together with the visual and layout guidelines for user interfaces, these make up the *look and feel* of the interface. The *look and feel* is usually a set of global specifications which are independent of any application.

Interactions are specified by *interaction tasks* and *interaction techniques.* The tasks classify the information that an application might need, such as position, text, selections and quantity. The techniques define how this information is communicated through the interface, including any feedback the system might give the user. Interactions can be formally specified in a state diagram, as shown in figure 4.1. As a result, several user interface design systems are based on state diagrams[12].

Constraints define the behavior of an object in terms of other objects. An example is where a slider is used to set an integer parameter. As the slider is moved, a label is constrained to show the value of the slider[6]. Hence constraints can sometimes be part of the interaction definitions.

Figure 4.2 shows three mechanisms, with different look and feel, for selecting from a list of mutually exclusive items.

4.3 Software for creating and supporting user interfaces

Given a set of input and output hardware, consisting of a CRT display, a keyboard and a mouse, the application developer needs software to help her control these devices. Besides providing a computer language interface to these devices, software can also help an application programmer to write consistent user interfaces.

For the *command line* and the *question and answer* dialog styles, little support software is

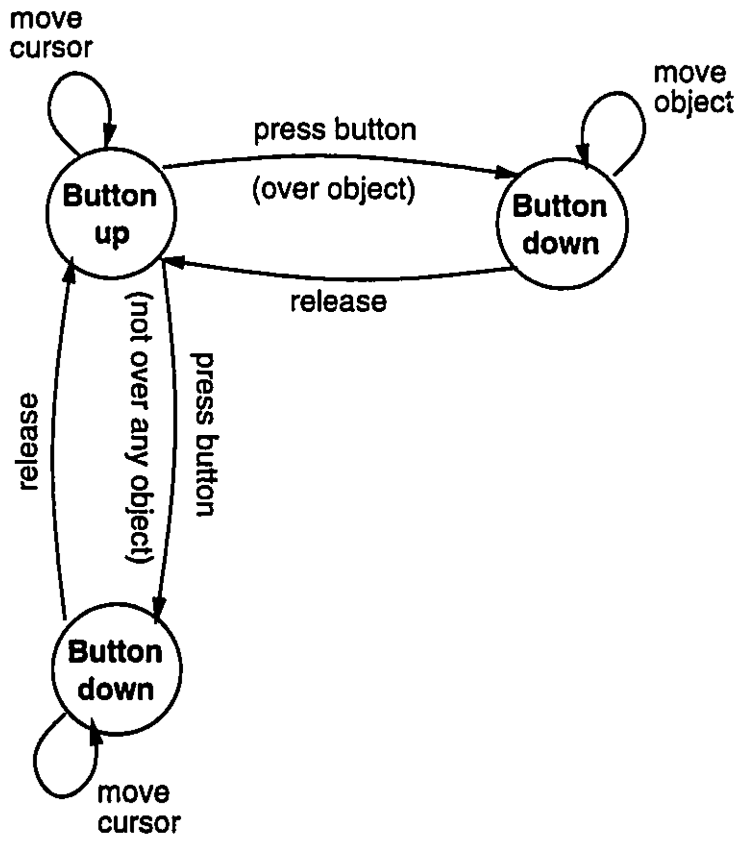


Figure 4.1: A state diagram defining the interactions of moving an object from one location on the screen to another.

needed, just simple text I/O system calls, such as `printf` and `scanf` in ANSI-C. For *natural language*, the same is true, except that natural language processing routines are also needed to make sense of the typed commands. The *fill in form* style requires more sophisticated I/O routines, able to randomly address any character on the screen. Libraries such as the `curses` library allow the programmer to write and read character to and from anywhere on the screen[47].

For *direct manipulation* dialog style, the programmer needs a graphical library that allows her to draw primitives like lines, text and ellipses, and to handle input from both the keyboard and the mouse. Programming applications in this way is very difficult, requiring a lot of tedious work. For this reason, most direct manipulation style interfaces are written for a windowing environment, using support tools, summarized in figure 4.3.

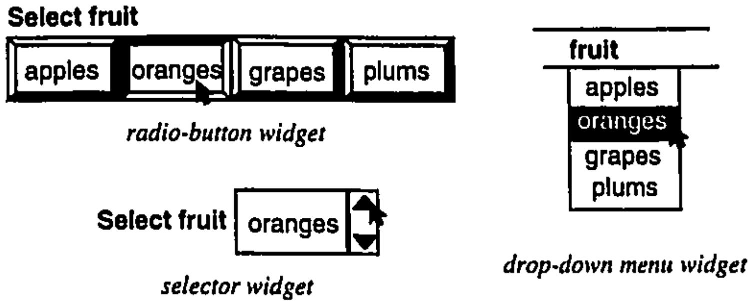


Figure 4.2: Three mechanisms with different look and feel that perform the same task; select one item from a list of mutually exclusive options.

4.3.1 Window management systems

With a large graphics screen, it is possible to display a lot more information than with a text-only screen. Window management systems attempt to help manage this information, by dividing the screen into windows, each of which can display information from different parts of a program or from different programs. To the applications programmer, it provides routines to create, delete, move and resize a window, and to draw into a window. It also handles situations much as overlapping windows, often without disturbing the application. The window management system also handles the input from the mouse and keyboard.

Often there is a distinction made between the *windowing system* and the *window manager*. While the windowing system does the actual work, the window manager is what the user interacts with to perform functions such as moving and resizing windows. The window manager determines the look and feel of the windowing system as a whole. For instance, there must be some policy describing which window will get input from the keyboard. This *keyboard focus* policy is implemented by the window manger, and may determine that the window in which the mouse pointer is in, gets the input.

Windowing systems treat input as events. When a key is pressed or when a mouse button is clicked, the windowing system will send an event to the corresponding program. Depending on what caused the event, the event data structure will contain information, such as the co-ordinates of the mouse cursor within the window, the key pressed, or the mouse button pressed. Also, the events are stamped with the name of the window in which the event occurred.

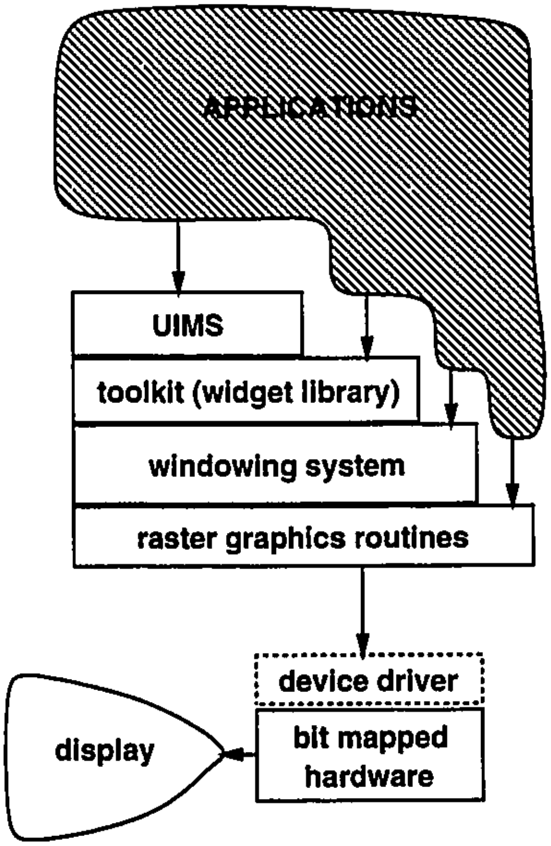


Figure 4.3: A summary of software for programming to a bit mapped display.

Events can be used for output too. When a program deletes one of its windows, another window may become exposed, and its contents will have to be redrawn. In this case, the windowing system sends an *exposure* event to the program, containing the co-ordinates of the window area that needs to be redrawn. Table 4.3 shows a list of the basic windowing system events.

The Apple Macintosh[54]

The Apple Macintosh was the first commercially successful computer to use a direct manipulation user interface as its primary interface. It was also the first to make popular the concepts of icons and window management systems. The Macintosh's windowing systems is tightly coupled to the operating system, in fact it is part of the operating system. Also,

KeyPress	Keyboard key was pressed.
KeyRelease	Keyboard key was released.
ButtonPress	Mouse button was pressed.
ButtonRelease	Mouse button was released.
Motion	Mouse cursor has moved.
EnterNotify	Mouse cursor has entered window.
LeaveNotify	Mouse cursor has left window.
Exposure	A part or all of a window has been exposed.
Resize	Window resizing has been requested (usually by the window manager).
Timer	A previously set timer has gone off.

Table 4.3: A list of typical events for a windowing system. Such events are usually time stamped, contain the name of the window in which they occurred, as well as other event specific information, such as the ASCII code of the key pressed.

there is no separation between the windowing system and the window manager, so the look and feel is consistent across all Macintoshes. Application programs can access the windowing system through low level system calls, though most Macintosh programs are written using the Macintosh toolkit, which provides a more convenient interface to the interface widgets. As all programs written for the Macintosh use these widgets, they all share the same look and feel at the widget level. To keep applications consistent within and among themselves, Apple publishes a set of hardcopy guidelines for developers to follow.

Microsoft Windows

Microsoft Windows is essentially a extension to the MS-DOS operating system running on IBM PC compatible computers, providing a co-operative multitasking windowing environment. The windowing system manages every system resource, except for the file system, which is handled by MS-DOS. These resources include output devices, window management, input devices, system memory and usage of the CPU. In this respect it is more a full blown operating system than simply a windowing system.

The Windows Software Development Kit includes libraries which provide a low level interface to the windowing system functions[7]. It is the windowing system which provides widgets, such as menus, sliders, and command buttons, and specifies the low level interactions and constraints.

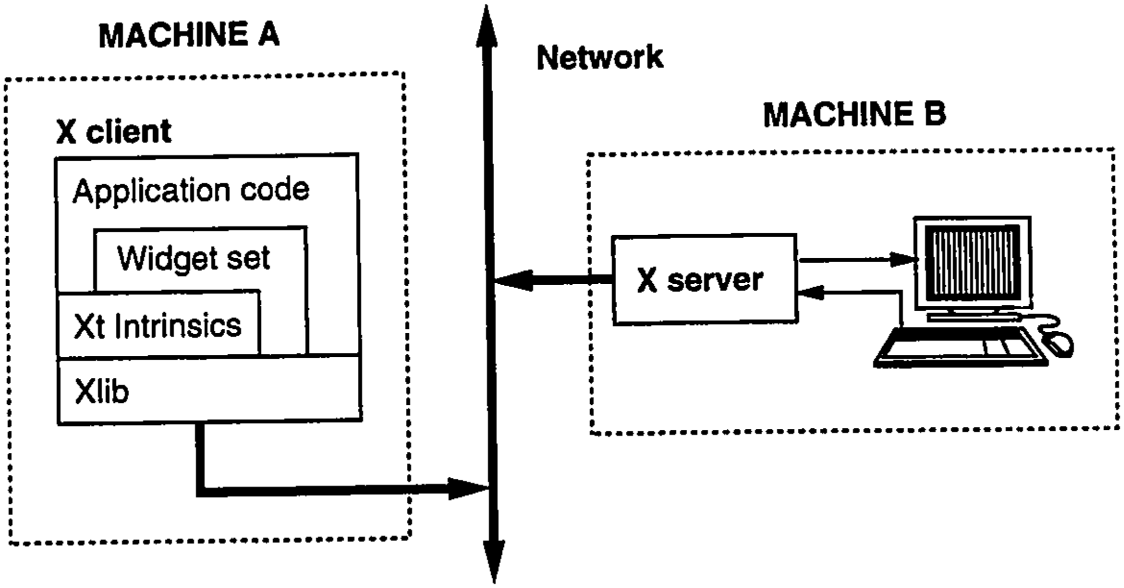


Figure 4.4: The X window system is a *client/server* where the input and the output devices are resources that are shared by clients (applications). These resources are managed by the server, through the window manager.

The X Window System[14, 50]

The X window system (or simply X) was originally developed at MIT to make use of the dozens of networked computer workstations with bitmapped displays. As a result, one of the underlying concepts of X is that its functionality is actually defined as a network protocol, and is designed to be portable over a wide variety of hardware and operating systems. Today, the continuous development of X is manged by the X consortium, a group of computer manufacturers who all use X as the windowing system on their hardware.

As shown in figure 4.4, the basic architecture is *client/server* based. The server is responsible for multiplexing the display, keyboard and mouse between clients (applications), which are either running on the local computer, or on another machine, connected by a network. The server naming strategy is hence based on the network node naming, so that clients request connections to a server by specifying the node on which it is running. Window management is performed by a special client of which there is only one per server.

Simple X clients are often written using a low level library called Xlib, which translates procedure calls directly into a network request for a server. More complex applications are written using a toolkit, such as Interviews which provides a set of widgets, or using the general purpose Xtoolkit and a separate widget library such as Motif.

4.3.2 User interface toolkits

Despite the fact that windowing systems provide many services to aid the application programmer, developing highly interactive interfaces can still be tedious. User interface toolkits provide the application programmer with libraries of widgets, like menus, command buttons, scroll bars and the like. Access to these widgets is done through the usually system language interface (i.e. subroutine calls). As a result, they reduce the complexity of the programming interface. The programmer no longer has to specify all of the interactions and constraints of her user interface, as these can be defined by the toolkit. All of the user interface widgets in the application share the same look and feel, hence applications are consistent at the widget level, making them easier to learn and use. Also, changes to the look and feel of the widgets can be done by reprogramming the toolkit, without modification to any of the applications written with it.

When writing a program with a toolkit, the program first initializes all of the user interface objects to be used. In parallel, *action routines* or *callbacks* are attached to the widgets. These action routines are called whenever a specified event occurs in a widget's window. The code that does the calling is usually part of the toolkit. It comprises of an event loop, which receives events and checks to see if anything should be done with them. This form of program control is referred to as *external control*, as it is not code written by the application programmer which controls the application.

Using toolkits, however, is not without its problems. Toolkits are still not a very intuitive way to build user interfaces. Even the smallest change to the user interface, such as correcting the spelling of a label, requires changes to the program code, and another compile and link cycle. The user interface code and the application code are tightly coupled, and highly dependent on one another, so changes to the user interface will probably require changes to the application code and vise versa.

To change the look and feel of a toolkit requires that a programmer rewrite some of the toolkit code. However, graphic artists and human factors specialists are better able to decide on how the look and feel should be, yet are not equipped to re-programme the toolkit. Another problem with toolkits is that they do not enfore consistency of the whole interface, only at the widget level. Hence programmers often have to rely on hardcopy style guidelines to ensure their applications have consistent interfaces.

The Xt Intrinsics[26]

In keeping with the policy and style free nature of X, the Xt Instrinsics library supplies all of the usual toolkit facilities, except for a set of widgets. It supplies all the necessary mechanisms for the development and running of widgets, such as external control and an interface for automatically calling action routines and other event handlers. Application programmers use a widget library with the Xt library, such as the Athena widget library, which is supplied with X, or another commercial set, such as OSF's Motif. The widgets are set up through calls to the Xt library, though some widget libraries supply "convenience routines" which are more efficient or easier to program, but do essentially the same thing.

The Athena widget set[36]

The Athena widget library is supplied with the X development system as an example of how to write a widget library. Although very basic, it is still a useful set of widgets, and hence most of the clients supplied with X are written using the Athena widget. Also, because it is supplied with the X distribution from MIT, almost every computer running X has the Athena widget libraries.

The Athena widget library supplies four sets of widgets. There are simple widgets, such as command buttons, scrollbars (sliders), toggles, labels, lists (item). Menus make up their own set, the variation being whether they select commands or items. A special set was constructed for dealing with text, including sink and sources. The last set is the composite and constraint widgets, which are for the layout of child widgets, and for providing viewports into larger widgets (e.g. a scrollable list widget).

