2000

Digital multimedia development processes and optimizing techniques

David L. Thompson
The University of Montana

Digital Multimedia Development Processes and Optimizing Techniques

by

David L. Thompson

B. S. Montana Tech of The University of Montana, 1993

presented in partial fulfillment of the requirements

for the degree of

Master of Science

The University of Montana

December 4, 2000

Approved by:

____
Chairperson

____
Dean, Graduate School

12-11-00
____
Date

Thompson, David Lance, M.S., October 2000 Computer Science

Digital Multimedia Development Processes and Optimizing Techniques (107 pp.)

Director: Ray Ford and Jerry Esmay

This thesis explains how to minimize a subset of all the risks involved with multimedia development. Changes from original product specifications can increase costs more dramatically than expected. However, with a development method rooted in software engineering design, project teams can minimize these cost increases. Using a digital media library to store separate archival quality media, creating prototypes using the WinWin Spiral model, and completing the project with a optimization step for each delivery system will reduce final production costs. Not only do the major stakeholders receive the completed project but also receive the digital media library useful in reducing costs in future projects.

Contents

List of Figures

List of Tables

Chapter 1

Overview

Selling a product, educating students, or entertaining large groups of people have been challenges for salesmen, educators, and directors. New techniques for capturing and holding the attention of an audience have been created over years and years of experience. Audiences have become accustomed to flashy Hollywood style deliveries with fancy video and audio that entertain and keep their attention focused. This is where digital multimedia can help. Digital multimedia integrates information from two or more media types under the control of a digital device such as a computer. Separate media for the presentation can be gathered and integrated into a flashy, audience grabbing form, arranged to present Hollywood effects that can be displayed on demand.

Producing digital multimedia titles can be a risky business. There is a wide range of risks that the developer must consider when producing an outstanding multimedia title: aesthetics, human interaction, responsiveness, delivery systems, and machine dependencies. Large products usually mean that more than one developer will be involved, thus team management, project organization, and product specifications become major risks. Each of these risks must be addressed for development to be successful.

Multimedia Development Risks

Aesthetics

Consumers have become experts in judging entertainment quality. From experience with television, theater, radio, magazines and studio movies, audiences can be assumed to have considerable experience viewing media of certain types, leading to relatively high quality expectations. They will be able to tell very quickly if the quality (i.e. versus the content) of a multimedia presentation is good or bad. If the audio fluctuates, animations begin to flicker, or color fidelity is lacking, the audience usually assumes that the content is not top quality and quickly loses interest.

User Interface HCI

All multimedia products do not have to be designed to accommodate everyone, but they do need to be designed for the needs and capabilities for the audience which they are intended. For an audience accustomed to viewing television and using the television remote control, a multimedia product that involves typing long archaic commands is inappropriate. Computer scientists have studied the techniques of making computer software user friendly and usable. This field of study is called Human Computer Interaction (HCI) and can be used to help the multimedia developer create effective products.

Responsiveness

In this world of media on demand through radio and television, users have come to expect that they do not have to wait for a response. The multimedia title that does not provide user feedback immediately will result with audience loss. Even if the response is to provide the user with information that there will be a time delay, the producer will have a better chance of retaining the audience.

The World Wide Web (WWW) has created users which have a small amount of patience when images are being downloaded, but they still will not wait for excessively large amounts of time. A video clip that requires over an hour of downloading is not acceptable by most users standards.

Delivery system requirements

Networks and CD-ROMs are two separate ways to deliver multimedia information to an audience. Each delivery system that a producer can use to reach the audience has its limitations. A standard network available to most computer users is the WWW. For most of the audience that could access information in this manner, the speed is extremely limited. Using a 33,600-baud modem, a single image of 1 megabyte would require at least four minutes to download before it could be viewed. The producer must know before hand what he can expect performance wise for each media used with each delivery system he intends to use.

Machine dependencies

On software boxes all over the country, a sticker exists that explains what the minimum requirements are for the software to perform properly. As the multimedia producer adds multimedia to a project, the complexity increases which in turn increases