The Motif widget set[31]

The Motif widget library is a set of widgets put forth by the Open Software Foundation, whose members include Digital Equipment Corp. and Hewlett Packard. In functionality, the widgets are similar to the Athena widgets, although visually, they have a 3-D appearance which gives them a more tangible feel.

In addition to the Xt Intrinsics interface and convenience routines, the user interface can be specified in a User Interface Language (UIL) which is compiled separately from the application code (the action routines). This allows look and feel changes to be made without requiring a compile/link cycle.

OSF also publishes a the Motif Style Guide[32], which lays out the basic UI design principles and discusses look and feel issues with respect to the Motif set of widgets. By following the guidelines set out in this document, all applications written with motif will have the same overall look and feel.

The InterViews toolkit[24]

The InterViews toolkit from Standford University is novel in that it is written in C++. This takes advantage of the fact that user interface concepts map elegantly into an object oriented environment. As a result, only about six thousand lines of code was required to implement a full set of widgets. The price to be paid, however, is that one must proficient in C++ to get at these widgets, so there is a steeper learning curve associated with using Interviews.

The latest version of InterViews attempts to make up for this by providing a WYSIWYG editor for building the user interface[56]. The code that drives the interface, however, is still mixed with the application code, making changes to either difficult.

The Andrew toolkit[33]

The Andrew Toolkit (or ATK) is part of a very large distributed computing environment developed by Carnegie Mellon University and IBM. Although the base computer language for writing Andrew applications is C, pre-processors allow the user to use object oriented *like* mechanism to interface with the toolkit. This provides some of the basic advantages of object orient languages without the overhead associated with learning to use them. As a side benefit, all parts of the user interface are treated uniformly as objects. These objects can be the usual buttons, and scroll bars, or they can be editable text objects, or graphics. The ability to add new objects makes Andrew very extensible, and as a consequence the look and feel is easily modified.

4.3.3 User Interface Management Systems (UIMS)

UIMSs attempt to take toolkits one step further towards efficient UI programming. Most important is that UIMSs separate the user interface from the application code. This allows someone, such as a graphic artist to design the user interface, while the programmer writes the action routines. Modern UIMSs, like Interviews[24], allow the UI designer to build the UI with a WYSIWYG editor, where the designer uses a drawing program to specify the UI. This drastically reduces the amount of time that it takes for a designer to learn how to create an interface. Older UIMSs skipped the programming of the interface by having the UI designer specify the UI interactions in some formal specification language, such as BNF, or graphically as state diagrams.

HyperCard

HyperCard for the Apple Macintosh is widely used for user interface prototyping because of the functionality and flexibility of its design tools [17]. With HyperCard, the developer draws the user interface with a graphics editor, and then defines 'hot-spots' on the interface. When the user does something to these 'hot-spots' (e.g. click the mouse), an action routine, written in a script language called HyperTalk, is called. In most cases, however, applications prototyped with HyperCard are re-written using the standard Macintosh toolkit, as the interpretive language is too slow.

The Simple User Interfact Toolkit (SUIT)[34]

The Simple User Interface Toolkit, designed and developed at the University of Virginia, was originally intended to be an educational tool for courses in software engineering. There was a need for students to develop applications with high quality GUIs, across a range of hardware platforms and operating systems. The problem was that even the best UIMS required a couple of weeks to master, and they assumed the developer was familiar with the UIMS model.

The resulting system is essentially a toolkit interface to a database containing a description of the user interface. The toolkit interface is in C, and uses the Simple Raster Graphics Package (SRGP)[12] to achieve portability across MS-DOS, MS Windows, Macintosh, X, and Silicon Graphics GL. The result is that a SUIT application will have the same look and feel, regardless of the platform it is running on. One disadvantage of this is that their look and feel is different from all other applications running on the native platform.

Khoros' User Interface Development System (UIDS)[53, 41]

The Khoros system, from the University of New Mexico, is primarily a scientific visualization and data analysis tool. However, its developers realized the importance of have good interactive GUIs for visualization applications, and so developed a User Interface Development System to allow scientists and other technical people who are not primarily programmers, to develop GUIs.

Like SUIT, the Khoros UIDS is essentially a library front end to a GUI database. Unlike SUIT, however, the GUI is created and changed by editing the database file with an ordinary text editor. Khoros provides a tool, called `preview`, for examining the layout and testing the interactions of the interface while it is being edited. Once the GUI is complete, another tool called `conductor` is used to generate C code which, at runtime, will initialize the applications and create the GUI from the database file. This C code also contains the stubs where the action routines are attached. In most cases, the GUI designer can re-edit the database file and regenerate the C-code, without disturbing the existing application code.

The next release of Khoros (release 2.0, due out *real soon now*) is to have a more comprehensive UIDS, which can be used on top of the Athena, OpenLook or Motif widget sets. The `preview` tool will be an interactive WYSIWYG editor for building the GUI[1].

4.4 User centred design

Traditional computer system design methods, such as those presented by Fairley[11], evolved from the time when most problems encountered in the computer software industry were computational problems. Today the problems encountered in designing computer software are much different. The technology has advanced to the point where most software developed today tackles problems that are computationally well understood. As a result, developers are now able to see and concentrate on solving usability problems in the interactive software they write.

The user centred design method differs principally in that it first considers the design from the point of view of the user. For example, questions like "how will the user enter the datum" and "how will the user tell the system to process the datum" are examined in more detail, often before determining, or independently of determining, how the system will process the

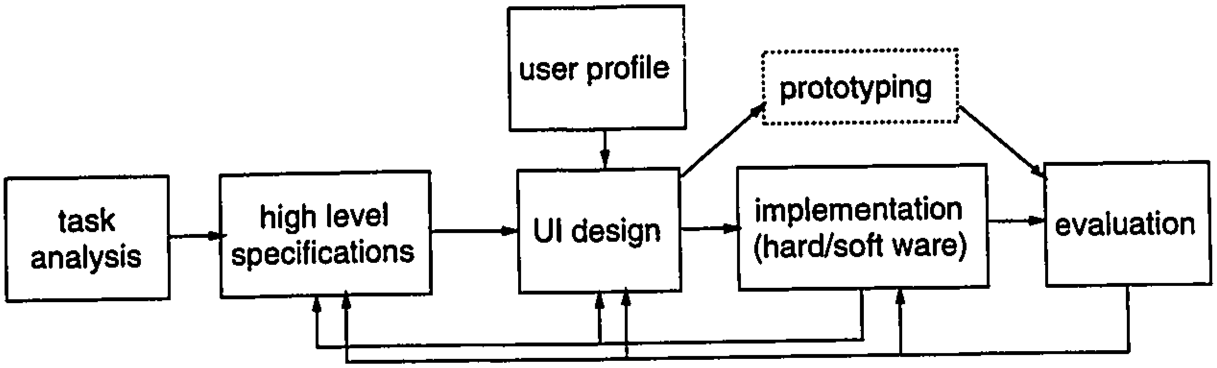


Figure 4.5: A user centred design methodology designs the system from the point of view of the user, and iterates until specific design goals have been met

data. Figure 4.5 illustrates the general procedure. First, a *Task Analysis* is performed, in order to yield a detailed understanding of the task at hand, called a *task model.* From this, formal specifications are produced. A *user model* is devised, and is combined with the task model and the specifications to design the user interface. This preliminary user interface must then be tested to rectify any errors in the task model or the user model.

4.4.1 Task analysis

The purpose of task analysis is to produce a task model. In the case of designing a system to automate some task, task analysis consists mainly of observing the intended users of the system. The results can reveal:

- the tools that people currently use,
- the frequency that the task is performed,
- the order in which tasks are done,
- the time spend on certain tasks, and
- the number and types of errors people make.

These results determine what functionality the system must provide, how it will be accessed by the user, what kind of errors must be protected against, and how the system will be incorporated into the current environment[40].

Task analysis must be as structured as possible. Observations and interviews must be done with the proper people (e.g. not the managers, but the clerks who use the system directly). The appropriate data collection methods must be chosen, based on resources available. While it is possible to record every key-stroke of a data entry system, analysing this data effectively, may take months, and may provide the same answers as asking users several well thought-out questions.

The resulting task model must be allowed to change. Often mistakes will show up when testing the designed system. Rather than fix the specific aspect of the system, it is better to correct the initial mis-understanding that casued the design flaw, and redesign from there.

4.4.2 User modeling

In designing a user interface, there are many decisions to be made which depend on who the user is. While task analysis attempts to deal with physically how and what the user performs, user modeling attempts to identify other important factors, such as why the user does a certain action, and what cognitive skills they posses to help them. User modeling tries to determine the psychological make up of the typical user[3].

A user profile may include factors such as age, physical ability, experience, education, attitude and motivation, and even subtle characteristics such as cultural background. Users are also classified as to how often they are expected to use the system. Naive users generally have to relearn most of the system each time they use it. Intermittent user remember what the system does (semantics), but forget how (syntax). Expert users remember both. These characteristics help us decide what kind of dialog style to employ, what kind of metaphors would be appropriate, and even how usable the system has to be.

User interface metaphors employ existing cognitive models that the user has of real world environments. The best known example of a user interface metaphor is the desk top metaphor found on Apple Macintoshes, which represents the file system of the Macintosh computer. While good metaphors reduce the time the user spends learning a system, a bad metaphor can have the opposite effect[39].

4.4.3 User interface evaluation

Once a user interface has been designed, it has to be validated. In using the results of the task analysis and the user modeling, numerous assumption may have to be made. The evaluation phase is where these assumptions are tested. Here we will look at four evaluation techniques, heuristic evaluation, usability testing, cognitive walkthrough and guidelines, and compare them based on expertise needed, cost, formality, and time required to perform the evaluation.

While these four techniques are presented separately, they are not mutually exclusive. It is common practice to mix evaluation techniques, for example, to use guidelines during design of the user interface, and then later evaluate via usability testing.

Heuristic evaluation

Evaluation of an interface by heuristic evaluation is performed by having several user interface experts examine and play with the user interface, to find errors and design flaws. The advantage is that this evaluation can be low cost, quick and informally done. The main shortcoming is that the quality of the results are heavily dependent on the level of expertise of the testers. Miller and Jeffries[27] estimate that it takes six non-experts to get the same results as one expert. Another shortcoming, is that this method tends to identify a large number of relatively un-important problems, that are more due to the personal opinions of the individual testers than scientific fact. As a result, when the report of problems are passed on, the developers must filter out these irrelevant problems from the important ones.

Heuristic evaluation can often be done two or three times during the development of a system, as it is relatively low cost.

Usability testing

Usability testing is the most expensive of all the evaluations methods. It attempts to study the behaviour of the user interface under real world conditions, using representatives of the intended user population. These users are then observed interacting with the system, or a prototype. If the test is well designed, usability testing can yield good results.

A well designed test has:

- real end users, many of them.
- a real environment,
- real tasks, not fabricated for the test,
- well defined goals[37].

It is important to know what one is looking for when usability testing. If we are looking for usability errors, then the strategy is to observe users and to note where they make mistakes. If we are trying to determine if a new system saves time, then we must time the users, and judge quantitatively the improvement. If the user interface is to be easy to learn, *protocol analysis* (having users think aloud) can reveal how the users are learning. Often, data can be collected by having users fill out *questionnaires*, after using the system[38].

To be realistic, compromises often have to be made when usability testing an interface[29]. The key is to understanding the consequences of these compromises, and minimizing their effects.

Due to the expense and time required to do good usability testing, it is often done only once or twice during the design of a system.

Cognitive walkthrough

This form of evaluation combines a software walkthrough with a cognitive model of learning through exploration[21]. It can usually be performed by software engineers, who look at every task the user may perform, and evaluate it with respect to a "check list" which has been prepared by someone familiar with cognitive psychology. As a result, it is a low cost evaluation method, that identifies most low level problems, and errors, such as inappropriate labeling. However, the process is often tedious, and does not find the global, more serious error, such as choosing an inappropriate metaphor.

As it is low cost and can be performed by the software designers, it is useful to perform cognitive walkthrough on sections of the software, as they are developed, and hence may be performed dozens of times during the development cycle.

Guidelines

Evaluating an interface with guidelines involves comparing various aspects of the interface with a list of "rules of thumb". These "rules of thumb" are usually reminders of good design principles. As they are very general in nature, having being written for a class of applications, they must be tailored to the particular application before they can be used directly[51]. This is especially important if several designers are working on an application's interface. For example, a guideline that states, "displays should be identifiable" might be translated to mean, "put the name centred in bold at the top" Another designer may decide to put the name in all capitals at the left. This then contradicts a second guideline stating that screen layout should be consistent.

Although somewhat tedious, guidelines are cheap to apply, and usually eliminate low level usability errors. Also, they can be applied at almost any stage in the user interface development; before, during and after implementation.

4.5 Summary

In recognizing the importance of easy to use systems, we have presented an overview of software tools and practices aimed at producing highly usable user interfaces. We have shown that there are numerous development and user environments that provide direct manipulation style interfaces, a style interface which is generally more intuitive to user than the traditionally syntax oriented interfaces. In addition to tools, we described a user centred design methodology, an iterative design process that looks at the system from the point of view of the user, in order to maximize system usability. It also acknowledges the fact that it is impossible to gauge ease of use without having people use the system.

By combining these tools and design methods, we can ensure that our applications are easy to use for everybody. Once these ideas are common place, developers of applications for end users no longer have any excuse not to produce easy to use systems.

Chapter 5

Design of a Workstation for Cardiac Image Sequences

In this chapter we present the preliminary design for a workstation for cardiac image sequences. By "preliminary design", we mean the design of the workstation up until it was first presented to test users for evaluation. The final design, which is to be presented in the next chapter, is reached only after usability testing has been done, and the workstation is considered satisfactory for day to day use.

This chapter is divided into five parts, according to the first five stages of the user centred design methodology presented in chapter 3. The first section presents the task analysis that we did at the beginning of the project and is followed by a description of the requirement derived from, in part the task analysis. Next is the user profile, describing in general term the makeup of our target user. The information contained in these three sections was then used to design out preliminary user interface, which is presented in the fourth section. The fifth section describes the hardware we chose to use, and the sixth section discusses how we used software to map our design onto the hardware platform.

5.1 Task analysis

In order to get a clear idea of our objectives, we did a task analysis of the current referral and consultation methods between attending cardiologists and the specialists who interpret the angiograms and echocardiograms at the University of Ottawa Heart Institute. The task analysis was arrived at by interviewing people at the Heart Institute, not just the potential users of the future system, but also people who they rely on or are relied upon, as they can often provide important clues concerning motivations and circumstances that might affect the users.

An initial analysis was carried out through interviews to determine what sort of hardware we were going to need for this workstation[46]. Later, this analysis was refined with further interviews, and by observing various people performing their jobs at the Institute. The results on this refinement are summarized in the following.

The University of Ottawa Heart Institute is a teritary care facility serving all of Eastern Ontario. Patients who visit the Institute are referred there by doctors from other hospitals for special care. Critical care patients are put in the eight bed Coronary Care Unit (CCU), where they can be monitored closely. Patients in the CCU are typically being treated for some pathology that requires frequent examination of the patient, and quick decision making on the part of the attending cardiologist.

Two frequently used examinations are cardiac angiography and echocardiography. For all patients in the CCU, regardless of pathology, about 50% receive an angiogram and 40% receive an echocardiogram. Whether or not they receive one of these exams depends largely on the problems they are experiencing and who the attending physician is.

For cardiac angiograms, CCU patients are "worked-in" to the schedule for the next two days of elective procedures. In more urgent cases, a procedure can be performed the same day. The Institute has three facilities for performing angiograms, one bi-plane Siemens system, equipped with a 1024 line fluoroscopic digital TV system, and a 35mm cine film recorder. The other two are older uni-plane installations. The examination itself is performed by a cardiologist who specializes in the procedure. The exact procedure will depend on many factors, primarily on the patient's condition. After the film from the examination is available (usually within an hour of the examination), the cardiologist will dictate her findings, describing how the procedure was carried out, and what, if any, pathological evidence was discovered. This report is later transcribed and then added to the patient's chart, for review by the attending physician. Sometimes the CCU cardiologist will talk to the cardiologist who performed the procedure, rather than wait for the typed report. This is possible as CCU and the CathLab are right next to each other, in the basement of the Institute.

Often the film and requisition are also sent to radiological sciences department of the Ottawa Civic Hospital, to be interpreted by a radiologist, and hence a second report will be available.

For echocardiography, the requisition process is very similar. The Institute has four HP, two Toshiba, and one Siemens ultrasound scanners, which are used to acquire echocardiographic studies. Usually, the examination is performed by a technician and then viewed later by an echocardiologist. The examination is sometimes performed by a echocardiologist, or by a technician with an echocardiologist present, or watching the exam on a monitor in a nearby viewing room. Again, the exact procedure depends on the patient's condition. While the exam is being conducted, significant portions are recorded onto video tape for later interpretation. If the echocardiologist is present during the exam, he or she will often prepare a preliminary report to go into the patient's chart. Review and interpretation of the video tape will be done in one to three days. The exam takes roughly an hour and produces about ten minutes of video tape on average.

While this system tends to work very well, with the reports being available to the attending physician in about a day or two, cardiologists tend to think spatially, and hence often want to see the film or video tape themselves. In the case of angiograms, they tend to view the actual study about 90% of the time, rather than just read the report. This is especially true where surgery or a procedure such as angioplasty is planned. For echocardiograms, the average is less than 30%. This is due in part to the difficulty involved in getting at the studies, since the attending cardiologist has to go to another floor, get someone else to find the right video cassette, and then search through two hours of tape for the one ten minute study they are looking for. Another reason for this is that most cardiologists, except for the echocardiographers, have relatively little experience with reading echocardiogram. This lack of experience is, of in part, due to the lack of accessibility of the echocardiograms. Figure 5.1 gives an overview of the flow of information between CCU and the CathLab and Non-Invasive Lab.

By providing the CCU physicians with better access to the examinations, we might not only reduce the amount of time they spend consulting the studies, but also increase their level of experience with echocardiograms, giving them another source of information to use when making decisions. Also, having access to the examinations in CCU will improve the continuity of information between the night and day staff. As the studies are available in CCU, they can be used as illustrations during the morning rounds, where the residents discuss the patients with the physicians who were on call during the night. Hence the examinations could be used as additional teaching material for the residents.

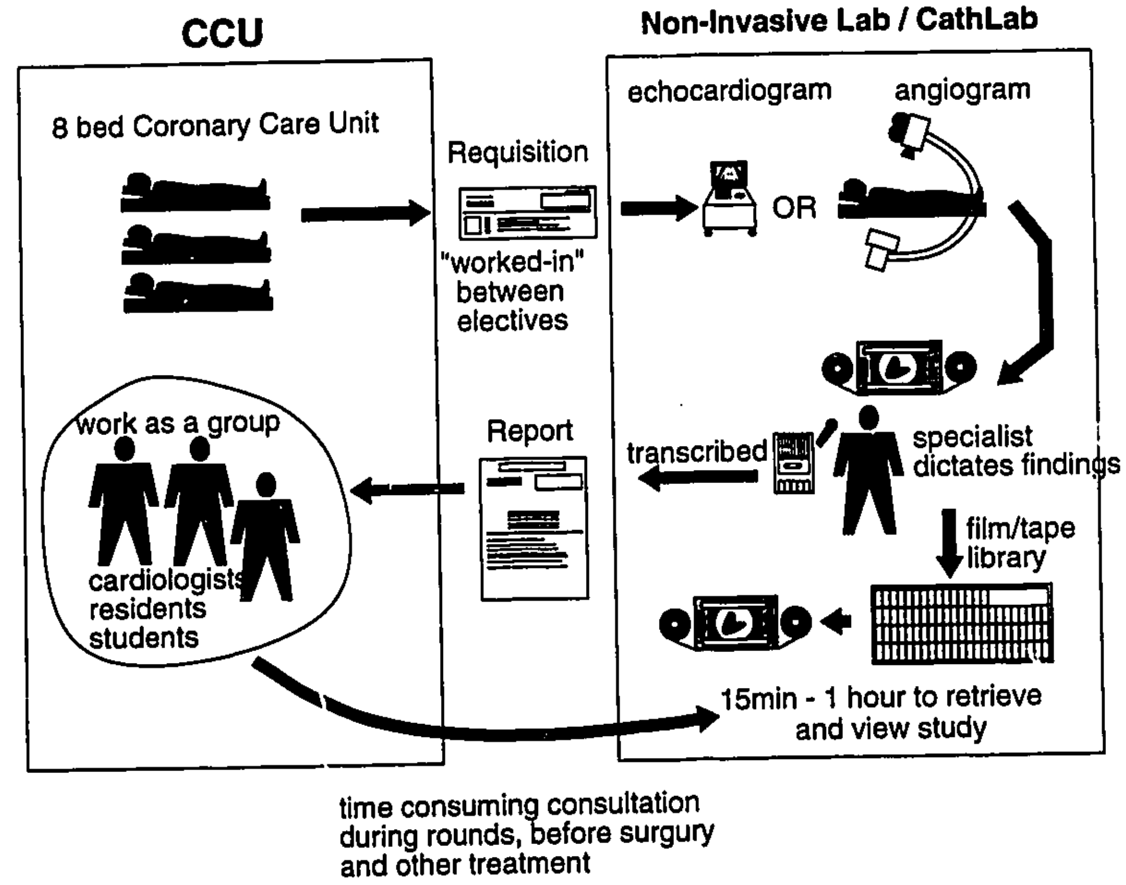


Figure 5.1: Flow of information between the CCU, and the CathLab and Non-invasive Lab.

5.1.1 Report formats

As mentioned above, there are three types of reports, related to angiograms and echocardiograms, that the attending cardiologists refer to. They are 1) the cardiac ultrasound report from an echocardiographer, 2) the cardiac catheterization report from the Heart Institute, and 3) the catheterization authenticated report from the Department of Radiological Sciences. The format of the reports is as follows:

Cardiac Ultrasound report contains the patients name, unique number, gender and date of birth. The name of the referring physician(s) also appear on the header. Below the header is a title, indicating the exact type of examination, (e.g. "Color Flow

Doppler"). Next is the reason for the exam, followed by measurements of relevant structures. Below that is the echocardiographer's comments (usually one paragraph) and a list of diagnoses. The echocardiographer's signature is on the bottom.

Cardiac Catheterization report headers contains the patient's name, date of birth and unique number. It also contains the date the procedure was carried out, the name of the physician performing the procedure and a "cath file number". This is followed by the patients height, weight and body surface area. There are eight sections to the report, 1) a description of the procedure, 2) heart rate and rhythm, 3) hemodynamic data, 4) various pressure measurements of the left ventricle and the aorta, 5) wall motion data, 6) report of the coronary arteriograms, including size, stenosis, and comments, 7) general conclusions, and 8) a cross index code. The cardiologist's signature is at the bottom of the report.

Catheterization report comes from the radiology department of the Civic Hospital. The patient's name, gender, age, birth date, unique number, registration number and visit number appear on the header. An order number precedes the referring physician's name and address. The report starts with the date and a description of the examination (e.g. "Coronary angiogram") with a number indicating the film number (something like "3161/92"). Next, come descriptions of the patient anatomy being examined. In the case of a coronary angiogram, there are four areas: 1) Left main trunk, 2) left anterior descending artery, 3) circumflex artery, and 4) right coronary artery. The report ends with the radiographer's "impression".

In creating these reports, the specialist will view the film or video tape of the examination, while dictating her findings into a tape recorder. This free-form speech is then later typed into a computer, using software specific to the modality, by a stenographer who specializes in transcribing medical dictations. The stenographer, with help from the computer software, enforces a structure upon the text, in order to eliminate any ambiguities. Naturally, this calls for some interpretation on the part of the stenographer, hence the typed report is later verified and signed by the reporting physician, before the report is added to the patient's chart.

From this report, the referring physician can gather most of the information that she needs to make decisions concerning her patient's treatment. In some cases, however, particularly when surgery is planned, the responsible physicians will want to view the examination, in order to form a spatial model of the patient's pathology.

Currently at the Heart Institute, accessing the films and video tapes presents a bit of a problem for the attending physicians. In the case of angiograms that were performed in the past couple of days, the physician need only walk down the hall to the viewing room, where the film projectors are located, and pull the appropriate study off of the self. As there about twenty angiograms performed per day, this does not pose much of a problem. In the case of past angiogram, where the physician may not exactly when they were performed, she must first look up the films ID number in a card catalog, located in another room. On average, the amount time it take to go from deciding to look at a film to actually viewing the film is rarely more than five minutes (unless all the projectors are in use).

For the echocardiogram video tapes, the process is longer. First of all, the physician must get someone else to find the appropriate video cassette, as there can be up to ten different examinations on a two hour tape. Once a technician has located the correct tape, and queued it to the required examination, the physician must go to the Non-Invasive Lab viewing room, on the first floor. The amount of time between decision to view the examination and actually viewing the examination can be as long as an hour. The main delay in this procedure is in the availability of the tape. If the examination was performed that day, the video tape is often still in a VCR recording another examination. Given that examination can take up to an hour, the tape will be unavailable for this time.

5.2 Requirements

From the task analysis, we determined that the system must have the following capabilities:

- Navigate the patient folders to select an image sequence, based on patient, exam, and view.
- View a selected sequence with variable frame rate, including freeze frame, and manual stepping back and forth through the frames.
- View two sequences simultaneously, with the possibility of synchronizing the two, such that the end diastolic frames appear at the same time, and the framing of the two be controlled with one set of controls.
- Control contrast and brightness of the image sequence though a control of the image's gamma value.
- draw lines, circles, polygons and text on a given frame, to be save for later Viewing. Each sequence can have any number of annotations.

- Enter a textual note about a particular image sequence.
- Measure features on the individual frames, both linear and area measurements. The units might be difficult or impossible to determine in some cases.

Another class of users are the technicians responsible for acquiring the image sequences. This user needs access to the workstation in order to digitize the sequences captured on S-VHS video tape. For them, we need a separate sub-system capable of:

- Creating a new patient folder. This includes entering the relevant information about the patient.
- Creating a new exam folder.
- Digitizing an image sequence from the VCR. The technician will do this by first positioning the video tape at the start of a sequence, and 'tagging' the current frame as the first frame. Then, the tape is advanced to the last frame, and it is tagged. The system then automatically takes control of the VCR and digitizes the image sequence.
- Once a good sequence has been digitized, all auxiliary information pertaining to it needs to be entered (view, date, etc.).
- Enter a textual note to be associated with a given sequence.
- Quit from the system at any time.

5.3 User Profile

The workstation's intended users would typically be cardiologists, or technicians, they are all highly educated and are generally highly motivated individuals. While they have exposure to highly complicated technical equipment, their work does not require them to use computer equipment, and hence computer skills vary widely.

We categorize our users based on the characteristics in Table 5.1[3]. The cognitive style (how they think) partly determines what style of user interface we can use. As our users frequently deal with spatial concepts and are able to analyse spatially, a direct manipulation style of user interface would be appropriate. Also lack of skills, such as touch typing, would rule out command line interfaces being practical.

The motivation and attitude of the cardiologists dictates that our user interface must be especially easy to learn and use. Their attitude is neutral as there is no reason why they

psychological characteristic	cardiologist	technician
cognitive style	spatial and analytic	spatial and analytic
attitude	neutral	neutral - negative
motivation	moderate	high
knowledge and experience	**cardiologist**	**technician**
reading level	high	high
typing skill	low	low
education	very high	high
system experience	none	none
task experience	expert	expert
application experience	low	low
use of other systems	low	high
computer literacy	low	low

Table 5.1: User characteristics, of cardiologists and of technicians

should try a new system, when the current system appears "good-enough". Their motivation is moderate, as they are probably willing to try something new, so long as it does not interfere too much with their current routine. As the technicians will essentially be delegated their tasks, their attitude may be low, but their motivation is high.

We cannot assume that all users have any experience with computer systems. Hence we must not try to adhere to any standard user interface guidelines, such as the Macintosh or OSF/Motif. In fact, the varying computer experience makes our job more difficult, because we will have to accommodate the subset of users who have experience with the MacIntosh and MS Windows interfaces. Each group will expect the system to behave like the systems they are used to working with. As it is impossible to have a GUI which behaves both like a MacIntosh and MS Windows, we will aim for the lowest common denominator; someone with no computer experience.

We can take advantage of the fact that our users have a high reading level and education, by providing terse help screens and messages. The fact that they are experts in what they do means that we have to design our system to fit in with their current environment.

5.4 The User Interface

Creating a user interface entails the design of processes which the user has to perform in order to accomplish a given task. In order to do this properly, we must know exactly what the user wishes to do (requirements), and how much trouble they are willing to go through in order to do the task. For this, we divide the design between the tasks the cardiologist perform and those that the technicians will perform. We consider the two cases, the cardiologist and the technician, separately.

5.4.1 General guidelines

In designing our user interface, we need not follow any specific style guidelines (such as the Motif Style Guide, or Apple's Human Interface Guidelines), as the software is designed to run alone. All aspects of our user interface will be tuned specifically to this application. However, in order to ensure consistency within the application, we first established a short list of guidelines for ourselves to follow during the design.

Only one mouse button is needed. User interfaces which use more than one mouse button tend to be confusing, as the user must be told, or must learn which button does what under which circumstances. We feel it is better to employ other mechanisms, such as double clicking, as users who do not use mice often, have to adjust their hold on the mouse in order to use the right button. The system will also prompt the user any time that they have the option to double click.

To convey necessary workstation status and error messages to the user, there are two mechanisms, a status line and popup windows. The status line will display instructions or information that naive users may find useful, but that experienced users can ignore. For occasions where the user must not ignore the message, a popup windows is displayed, which freezes the system until the user dismisses the popup by pressing a button on it. Popups can also be used for receiving confirmation from the user, for instance, when deleting an entire patient folder, we popup a window that asks the user if they really want to do this. Again, the system will be frozen until they press either the 'Yes' or 'No' button.

The design of our user interface must anticipate user errors, and provide mechanisms with which to deal with these errors. At any given point in the operation of the workstation, we want to limit the possible action that the user may take to those that are valid. For instance, pressing 'play' is not valid when no image sequence is displayed. One way to deal with this is to 'gray out' the 'play' button until a sequence is displayed. In some cases, it is better to let the user make the error, and simply tell them what they should have done. For instance, instead of graying out the 'play' button, let them push it, and then display a popup window informing them that they must first display an image sequence.

Given a colour display, we must be careful of our choices of colours[44]. While the use of bright, highly saturated colour can help to lead a user's gaze to a particular area of the screen, all other colour can be relatively muted so as not to confuse the user. Background colours can be either neutral or subdued to not conflict with the foreground colours. The total number of colours on the screen at any one time must be kept to a minimum, otherwise the screen will appear unnecessarily busy, and the user interface too complex.

After our initial user interface was designed on paper, we also checked it against the first four sections of the Smith and Mosier[51] "Guidelines for Designing User Interface Software", to eliminate as many bad design decisions as possible, early on in the development.

5.4.2 Sequence viewing and annotation

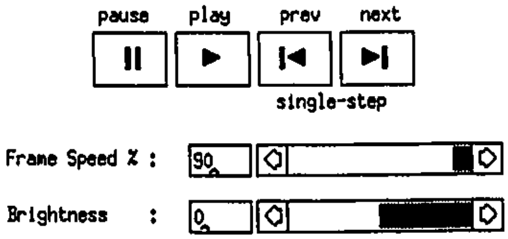
For this part of the user interface, a direct manipulation dialog style (as in the Apple MacIntosh) is appropriate for most of the system. The reason for this is because cardiologists are accustomed to working and thinking in a spatial fashion and are not willing to learn typed commands nor how to navigate large hierarchical menus.

The screen for the workstation will be divided into an area for displaying the image sequences, and another for workstation controls. It is better to keep all information on one screen, so that the user need only move their eyes to see everything. By putting the controls on one screen and the images on the other, the user will need to continually move the head to direct the gaze from one screen to the other.

The image sequence display area will be divided into two areas, for displaying two image sequences simultaneously. The control section is divided into a section for selecting image sequences (navigation) and control, a section for annotation and measurement, and a section for administrative functions, such as adding and removing patients and image sequences. The display of these three sections are mutually exclusive; the user selects which one is visible by clicking on one of three virtual buttons on the screen.

To navigate through the image sequences, rather than display a list of all the image sequences on the system, sequences are grouped into patient 'folders' to reduce the amount of information that is on the screen at any one time. Each line of the patient list consists of the first name, last name, unique ID, last name of the attending physician, date of birth and gender. Each line of the images sequence list consists of the modality name, the view, exam ID, and the date on which the study was performed. The user selects items from these lists by clicking once on the line to highlight it, and a second time to perform the action, which is either to bring up the list of image sequence for that patient, or to load the desired image sequence for display. When the user displays an image sequence, a toggle widget is used to select one of the two image display areas.

For the display of the image sequences, we employ a VCR metaphor for controlling the animation and selection of frames. A set of buttons lets the user 'play' the images sequence, 'pause' it, and single step through the sequence. Sliders are used to set any scalar values, such as frame speed or line thickness. The controls depicted in figure 5.2 are the result of a re-design of the user interface of the `animate` application supplied with Khoros.



Frame Speed % : 90

Brightness : 0

Figure 5.2: Buttons and sliders implementing a VCR style metaphor for the control of the image sequences.

As the workstation is intended to provide fast access to information, we will not bother with creating accounts on the machine for each physician who intends to use it. Rather, the physician simply walks up to the workstation and can start using it, in much the same way as someone can access a filing cabinet. Because of this, we cannot allow users to customize the workstation in anyway, in much the same way as we cannot allow every user of a filing cabinet to rearrange the files to suit their own needs.

5.4.3 Data entry

The task of data entry involves both the entering of the textual patient and sequence information, and in the digitization of the image sequence data. The latter task is, due to its mechanical nature, inherently difficult, requiring a complex sequence of operations and a significant amount of time.

For the entry of patient and image sequence description information, a form filling dialog style is the most appropriate, where the user fills in the first field and the system automatically advances to the next field when the user hits return.

5.5 Hardware and system software

Ideally our system will have a large screen capable of displaying simultaneously two image sequences and the necessary controls for manipulating the sequences and for navigating the patients and their examinations. The resolution of image sequences (when digitized) is 640 pixels by 480 lines, at 30 frames per second. To display two sequences simultaneously, we would need a screen size of 1280 by 1024 lines.

We calculate that the system needs at least 1.32 gigabytes of high speed disk storage for the image sequences. We arrived at this figure by assuming that each study is composed of at most six diagnostically relevant views, and that each view can be adequately represented by continuously looping one complete cardiac cycle i.e. about one second of video. Therefore each study can be captured with about 180 frames of video. Given that each frame has a size of 640 by 480 lines, and we only digitize in 256 graylevels, each study requires about 55 megabytes of disk space. Given an average of three studies per patient and 8 patients, we need at least 1.32 Gbytes of disk space for the system.

As for the speed of the disks, each second of video is about nine Mbytes in size, hence the imagery must be buffered in RAM before it can be displayed, as computers with disks, bus, and displays fast enough to deliver data at nine Mbytes per second are both quite rare and are still prohibitively expensive.

Figure 5.3 shows the hardware we selected for this system. In order to keep costs down, and to give us a better selection of peripherals, we elected to use an IBM AT compatible computer, specifically an Hewlett Packard Vectra 486/33T, which has a 32-bit EISA bus, with 64 Mbytes of RAM.

For the parallel disks, two Segate Elite 1.3Gbyte drives were used, each with its own Adaptec 1740 host adapter. Drivers from Chantal Corp. were used to stripe file systems across the two drives. As a result, we achieved a read rate of 3 Mbytes/second from the file systems on these drives. This is roughly six times the rate possible with a single drive on an ISA based PC.

For display, we used a Microfield V8, which under X windows gives us the 1280 by 1024 lines we wanted. To digitize the video, we used the HI*DEF card set from Imagraph Corp. For digitizing video from a VCR it is necessary to remove the time base errors from the video signal before passing it to the frame grabber. For this we installed a VT-2000 Time Base Corrector (TBC) made by Digital Processing Systems Inc. in an empty ISA slot, from which it draws. The VCR was a NEC PC-VCR; an S-VHS deck with an RS-232 port which allows it to be controlled under software.

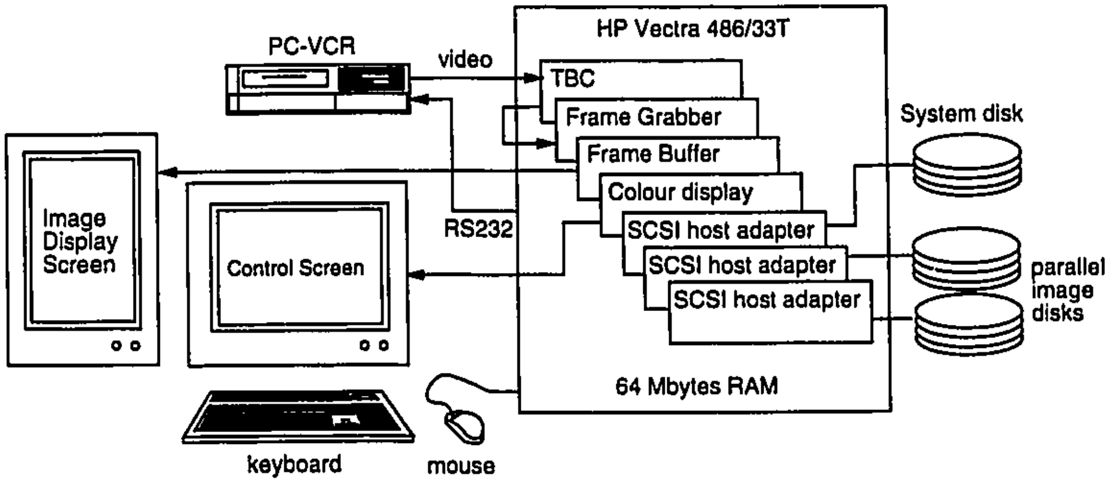


Figure 5.3: Hardware components of the cardiologist's workstation.

While we recognize the advantage of having a single screen which displays all of the information, the overhead associated with running the X windowing system on an ISA based card meant that we could not have the image sequences display on the main screen, and have them displayed at anywhere near the proper frame rate. Hence we decided to display the cardiac image sequence on a monitor connected to the HI*DEF frame buffer card. This monitor is usually only meant for previewing the video that is being digitized, however we wrote a special device driver that allowed us to easily and efficiently write images to the

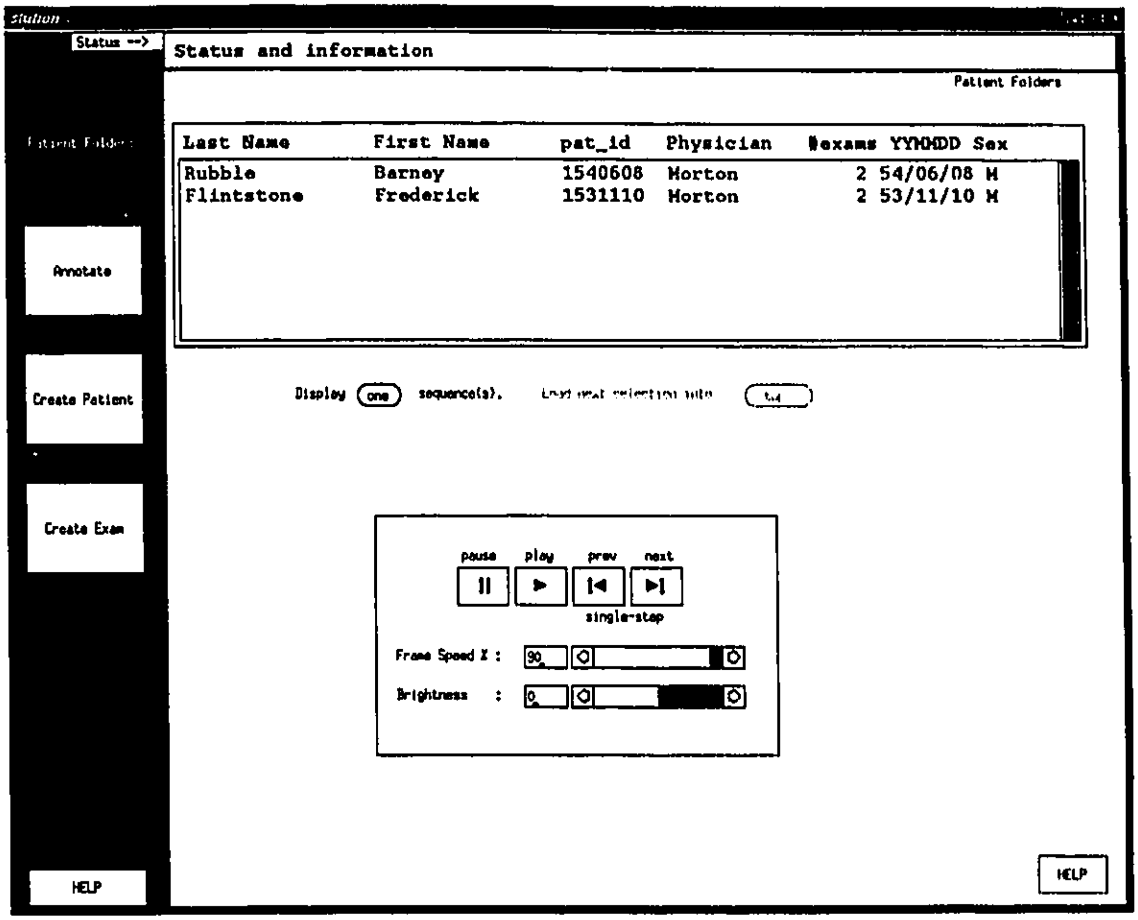
HI*DEF frame buffer. Still, as the HI*DEF cards are also ISA based, we do not have full video resolution, only 5 frames/second at 640 by 480 lines or 20 frames/second at 320 by 240 lines. While we realized that this might be a problem, we nevertheless decided to build the workstation with the hardware we had, and to later upgrade when better EISA display cards became available. Also, as the workstation is not intended for primary diagnosis, the spatial resolution of the images is less critical.

For an operating system, we chose the Santa Cruz Operation's version of UNIX System V for a number of reasons.

- UNIX provides a wide range of facilities to the application developer, that are not found in MS-DOS.
- Rapid prototyping tools for building User Interfaces are available for the X windowing system.
- The multi-tasking nature of UNIX simplifies the design of time-dependent tasks, such as animating image sequences.
- There is support for parallel disks.
- Of all operating systems for the Intel 386, UNIX is the one we are most familiar with.

For building the graphical user interface (GUI) under the X windowing system, we chose to use the UIDS facilities of Khoros. As discussed in chapter four, Khoros provides tools which allows one to build a GUI interactively, and then automatically generate skeleton C files[41]. The functionality is then added to these files. These tools also permit the developer to later change the GUI in the same fashion, and regenerate the skeleton files, without overwriting the existing function code. We chose Khoros over InterViews, Andrew, SUIT and commercial Motif UIMS because it was technology we were familiar with (C language, Athena widgets, Xt), we knew it was available and a stable environment, and it didn't cost anything.

Having the image sequences displayed on a different monitor changes our user interface somewhat. In particular annotating the images cannot be done on the second monitor, as the X server monopolises the mouse making it impossible to have the mouse work across both screens. So we take a snapshot of the image display, and copy it to an area on the control screen, where we use the annotation facilities of Khoros. The resulting user interface is presented in figures 5.4 and 5.5. For the image sequence display, a portrait mode monitor was used, so that two image sequences could be "stacked", one on top of the other, with textual information (such as exam ID and frame number) to the right of each sequence.



Status and information

Patient Folders

Last Name	First Name	pat_id	Physician	#exams	YYMMDD	Sex
Rubble	Barney	1540608	Morton	2	54/06/08	M
Flintstone	Frederick	1531110	Morton	2	53/11/10	M

Display (one) sequence(s).

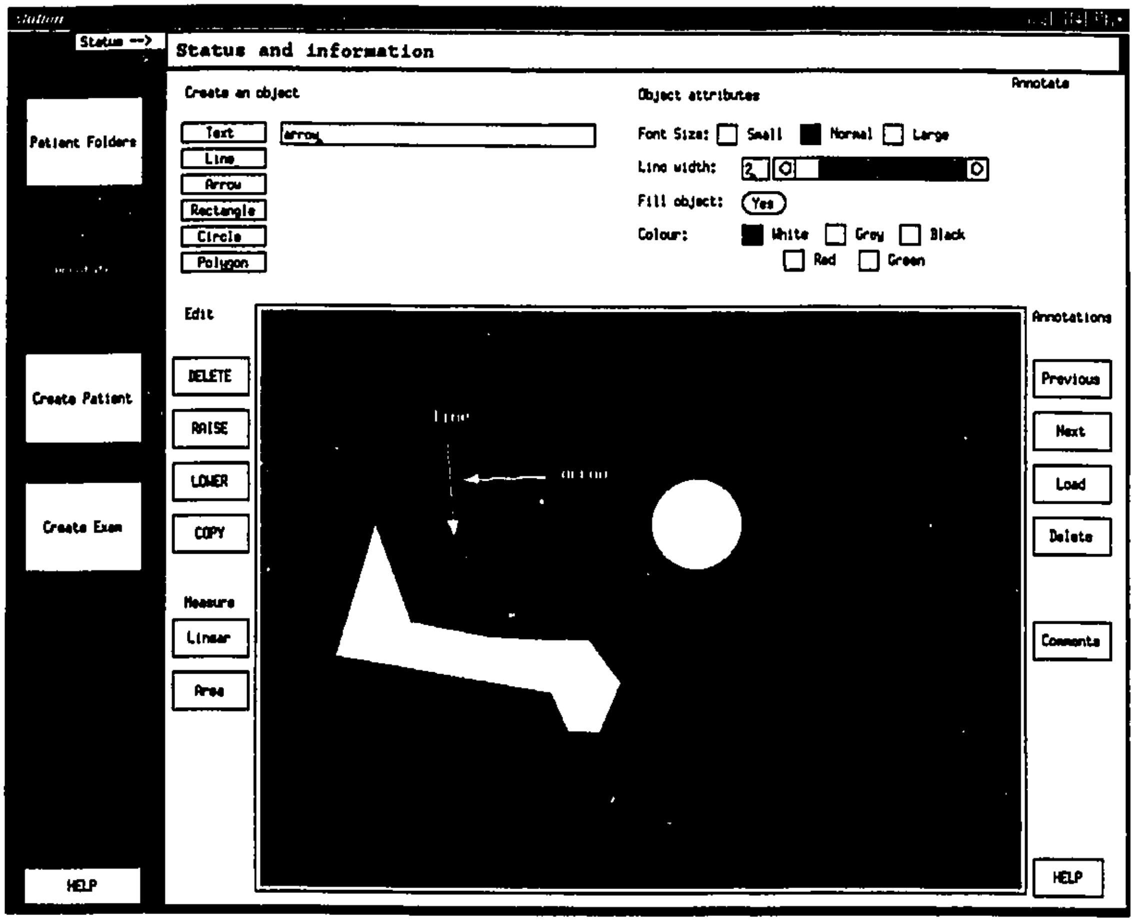
Frame Speed % : 90

Brightness : 0

Figure 5.4: The View Sequence screen, through which the user can load and control image sequences.

5.6 Software Architecture

The functionality of the workstation software is divided between two processes (i.e. programs) which run simultaneously and communicate through System V style message queues. The *display manager* is responsible for managing the control screen, and the patient and image sequence database. The *video server* is responsible for acquiring image data from the VCR, and for displaying the image sequences on the image display screen. We have imposed this division for two reasons. First, when loading image sequences off disk into memory, or when digitizing a series of frame, the *video server* need not constantly poll for user input, as the *display manager* takes care of this thereby avoiding presenting the user with a seemingly



arrow

Font Size: ☐ Small ■ Normal ☐ Large
Line width: 2
Fill object: Yes
Colour: ■ White ☐ Grey ☐ Black ☐ Red ☐ Green

Figure 5.5: The Annotation screen, where the user can transfer a still from the display screen, and annotation it.

dead system while performing time consuming operations. Secondly, this division permitted us to put all of the hardware dependent code in the *video server*, so that changing frame grabber, image display or VCR only requires changes to the *video server*. In fact, the *display manager* compiles and runs on SUN workstations with no changes to the code.

Messages between the two processes consist of requests from the *display manager* to the *video server*. An example would be to load a given image sequence for display. User actions such as pressing 'play', 'pause', or adjusting the brightness also result in requests being sent. The *display manager* also makes requests for the *video server* to display the output of the VCR on the screen, and to digitize a given number of video frames.

Status and error messages are sent from the *video server* to the *display manager*, so that the *display manager* may inform the user of the error, or, when possible, take action to correct the problem (i.e. restart the *video server* in case it should crash).

Each image sequence is stored in a flat, unstructured file, whose name is stored in the system database. Along with the patient information, and exam type, the database also keeps track of each image sequence's frame size (usually 640 by 480) and the number of frames. When a user selects a image sequence for display, the *display manager* extracts the filename, frame size and number of frames, and sends this information to the *video server*, enabling the *video server* to be ignorant of the database structure.

5.6.1 Data organization and management

From the requirements, we know that the database must store the following types of data:

Ciné loops: comprising, on average, 30 grayscale images of fixed width and height, which are to be animated from freeze frame to 30 frames per second.

Annotations: graphical objects, such as lines, circles, and text, to be overlayed on stills of the ciné loops

Text: unstructured ASCII text comments (made by the cardiologists).

There is also the miscellaneous patient and examination information, needed to index the above information.

There have been many sophisticated databases that been developed over the last few years that can manage the information described above. Karmouch *et al* [22] describe a multimedia document architecture and database design for the radiological environment, which is based mainly on the Office Document Architecture (ODA) standard [18]. This design deals with still images (radiographs) and voice, and hence solves similar spatial and temporal database management problems that one encounters with ciné loops.

While their mapping of multimedia information onto a relational database would seem suitable for our system, we chose to implement our own simple file based database for the following reasons:

1. We need to manage only two relations: one for the patient information, and another for the ciné loops.

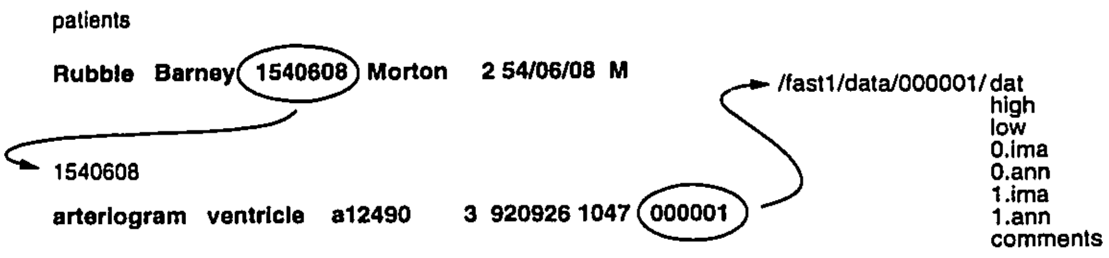


Figure 5.6: Organization of the patient information, sequence information, and of the image sequences and image annotations.

2. The number of entities in each relation that we need to keep accessible at any given time is minimal: 8 patients and up to 18 ciné loops per patient, for a total of 144 entities in the entire database.
3. Searches of the database are simple and always the same, i.e. find a ciné loop with a given view, taken on a given date, for a given patient.

The database, as implemented, has the following file structure. The patient information (relation) is kept in a file, appropriately named `patients`. For each patient, there is a file named after the patient's UNIQUE number. These files contain information describing the ciné loops of that patient. Aside from the type, view and examination ID, each record contains the name of a directory which contains the data for that ciné loop, including the annotations and the text comments. The directory names are unique. The text is stored in an ASCII file. The annotations are stored in files formatted by the Khoros graphics library routines. The ciné loops are stored in unstructured files ($height \times width \times \#frames$ bytes in size). Their structure is obtained from a data file in the same directory which contains the height, width and number of frames. This data file also contains all the patient and other ciné loop information associated with this ciné loop, so that the other database files can be recreated in the unlikely even that they become corrupted somehow. Figure 5.6 illustrates this organization.

The files containing the ciné loop data are unstructured because this data is first read into memory before the loops are manipulated. Hence the file is read in one go, with one system call, and the structure is created once the data is in memory.

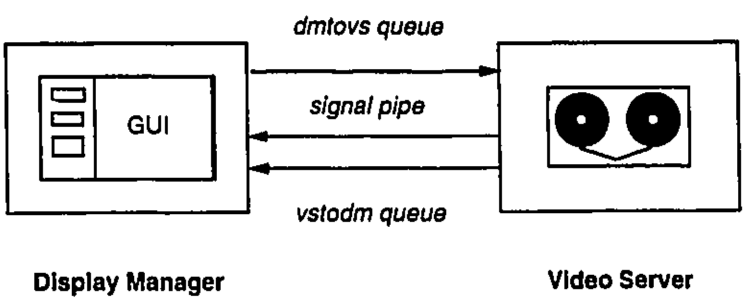


Figure 5.7: Division of the Cardiologist's Workstation into two processes.

5.6.2 Inter-Process Communications

The two processes communicate via UNIX System V style message queues, as shown in figure 5.7. The video server also uses a pipe to signal the display manager about certain asynchronous events, such as fatal errors. The file descriptor for this pipe is plugged into the Xt toolkit's select call of the display manager, so that polling is not needed.

The display manager is responsible for the presentation of the user interface. It receives actions from the user, sees that the UI is updated properly and delegates tasks to the video server as necessary. It is also in charge of maintaining the image sequence database; recording and display patient and exam information, and translating this into the filenames of image sequences as needed.

The video server is responsible for recording and play back of the image sequences. This involves reading and writing files to and from the file system, configuring the HI*DEF cards for operations, copying images to and from the HI*DEF cards for display or grabbing, ensuring (as close as possible) the correct timing for displaying and grabbing, and finally, the control of the VCR for digitizing the images.

For these two processes to communicate, there is a strict command language that is passed back and forth, via message queues. This language is presented in table 5.2.

5.7 Summary

To design our workstation, we have followed the user centred design methodology presented in chapter 4. In this chapter we have present the first four parts of this design process, the results of which are a workstation which meets all of the technical requirements, but whose user interface is still unsure. The evaluation of the user interface and changes that were made as a result, are presented in the next chapter.

command	parameters	description
load	directory file x y	Load into memory and display the sequence in *file* at (x, y).
play		Loaded image sequences is animated.
next		Display of sequence is stopped (if not already) and the next frame is displayed.
previous		Display of sequence is stopped (if not already) and the previous frame is displayed.
pause		Display of sequence is stopped (if not already) and the current frame is displayed.
clear	[bottom | top | both]	Clear specified display, and remove its sequence from memory.
speed	[0,100]	Change animation speed to a percentage of the maximum.
reverse	[0 | 1]	Reverse display contrast.
gamma	[-10,10]	Change display gamma as specified.
save	directory file x y h w	Takes an $w \times h$ image at (x, y) and writes it to *file*.
record	[cath | echo]	Server is reset, and the VCR's video signal is displayed, with gain setting determined by sequence type.
mark		If the VCR is in pause-play mode, the current tape index is noted as the beginning of the sequence.
grab		VCR is paused at end of sequence, server notes index and positions tape at previoused marked index. The server single steps through sequence until end, digitizing frames into memory.
keep	directory	Digitized sequence is written to disk in the specified directory.
abort		Abort record mode, any digitized sequence is discarded.
reset		Reset server to its initial state.
status		Server returns information on its status.

Table 5.2: The complete command set for the video server.

Chapter 6

Evaluation of the cardiologist's workstation

As discussed in chapter 4, it is difficult to determine how well users can operate a system without actually having them try it out. In the previous chapter, we have presented a workstation for cardiac image sequences that meets the technical requirements. In this chapter, we present a set of tests that were performed with the workstation in order to determine its usability and its suitability.

6.1 Goals

The objectives of these tests were as follows:

1. Complete the user centred design process by evaluating the design. Identify as many usability errors in the design as possible, so that they can be fixed, and the workstation rendered more usable (formative evaluation).
2. Determine, in a quantitative fashion, the usability of the workstation (summative evaluation). Our user interface must be intuitive and discoverable, as the cardiologist will receive little training.
3. Given that we must trade frame rate for spatial resolution, determine which format is more useful; 320x240 at 20 frames/second or 640x480 at 5 frames/second.
4. Identify the limitations of storing and displaying only grayscale images.
5. Evaluate how successful the workstation is in CCU, and identify specific improvements that can be made to render its use more acceptable.

To satisfy the first two objectives, we performed usability testing on the workstation, with cardiologists as test subjects, real data, and a set of typical tasks, derived from our task analysis. The third objective was satisfied by having the cardiologists view and manipulate the image sequences at the two different resolutions. The last objective was met by installing the workstation in the CCU for one week, and having the cardiologists use it there for real patients.

6.2 Usability testing

Usability testing is a process where potential users are presented with a prototype of the system, and are given a set of *tasks* to perform, which together make up an approximation of a typical user's session. The actions and comments of these users are recorded and analyzed to uncover usability defects. This *formative* evaluation then helps designers in specifying later revisions of the system. Users can also be timed and questioned in order to obtain more quantitative assessment of the system's usability. This *summative* evaluation allows us to calculate the usability of the system after each revision, to ensure that fixing one usability defect has not inadvertently added new ones[3].

To test our workstation, six cardiologists participated, all of whom spend some portion of their time attending to patients. Over the period of a week, each of these cardiologists were individually put through a usability test aimed at uncovering usability defects and assessing the usability of every function of the workstation, except for the administrative sections.

The test is divided into five phases. The first phase consisted of a series of questions concerning their specialty, their level of training, their activities at the Institute, their use of the two imaging modalities, and their level of experience with computers.

The second phase was a short training exercise aimed at getting all the test users equally comfortable with the mouse, thereby reducing the influence of the different levels of computer experience on the test scores. During their five minutes of training, users were asked to move the mouse cursor through a set of squares, practice clicking on buttons, moving sliders, and scrolling through lists. The script for this can be found in Appendix A.1, and borrows heavily from the one used by Roger[48].

The third phase of the test was the usability test. Users were given fourteen tasks to accomplish[1]. The first tasks were easy and simple and gradually became more complex,

---

[1] See Appendix A.2 for the usability testing task script.

requiring one or more intermediate steps to be performed before the task at hand could be accomplished. Briefly, the tasks included bringing up a patient's list of image sequences, loading an image sequence, playing the sequence, selecting a specific frame in the sequence, adjusting the brightness of the screen, adjusting the the speed of the loop, simultaneously display two different views from the same examination, annotating the two sequences, creating a comment entry for the examination, loading a different patient's image sequence and editing a mis-labeled annotation of that sequence. As users performed the tasks, they were asked to 'think aloud', so that we would know what their understanding of the task was, and what cues they were or were not getting from the user interface. For this, audio recordings were made of each user.

After completing a task, users were asked to rate how difficult they found the task was to complete, on a scale of one to five: one for very easy, two for required some thinking, three for possible, yet less than obvious, four for difficult and five very difficult or impossible. To consider our design a success, the average rating for all of the tasks should be two or less. A rating of three or more indicates that something needs changing.

Also, the time a user took to complete a task was measured, from the time the tester had finished reading the task and the user gave their rating. We measured the time from the audio recordings. We did not set any time limits for task completion.

The fourth phase of the test was another series of questions, asking the user about the appropriateness of the sequence labeling, and patient information, the speed of the animation, and system response. The users were also asked to rate, on a scale of one (definitely) to five (definitely not), the chances of them using this workstation, if it were put into the CCU the next day.

The last phase of the test was actually the image sequence resolution test, discussed in section 6.2.3.

6.2.1 Formative evaluation

During the course of the testing, users made mistakes and comments which could all, in some form or another, be used to improve the design of the system. In total eleven usability defects were uncovered by the six users. Eight of these defects were due to us making unreasonable assumptions about our users knowledge or perception. The user was lacking was instructions or cues on how to perform certain operations. These defects were corrected by changing the wording of some user interface labels, or by reinforcing other information with instructions

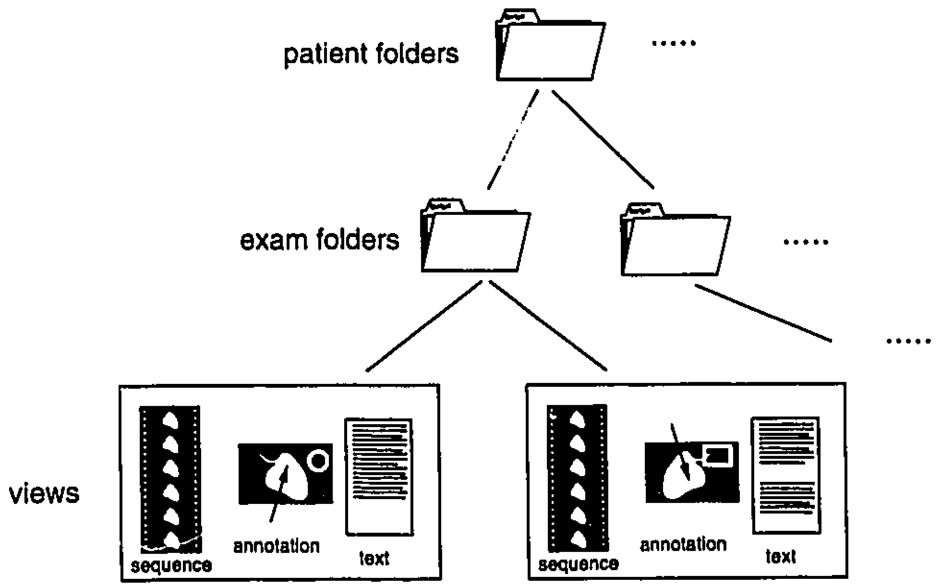


Figure 6.1: Alternate patient information organization, as suggested by cardiologists. Currently, all of a patient's views are in one folder, rather than divided into exam folders.

on the status line or in popup windows. Two other defects were due us not considering certain conditions that might occur as a result of user experimentation. These were fixed by adding code to handle these exceptions. The remaining defect was caused by a coding error. Table 6.3 (see end of chapter) lists the defects and the specific corrections that were made.

Two users suggested that the annotations not be saved, or that if they are saved, that the user be required to give some form of authentication. This is to avoid the unpleasant situation where a student indicates an obvious lesion on an image, but misses a more serious, subtle abnormality. Later, the physician responsible for this patient sees the annotation and takes it as gospel (if the computer says so, it must be true).

Three of the users specifically commented that they would prefer that the sequences from each examination be displayed as separate lists, reflecting the information organization shown in figure 6.1. So selecting a patient would bring up a list of examinations, and selecting an examination would bring up a list of images sequences from that examination. Comments were to the effect that this organization would create less screen clutter, and was more logical to the cardiologists, even though it would require the user to perform extra steps in order to do inter-examination comparisons of image sequences.

This suggests another extension to the workstation, where upon loading an image sequence for display, the system would automatically generate a list of image sequences on the system that the cardiologist may wish to compare with the loaded sequence. For instance, for cardiologists who wish to monitor the progress of treatment over time, the system would display a list of images sequences of the same modality and view from earlier or later examinations of the same patient.

An interesting problem was found, not by the users, but by those running the test. To digitize the sequences, the procedure was suppose to be the following:

1. Pause the video tape on the first (end-diastolic) frame of the image sequence.
2. Press "Mark" to record the tape index of this frame.
3. Advance the tape to the last frame of the sequence (also end-diastolic).
4. Press "Digitize" to record the index of this frame.
5. The video server will automatically rewind the tape to the first index, and then single step the VCR, digitizing each frame, until the second tape index is reached.

The problem with this procedure is that it required that indices be written to the tape, and PC-VCR requires this to be done to the *whole tape.* This indexing takes two hours, for a T120 S-VHS tape.

In order to get around this problem, we added two commands to the video server (shown table 6.1), and changed the digitizing procedure to the following:

1. Pause the video tape on the first (end-diastolic) frame of the image sequence.
2. The user sets the approximate number of frames in the sequence (e.g. if the heart rate is 60 bpm, then the sequence will be 30 frames in length).
3. The user tells the video server to digitize the frames, by pressing the "DIGITIZE" button.
4. The user then previews the looped sequence, and deletes any extra frames.
5. Once satisfied, the user presses "SAVE" and the sequence is written to disk and entered into the database.

The final user interface is presented in figures 6.3 and 6.4 (at the end of the chapters), except for the administrative function. The image display screen is presented in figure 6.5, showing a display of one angiogram and one echocardiogram.

command	parameters	description
grab	number	If the VCR is in pause-play mode, the server digitizes the current frame into memory. This is repeated a specified *number* of times.
delete		If a digitized sequence is in memory, the currently displayed frame is deleted from the sequence.

Table 6.1: Additional commands needed to support the alternate sequence digitization procedure.

6.2.2 Summative evaluation

After testing the six users, the average rating was under two for all but four of the tasks. Three of these tasks all had to do with annotation, the four involved changing patient folders. Even so, the ratings were still under three, and most of the high ratings came from the first couple of users before the major usability defects were removed from the annotation user interface. The average over all tasks, over all users, was 1.62.

To assess whether the system was getting easier to use or not, as a result of changes made to the system, each user's mean rating was multiplied by their mean task completion time to give the user's "score". Figure 6.2 shows roughly how the system became easier to use as usability errors were fixed between testing users. By the time we tested the last user, the score had dropped by 67%.

As far as previous experience with computers is concerned, our data showed that this had no effect on the users ability to perform the given tasks. By observation, we did note that Windows and Macintosh users would get confused for a few seconds, while they discovered the different interactions of the Khoros widgets.

Finally, as for the chance that the cardiologist would actually use our workstation, the average rating was 1.16, a resounding *definitely*, with only one cardiologist suggesting that we fix some of the usability errors first (which, of course we did).

6.2.3 Image sequence resolution

We had a choice of displaying the image sequences at a low spatial resolution of 320x240 at 20 frames/second or at a high spatial resolution of 640x480, but at 5 frames/second. To

Tasks		rating					average
		1	2	3	4	5	rating
1	sequence list	4	2				1.33
2	display sequence	5	1				1.17
3	play sequence	5	1				1.17
4	change frame rate	4	2				1.33
5	freeze frame	5	1				1.17
6	change brightness	5	1				1.17
7	display two sequences	3	3				1.50
8	prepare for annotation	4	1	1			1.50
9	place red circle	1	2	1	2		2.67
10	enter comments	4	2				1.33
11	change patients	2	1	1	2		2.50
12	display annotations	2	3		1		2.00
13	next annotation	3	3				1.50
14	edit annotation	2	1	2	1		2.33

Table 6.2: Occurrence of ratings given for each task. Overall average rating was 1.62.

determine which was preferable to the cardiologist, we had them compared the two with two high quality angiographic views, and two high quality echocardiographic views. The subject was to first load the sequence at one resolution, view it, and then load it at the other resolution. After each view, they were asked to state which they found would be more clinically useful.

All cardiologist found the 320x240 at 20 frames/second image sequences more useful, for both echocardiograms and angiograms. All of them stating that, first of all, they could not see any difference in spatial resolution between the two, and secondly, that having near video frame rate was more important. They all requested, however, that the higher spatial resolution versions of the sequences still be made available in case they should want to examine a still image.

Subjects were also shown a colour doppler sequence and asked to rate its usefulness, when displayed on a monochrome screen. The unanimous opinion was that it was not very useful. As mentioned in section 2.7.2, the pixels of the echocardiographic images can fit into six bits. To handle the colour, two bits of the byte can be used to select the colour, 00 for grayscale (i.e. no doppler image), 01 for red (towards the transducer), 10 for blue (away from the transducer). So for future versions of the workstation, the sequence would be digitized in 24

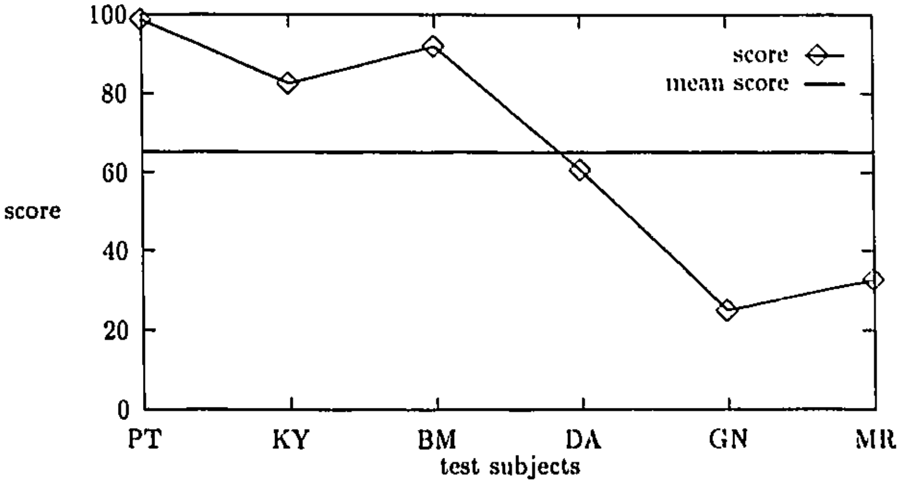


Figure 6.2: Plot of each test users' score in the order that they were tested. This shows roughly how the usability of the system improved as user errors initiated changes to the system, before the next user was tested. The score is a measure of difficultly, obtained by multiplying the average rating for the 14 tasks by the average number of seconds it took the user to complete each task.

bit colour, and then quantized down to 8 bits, possibly with a colour map for storage and display.

6.3 Clinical Trial

In order to meet our last objective, that is to assess the performance of the workstation in CCU, we simply put the workstation in CCU for one week[2], and observed its use.

6.3.1 Method

The workstation was placed in a storage room, around the corner from CCU, opposite the meeting room where the "hand-over" rounds are held every morning. Someone was on duty all week to:

1. Keep track of patients being admitted to and leaving the CCU.

---

[2] first week of February, 1993

2. Entering all examinations for all patients in CCU, performed in the past two years, into the workstation. For angiograms, this involves transferring the film to S-VHS video, and then digitizing the relevant frames. For echocardiograms, the sequences are already on S-VHS video tape.
3. Keep track of when patients are sent for an echocardiogram or an angiogram, and to enter the resulting examination into the workstation when it becomes available.
4. Observe the cardiologists when they consult examinations through the workstation. Record and discuss any problems they had.
5. Collect questionnaires from the cardiologist after they have used the workstation.

6.3.2 Results

During that week, 18 patients went through CCU, staying as short as a few hours, before being transferred to another ward, or as long as the whole week. In total, four angiograms and five echocardiograms were entered into the workstation. The workstation was used by the cardiologists for two principle reasons:

Consultation: During the morning rounds, the senior cardiologist, residents, and surgeons discuss the case of each patient, and decide on what, if any, treatment is to be carried out. As a result, each morning towards the end of CCU rounds (about 11:00 am), about twelve people would squeeze into the storage room and with one of cardiologists at the workstation, would proceed to backup previously made decisions with evidence presented by the workstation.

Teaching: From time to time (during the night, or a quiet moment during the day), the residents would assemble in front of the workstation, and the senior resident would use the image sequences to annotate their teaching about a particular pathology, or about the imaging modalities themselves.

The fact that the workstation was used only two or three times a day, and that it was used by groups, rather than individuals, meant that only two questionnaires ever got filled out. Hence all conclusions are based on observations and interviews with the cardiologist during and after the trial.

While having the workstation in CCU did appear to facilitate their work, the cardiologist did have some criticisms. Foremost was the poor quality of the angiographic studies, caused by a faulty camera on the film to video transfer projector. This fault was detected only once the trial was underway, and could not be fixed in time. As a result, the contrast of the angiograms on the workstation was too severe and detail was lost in the shadows and highlights.

In general, it was felt that the workstation was more useful for echocardiograms than for angiograms. This was due in part to the poor image quality of the angiograms, but also due to the relative time savings afforded by having the echocardiograms in the CCU. Having the angiograms in the CCU saved the cardiologist only a couple of minutes, as they only have to walk down the hall to view the original ciné films. For the echocardiogram, however, the time savings where anywhere from ten minutes to over an hour.

The annotations features of the workstation were used only once, as users found it was easier to draw with a pencil and paper (usually on the back of a questionnaire!), and to point using a pencil on the screen. This leads to the conclusion that perhaps the only useful annotation function would involved a light pen or a touch screen, that would allow the user to doodle right on top of the image, similar to some of the newer pen based portable computer systems.

6.4 Summary

The workstation presented in Chapter 5 was evaluated in order to complete the user centred design process. The formative evaluation uncovered eleven usability faults, that were consequently fix. Note that these errors were missed when evaluating the design against Smith and Mosier's set of user interface design guidelines[51].

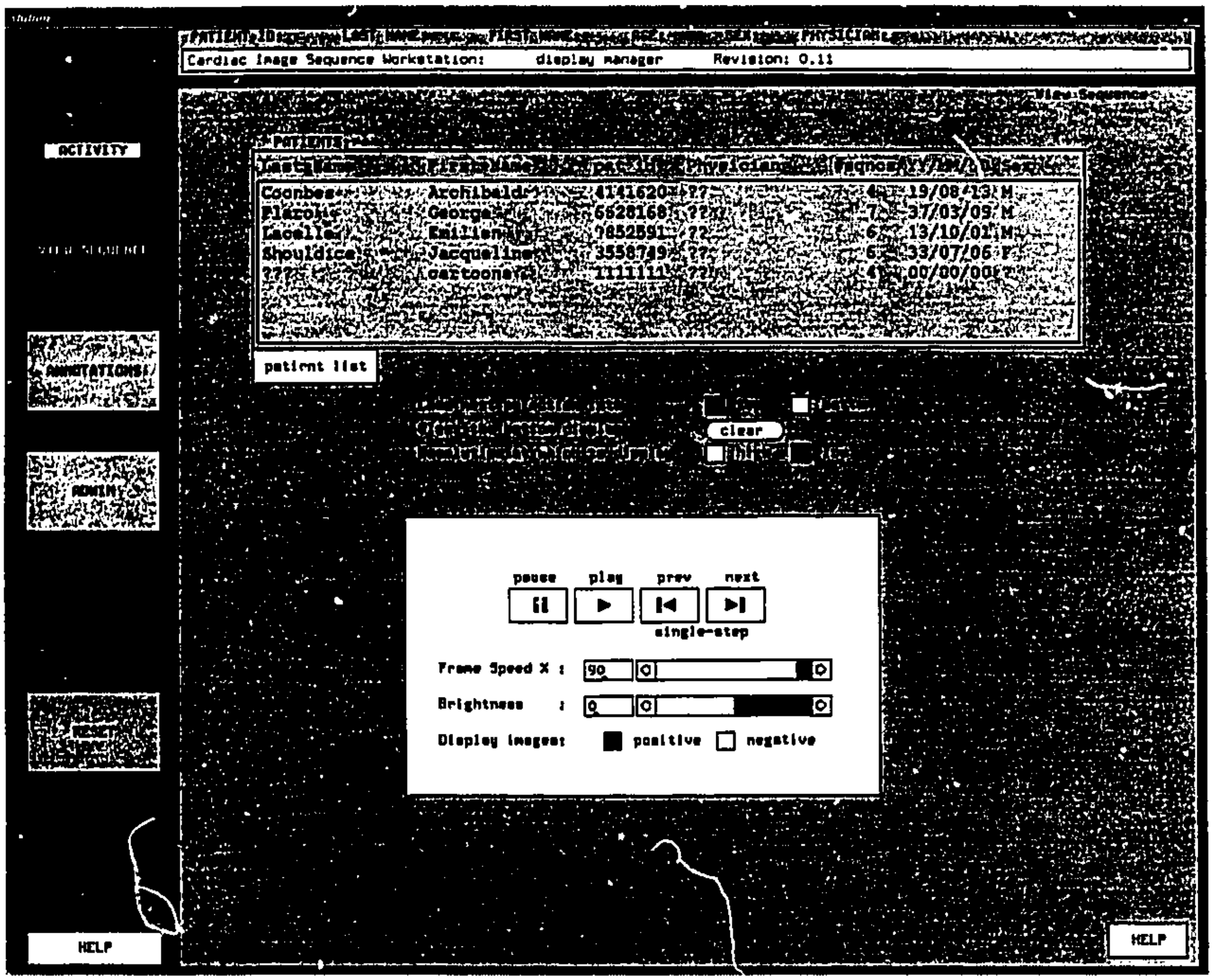
Quantitative evaluation of the user interface by cardiologists determined that the system was between "easy" and "very easy" to use. All those who tried out the system said they would definitely use it, if it were installed in the CCU.

For secondary viewing, it was determined that near video framing rate at the expense of spatial resolution was preferable to video spatial resolution at the expense of frame rate. Unfortunately, while suitable for most of the imaging modalities, the grayscale storage and display was not appropriate for colour doppler image sequences.

A one week trial in CCU found that the workstation was convenient for consulting echocardiographic studies during rounds. The workstation was also useful for teaching of the residents.

Error	Correction
'Patient Folders' label does not indicate the action performed on the screen.	Changed to read 'VIEW SEQUENCE'.
Listed number of sequences in patient's folder often wrong.	Corrected the bug.
Loading a sequence into the bottom display does not load its annotations, this confuses users.	Added a popup message to warn user of this, and how to load the annotations as well.
To change sequence list to list of patients, user clicked on first line, confusing if this line is scrolled off the top of the list.	Added a button under list which the user presses to get back the list of patients.
Triple clicking on a sequence will loaded it twice, causing user to wait unnecessarily.	Added an exception handler to trap this - loaded sequence cannot be loaded again.
Pushing a 'Create Object' button n times produced n objects.	Added an exception handler to ignore previous button presses.
Protocol for creating the various objects was not obvious.	Added step by step instructions in the message line for the user to follow.
Preparing a display for annotation not obvious.	Added instructions to the blank annotation window.
Interface does not give any indication as to how one might move an annotation object.	Added a 'MOVE' button which popups up a window containing instruction.
Users do not always notice that the message line has changed.	Background colour of message line changes whenever the message changes.
Users do not always notice, nor understand the watch cursor displayed when system is busy.	A message is displayed asking them to please wait.

Table 6.3: List of usability errors uncovered by the six cardiologists, and the resulting corrections to the system.



Cardiac Image Sequence Workstation: display manager Revision: 0.11

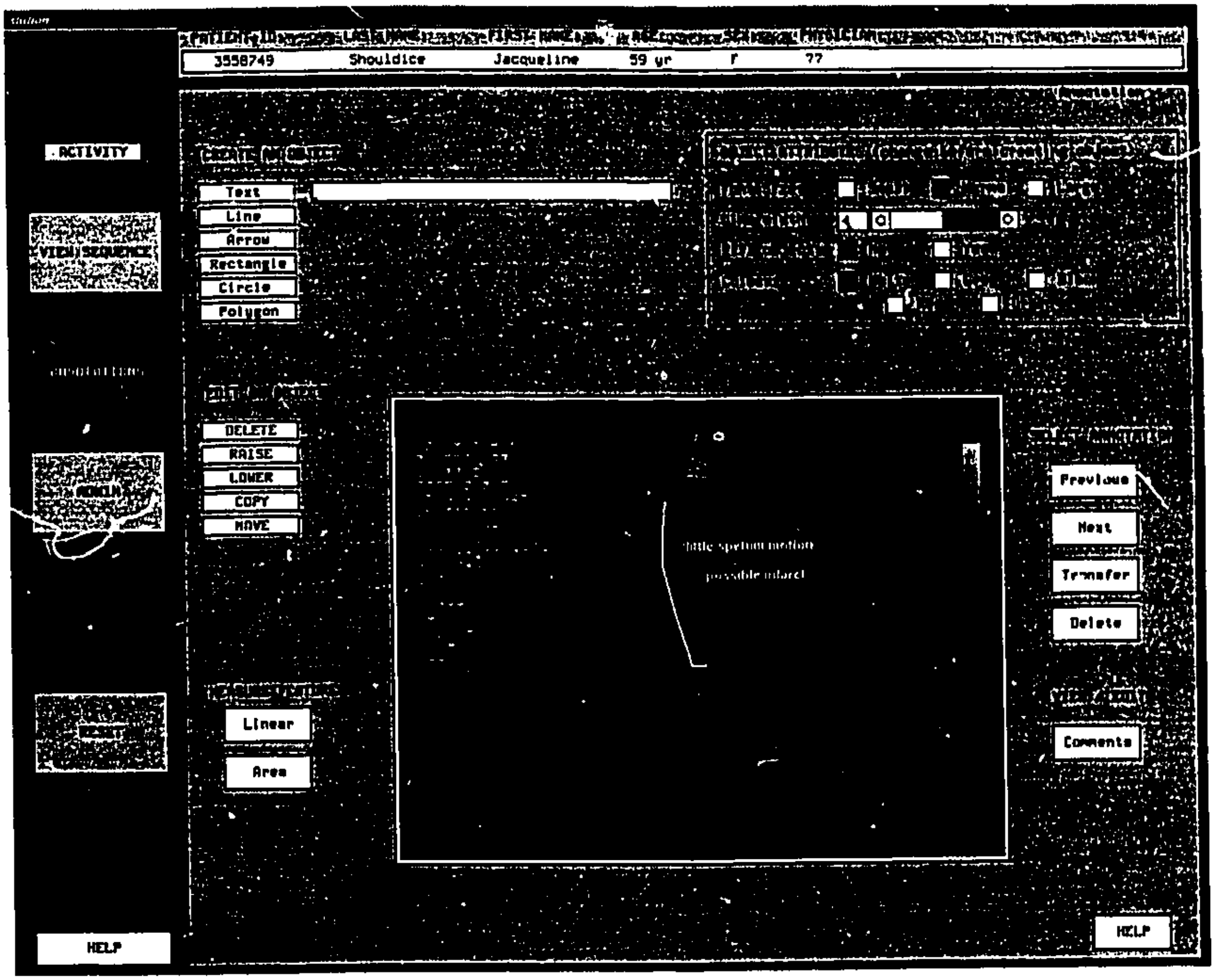
Last Name	First Name	pat.ID	Physician	Diagnos	YY/MM/DD	Sex
Coombes	Archibald	4141620	??	4	19/08/13	M
[illegible]	George	6628168	??	7	37/03/09	M
Lacelle	Emilien	7852591	??	6	13/10/01	M
Shouldice	Jacqueline	3558749	??	6	33/07/06	F
???	cartoonest	1111111	??	4	00/00/00	?

Frame Speed X : 90

Brightness : 0

Display images: ■ positive ☐ negative

Figure 6.3: The View Sequence screen, through which the user can load and control image sequences. Note that both administrative screens have been merged into one, called Administration, and there is now a Reset button, which brings the entire workstation to its initial state.



3558749 Shouldice Jacqueline 59 yr f 77

little septum motion

possible infarct

Figure 6.4: The Annotation screen, where the user can transfer a still from the display screen, and perform annotation.

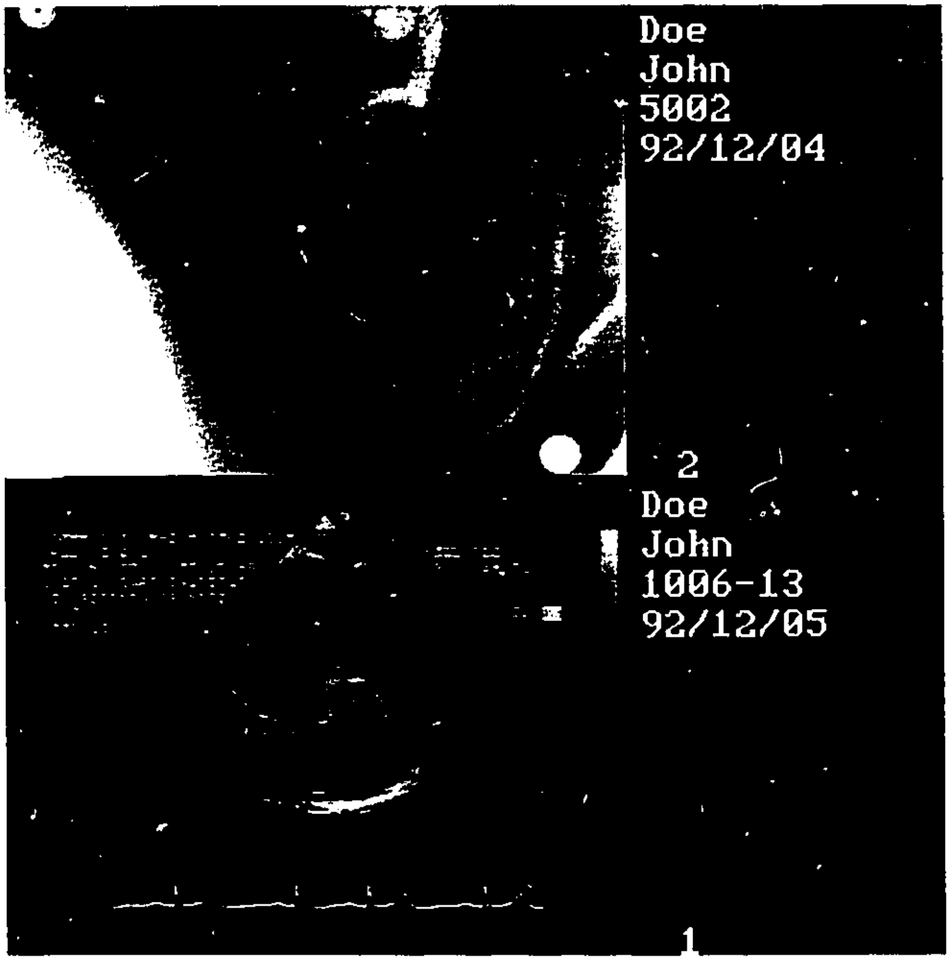


Figure 6.5: The top and bottom of the image display screen. Textual information is placed next to each image sequence in order to orient the user.

Chapter 7

Conclusions

For this thesis, we have developed and evaluated a computer workstation for cardiac image sequences.

From our review of cardiac imaging modalities and of computer video systems, we found that a proprietary digital video solution was needed to handle cardiac image sequences. Our solution, which consisted of "off the shelf" hardware and custom software, met the requirements of the cardiologists, specifically in terms of video resolution and speed.

Specifically, we built a system on an PC running UNIX. A video digitizing card, integrated into the system, captures the cardiac examinations from S-VHS video tape, for storage on disk. The user interface allows the cardiologist to select a view by indicating the patient's name, the examination date, and the view name, to display the sequence at a resolution of 320 by 240, at 20 frames per second.

By using a user centred design methodology, we successfully created a workstation that was both easy to use and useful for cardiologists. Summative evaluation of the user interface identified it as being between "easy" and "very easy" to use. This was further reinforced by unsolicited comments from the cardiologists who used the workstation when it was placed in the Coronary Care Unit (CCU). During the week that the workstation was available in the CCU, cardiologist found that it saved time, particularly when dealing with echocardiograms. Also, the residents found it to be a convenient tool with which to annotation their teaching.

7.1 Suggestions for further research

From the evaluation of our workstation, we conclude that further research is warranted in the following areas:

1. The annotation functions, while easy to use, were not as easy to use as a pencil and paper. A more manageable set of annotation tools is needed, perhaps using a direct form of input, like a light pen, which would allow the cardiologist to sketch directly onto the displayed images.
2. Digitizing from the VCR requires too much manpower. A more automated procedure is needed. Ideally, the examinations would be digitized directly from source, as the examinations are being performed, by a networked workstation attached to the scanner.
3. When the technology becomes available, better resolution display should be used, specifically concerning the frame rate of the video, and colour for 2-D colour Doppler echocardiography.
4. When Picture Archiving and Communication Systems (PACS) are introduced into the cardiologist's environment, many more forms of information will be available to the cardiologists. When that time comes, the workstation must be capable of integrating all this information in a usable fashion.
5. Having the echocardiograms and angiograms in digital format facilitates quantitative analysis of the heart function. Hence it is logical that a cardiologist's workstation should provide functions for measuring stenosis from angiograms or ejection fraction from echocardiograms.

Bibliography

[1] Danielle Argiro and John Rasure, "An X windows based application programming system", In *XHibition '92*, pages 151–158, San Jose, CA, June 1992.

[2] K. Blair Benson and Jerry Whitaker, *Television and Audio Handbook*, McGraw-Hill, New York, 1990.

[3] Paul Booth, *An Introduction to Human-Computer Interaction*, Lawrence Erlbaum Associates, London, 1989.

[4] Todd Brunhoff, "Video windows", *Systems International*, pages 44–48, March 1990.

[5] Stuart K. Card, Jock D. Mackinlay, and George G. Robertson, "A morphological analysis of the design space of input devices", *ACM Transactions on Information Systems*, 9(2):99–122, April 1991.

[6] J.H. Chien, S.T. Fu, E Horowitz, and C. Rouff, "RPP: A system for prototyping user interfaces", In *CHI'91 Conf. proc.*, page 419, New Orleans, April 1991.

[7] Microsoft Corporation, *Microsoft Windows Guide to Programming*, Microsoft Press, Redmond, WA, 1990.

[8] Subrata Das and Arthur Nadas, "The power of speech", *BYTE*, pages 151–160, April 1992.

[9] B. Diebold, J.L. Cloez, P. Péronneau, C. Scheublé, and T. Touche, *Échocardiographie Doppler: Bases Théoriques et Applications Cliniques*, Medecine et Sciences Internationales, Paris, 1986.

[10] John W. Donovan, "Intel/IBM's audio-video kernel", *BYTE*, pages 177–186, December 1991.

[11] Richard E. Fairley, *Software Engineering Concepts*, McGraw-Hill, New York, 1985.

[12] James D. Foley, Andries van Dam, Steven K. Feiner, and John F. Hughes, *Computer Graphics, Principles and Practice*, Addison–Wesley Publishing Company, Reading, MA, 2^{nd} edition, 1990.

[13] Rick H. Freeman, Cynthia S. Neubecker, Scott A. Keller, and James E. Steinke, "Operator interface design for a pacs primary diagnosis workstation", In *SPIE Vol. 1234 Medical Imaging IV: PACS System Design and Evaluation*, pages 430-441, Newport Beach, February 1990.

[14] James Gettys, Philip L. Karlon, and Scott McGregor, "The X window system, version 11", *Software-Practice and Experience*, 20(S2):35-67, October 1990.

[15] William Grossman, editor, *Cardiac Catheterization and Angiography*, LEA & Febiger, Philadelphia, 3^{rd} edition, 1986.

[16] Nancy M. Hickey, John G. Robertson, and Marj Coristine, "Integrated radiologic information system: A radiology multimedia communication system", *Journal of Thoracic Imaging*, 5(1):77–84, 1990.

[17] Deborah Hix, "Generations of user-inteface management systems", *IEEE Software*, pages 77–87, September 1990.

[18] R Hunter, P Kaijser, and F Nielson, "ODA: a document architecture for open systems", *Computer Communications*, 12(2):69–79, 1989.

[19] Andrew F. Inglis, *Video Engineering*, McGraw-Hill, Inc., New York, 1993.

[20] Masatsugu Iwase and Iwao Sotobata, *Clinical Echocardiography*, Kluwer Academic Publications, London, 3^{rd} edition, 1989.

[21] R. Jeffries, J.R. Miller, C. Wharten, and K.M. Uyeda, "User interface evaluation in the real world: a comparison of four techniques", In *CHI'91 Conf. proc.*, page 119, New Orleans, April 1991.

[22] Ahmed Karmouch, Luis Orozco-Barbosa, Nicolas D. Georganas, and Morris Goldberg, "A multimedia medical communications system", *IEEE Journal on Selected Areas in Communications*, 8(3):325–339, 1990.

[23] Joseph J. Lazzaro, "Even as we speak", *BYTE*, pages 165–172, April 1992.

[24] Mark A. Linton, John M. Vlissides, and Paul R. Calder, "Composing user interfaces with InterViews", *IEEE Computer*, 22(2):8–22, February 1999.

[25] Arch C. Luther, *Digital Video in the PC Environment*, McGraw-Hill, New York, 2^{nd} edition, 1991.

[26] Joel McCormack, Paul Asente, and Ralph R. Swick, *X Toolkit Intrinsics - C Language Interface*, MIT, X version 11, release 4 edition, December 1989.

[27] James R. Miller and Robin Jeffries, "Usability evaluation: Science of trade-offs", *IEEE Software*, page 97, September 1992.

[28] Frederick C. Mish, editor, *Webster's Ninth New Collegiate Dictionary*, Thomas Allen & Son LTD, Markham, Ontario, 1989.

[29] R.M. Mulligan, M.W. Altom, and D.W. Simkin, "User interface design in the trenches: Some tips on shooting from the hip", In *CHI'91 Conf. proc.*, page 232, New Orleans, April 1991.

[30] Nova MicroSonics, "Ultrasound information management", Technical report, Nova MicroSonics, New Jersey, 1991.

[31] Open Software Foundation, Cambridge, MA, *OSF/Motif Programmer's Guide*, revision 1.1 edition, August 1991.

[32] Open Software Foundation, Cambridge, MA, *OSF/Motif Style Guide*, revision 1.1 edition, August 1991.

[33] Andrew J. Palay, "Toward an "Operating System" for user interface components", In *Multimedia Interface Design*, New York, 1992.

[34] Randy Pausch, Matthew Conway, and Robert DeLine, "Lessons learned from SUIT, the Simple User Interface Toolkit", *ACM Transactions on Information Systems*, 10(4):320–344, October 1992.

[35] Randy Pausch and J.H. Leatherby, "A study comparing mouse-only input vs. mouse-plus-voice input for a graphical editor", In *Proceedings of the AVOIS '90 Voice I/O Systems Applications Conference*, pages 227–231, September 1990.

[36] Chris D. Peterson, *Athena Widget Set - C Language Interface*, MIT, X version 11, release 5 edition, 1991.

[37] Kathleen Potosnak, "Recipe for a usability test", *IEEE Software*, pages 83–84, November 1988.

[38] Kathleen Potosnak, "Evaluating how your user interfaces are used", *IEEE Software*, pages 112–113, January 1989.

[39] Kathleen Potosnak, "Mental models: Helping users understand software", *IEEE Software*, pages 85–88, September 1989.

[40] Kathleen Potosnak, "When a usability test is not the answer", *IEEE Software*, pages 105–106, July 1989.

[41] John Rasure and Mark Young, "An open environment for image processing and software development", In *1992 SPIE/IS&T Symposium on Electronic Imaging, SPIE Proceedings Vol 1659*, February 1992.

[42] Osman Ratib and H.K. Huang, "Low cost desktop image analysis workstation with enhanced interactive user interface", In *SPIE Vol. 1091, Medical Imaging III: Image Capture and Display*, pages 184–190, 1989.

[43] John M. Reid, "Doppler ultrasound", *IEEE Engineering in Medicine and Biology*, pages 14–17, December 1987.

[44] John F. Rice, "Display color coding: 10 rules of thumb", *IEEE Software*, pages 86–88, January 1991.

[45] Dominic Richens, Nicola Rougon, Isabelle Bloch, and Elie Mousseaux, "Segmentation by deformable contours of MRI sequences of the left ventricle for quantitative analysis", In *Fourth IEE International Conference on Image Processing and its Applications*, pages 393–396, Maastrich, 1992.

[46] John G. Robertson, "Cardiologist's image workstation: Preliminary report", Technical report, MCRLab, University of Ottawa, October 1991.

[47] Marc J. Rochkind, *Advanced UNIX Programming*, Prentice-Hall, Englewood Cliffs, 1985.

[48] Etienne Roger, "Image organization and navigation strategies for a radiological workstation", Thesis for masters of applied science, Electrical Engineering, University of Ottawa, November 1988.

[49] Mikel A. Rothenburg and Charles F. Chapman, *Dictionary of Medical Terms*, Barron's Educational Series Inc., New York, 2^{rd} edition, 1989.

[50] Robert W. Scheifler and Jim Gettys, "The X window system", *Software-Practice and Experience*, 20(S2):5–34, October 1990.

[51] Sidney Smith and Jane Mosier, "Guidelines for designing user interface software", Technical Report ADA177198, Mitre Corporation, Technical information service in Alexandria, VA, 1986.

[52] Ollie C. Woodard Sr. and Tom Long, "Display technologies", *BYTE*, pages 159–168, July 1992.

[53] The Khoros Group, *Volumes 1, 2 and 3 of the Khoros 1.0 Manual*, January 1992.

[54] Tom Thompson, "Mac programming revealed", *BYTE*, pages 279–287, April 1992.

[55] Sanjay Kumar Udani, "Experimental evaluation of a video capture board for networked workstations", Thesis for masters of science in engineering, Electrical Engineering, University of Pennsylvania, May 1992.

[56] John M. Vlissides and Mark A. Linton, "Unidraw: A framework for building domain-specific graphical editors", *ACM Transactions on Information Systems*, 8(3):237–268, July 1990.

[57] Peter Wayner, "Inside Quicktime", *BYTE*, pages 189–196, December 1991.

[58] Nick West, "Multimedia masters", *MacWorld*, pages 114–117, March 1993.

[59] Tom Yager, "Information's human dimension", *BYTE*, pages 153–160, December 1991.

[60] Tom Yager, "Edit video at your desk", *BYTE*, pages 260–261, May 1992.

[61] Tom Yager, "Video goes digital with Fluency", *BYTE*, pages 267–268, April 1992.

Appendix A

Usability testing scripts

A.1 Mouse and widget training

Read the following script to the user, instructions to the tester are in this type face.

Introduction

Before we being testing the Cardiologist Workstation, I want to first ensure that you have some basic computer skills that you will need to operate the workstation. The mouse on this computer is sensitive and I want to be sure that you are comfortable with before we test the workstation.

1. Handedness.

Are you right or left handed?

Move the mouse to the preferred side of the workstation.

2. Holding the mouse.

To use the mouse, it is best to rest the weight of your hand on the ball of your palm, lightly gripping the mouse by its sides with your thumb on one side and your ring finger on the other.

3. Moving the mouse.

Start by getting used to the way the mouse moves. Think of the human and mouse as a feedback system that requires tuning before it will work efficiently.

- move the mouse in a clockwise motion, passing the cursor on the screen through the four corners of the centre window. *Indicate to the user what a window is if they don't know.*

- notice that if you lift the mouse off of the table, the cursor on the screen stays put, allowing you to position the mouse on the table so that it is comfortable. *Have them try this out.*
- now pass the mouse cursor through the numbered boxes in the order that they are numbered. Notice that the box highlights when the cursor is inside of it. *Have the user do this as many times as it takes for them to become adjusted to the mouse.*

4. Clicking

Now get used to "clicking" Those little numbered squares can be thought of as "buttons" but instead of pushing it directly with you finger, position the mouse cursor inside the "button" and press the left mouse button with your index finger.

- move the mouse cursor into the button labeled "1" and press down on the mouse button. Notice that the number above is now "2" This tells you which button you can now press.
- Notice that the action occurs when the release the mouse button, not when you press it down. This is so that you can change your mind after you have pressed down the mouse button. Try this with button "2" holding down the button. Notice that it has gone black, to indicate what action will occur when you release the button. Move the mouse out of the button and then release it.
- now press the buttons in order. *If the user accidentally presses out of order, tell them how to get rid of the popup.*

5. Popups

Press a button out of order. A small popup window will tell you what you have done wrong. This is how the workstation informs you about errors, and also sometimes to ask you a "Yes" or "No" type question. When you get one of these, read the message, and, before you do anything else with the system, push (one of) the button(s) at the top right hand corner of the window. Press the "continue" button to get rid of the window.

6. Slider for integer selection.

In order to enter a numeric parameter into the program for something, you can, if you want to, type is into the little box from the keyboard. But the keyboard is a little awkward to use, so you can use the two arrow buttons to the right to increase or decrease the value.

- Use the arrow buttons to set the integer to "4" Now you can press the "4" button without going through 1-3.

You can also set it by positioning the mouse cursor in the gray area between the arrow buttons.

- For instance, set the number to "8" by positioning the cursor 80% of the way long and pressing the mouse button.

Ask the user if they want to practice any of the activities some more. When they are satisfied, go onto the next section.

7. Browsing a list.

Click on the "Browsing" button on the left.

We will now practice browsing a list. What you see before you is a box containing a list of the continents. Next to each continent name is its area and population. If you click twice on a continent name, the list will be replaces by a list of countries on that continent. Next to each country's name is its population and the number of hospital beds.

- I want you to tell me what is the number of hospital beds in Spain. The user should select Europe, but they won't see Spain. The list is too long for the box, and Spain is one of the countries off the list. To the right of the box is a slider and two arrow buttons which will display items in that direction. Whenever you see such a slider on the right of a list box, it is because there is more information than will fit in the box. Push the appropriate button until Spain is visible. You now know the population and number of hospital beds in Spain.
- Now I want you to tell me the number of hospital beds in Canada. *See if the subject can figure out by themselves how to get back to the list of continents. If they can't within one minute, show them how.*

Conclusion

So, do you know feel comfortable with the mouse?

If not, ask the user what they would like to play with, and let them.

A.2 Usability test script and questionnaire

Introduction

Bring up the Cardiologist Workstation window and close the mouse training window. Make sure that all the subject can see is the Workstation user interface.

What you see before is a workstation that has been designed to aid the cardiologist who work in the coronary care unit. We are hypothesizing that the cardiologists that are responsible for overseeing the patient's treatment would refer to angiographic and echo exams more often if these studies were easier to get at. at.

To test this hypothesis, this workstation was built. It can hold excerpts from the angiographic and cardiac ultrasound examinations of the patients in the coronary card unit. These excerpts, which we call image sequences, are stored digitally in a database on the computer, so that by selecting a patient, the exam number or date, and the view, the physician can, in a matter of seconds, view a patients examination.

Questionnaire

Before we being with your evaluation of the system, a few questions.

```
1.   Name:   -----------------------

2.   What is your specialty?   -----------------------
     Any sub-specialization? (what) -------------------

3.   What level of education do you have? ---------------------

4.   Describe briefly what it is you do.
     ---------------------------------------------------------
     ---------------------------------------------------------
     ---------------------------------------------------------
     ---------------------------------------------------------

5.   (for cardiologists) For what percentage of patients do you
     request:
               an angiogram?      -----
               an echocardiogram? -----

6.   (for cardiologists) How often (roughly) do you look at the:
               angiogram?      -----
          What is your most common reason for doing so? -----
               echocardiogram? -----
```

```
          What is your most common reason for doing so? _____

7.    (for cardiologists) If these exams were as easy to get at
      as the information in the patient's chart, would you look at
      it more often? _____  All the time? _____

8.    On a scale of one to five, rate your experience with
      computers.
                1    2    3    4    5

9.    On a scale of one to five, rate your experience with mouse
      and windows type of computer systems.

                1    2    3    4    5
```

Subject's instructions

Pretend that you and this workstation are in the coronary care unit. You have at your disposal all of the information that you normally have, such as the patients' charts.

I'm going to ask you to perform a series of tasks with an aim to finding out how intuitive and obvious this system is, and to help me identify where the system needs improving. The tasks I am going to give you are all possible, though sometime indirectly; you may have to perform some intermediate steps so that you may complete the task I give you.

When you have successfully completed a task, I'll ask you to rate how difficult you found the task, on a scale of one to five:

- one for very easy - no problem
- two for easy - took a little thinking
- three for okay
- four for hard enough that it needs changing.
- five for so hard I'd rather use a VCR or a projector.

It is important that you remember that what we are testing is the system, not you; if you can't figure something out, it's not your fault, the system needs changing.

Also, to help me in diagnosing problems, be as vocal as you can. Tell me what is going through you head as you go about performing the tasks.

Most important of all, be adventurous and experiment. Don't be afraid you might break something. If you do, somehow, manage to break the system, you'll simply be revealing to me something that needs fixing.

Before we start, have a good look at what is on both screens. Let me know when you are finished, and we'll being.

The Tasks

I. Image navigation, selection, and loading.

You are looking over the chart for C. Lanthier. The report from radiology states that there is stenosis of the circumflex artery, but you want to be able to see it for yourself.

- Q. Can you tell me how many image sequences there are in the system for C. Lanthier? _____

1. Bring up the list of the sequences stored in the system for this patient.

   - How easy? 1 2 3 4 5

2. Bring up on the display an image sequence that will show the reported stenosis. (Which one? _______)

   - How easy? 1 2 3 4 5

II. Sequence manipulation and frame selection

3. When first displayed, the sequences come up "freeze-frame", but you want to see the heart in motion.

   - How easy? 1 2 3 4 5

4. Slow down the motion to about half of what it is now.

   - How easy? 1 2 3 4 5

5. You want to freeze the image on the end-systolic frame.

   - How easy? 1 2 3 4 5

6. Adjust the brightness of the image so you can best see the image. Maybe even make it negative.

- How easy? 1 2 3 4 5

III. Loading a second sequence

7. In order to get a better impression of the patient's coronary arteries, load the ______ view of the heart into the bottom half of the image display, and look at the two views simultaneously in motion.

- How easy? 1 2 3 4 5

IV. Annotating

8. Display the frame that you feel shows the stenosis in both views. For future reference, you want to draw an annotation to indicate the stenosis. Prepare this frame for annotation.

- How easy? 1 2 3 4 5

9. Specifically, you want to put a red circle around the stenotic sections of the arteries.

- How easy? 1 2 3 4 5

10. There is a facility which allows you to enter, into the computer, your comments describing the annotation in detail. You enter these comments by typing it in at the keyboard.

- How easy? 1 2 3 4 5

V. More navigation and annotation viewing

11. Okay, you are finished with this patient. You are curious to see an echocardiogram of F. Pregent. Display a list of the sequences (examinations, views) in the system for this patient.

- How easy? 1 2 3 4 5

12. One of these views has some annotations, made by another cardiologist. You want to see these annotations.

- How easy? 1 2 3 4 5

13. There is more than one annotation, look at the other two.

- How easy? 1 2 3 4 5

14. One of the annotations has the labels for the right ventricle and right atrium mixed up. Fix this.

- How easy? 1 2 3 4 5

Post-experiment questionnaire

1. Concerning the list of patients, the list of image sequences, is there any information that you feel is missing from the list?
   Yes [ ] What? ___________________________________
   No [ ]

2. The order in which the patients appear in the list is in the order they were entered into the computer. Is there a better order?
   Yes [ ] What? ___________________________________
   No [ ]

3. How about for the list of image sequences?
   Yes [ ] What? ___________________________________
   No [ ]

4. When you select an image sequence for display, do you find that the amount of time the computer takes to load the sequence and display acceptable?
   Yes [ ]
   No [ ]

5. Would you ever want to look at the film loop running in reverse?
Yes ☐
No ☐

6. The frame rate for the images is about 20 frames/second when only one sequence is loaded, and about 10 frames/second when two are loaded. Was the motion of the images fast enough?
Yes ☐ What? ____________________________________
No ☐

7. Is the way in which the sequence type and view are labeled acceptable?
Yes ☐
No ☐ Why? ____________________________________

8. What are the chances that you would use this system, if it were installed, as it is, in the Coronary Care Unit tomorrow.

- What chance? 1 2 3 4 5
definitely maybe definitely not!

Image resolution / frame rate tradeoff

Due to limitations in technology (and money) there is a limit to the frame rate at which we can display the images. One solution to make the images smaller. The problem is that the images then have a lower resolution. I want to know which is more important, the image resolution or the frame rate.

Cardiac Angiogram

Go into John Doe's list of image sequences, and ask the subject to view each view, alternating first at high resolution, and then at low resolution. (tell them the patient has no pathology).

Which do you prefer? High ☐ Low ☐

Would you still like to be able to display the ____ version?
Yes ☐
No ☐

Remind the subject that the low res sequence can be slowed down.

Echocardiogram

Go into Wendy S.'s list of image sequences, and ask the subject to view each view, alternating first at high resolution, and then at low resolution.

`Which do you prefer?    High [ ]   Low [ ]`

```
Would you still like to be able to display the ____ version?
          Yes [ ]
          No  [ ]
```

Colour flow doppler

Go into Vladimir K.'s list of image sequences, and ask the subject to view the doppler study. As you can see, colour flow doppler scans are only rendered in black and white. How would you rate the usefulness of a black and white flow doppler?

```
   - How useful?   1    2    3    4    5
               useful   sometimes      useless
```

A.3 Raw results

	PT		KY		BM		DA		GN		MR	
	rating	time	rating	time	rating	time	rating	time	rating	time	rating	time
Q	-	30	-	45	-	2	-	2	-	5	-	2
1	2	30	1	10	2	10	1	2	1	5	1	2
2	2	30	1	10	1	15	1	12	1	10	1	30
3	1	5	1	10	2	15	1	4	1	15	1	5
4	2	20	2	2	1	25	1	45	1	10	1	15
5	1	40	1	20	1	10	2	50	1	10	1	20
6	1	10	1	15	1	10	2	45	1	10	1	10
7	2	10	1	20	1	20	2	55	1	40	2	50
8	3	30	1	50	2	15	1	30	1	15	1	40
9	4	190	2	80	4	30	2	50	3	75	1	45
10	1	10	2	15	2	60	1	45	1	15	1	15
11	3	65	4	50	2	150	4	140	1	20	1	20
12	2	50	4	110	2	295	1	75	1	55	2	95
13	2	15	2	40	2	10	1	20	1	15	1	20
14	4	155	3	90	3	75	1	20	1	30	2	110

Table A.1: The ratings given by the test subjects for each task, and the amount of time it took them to complete the task (in seconds).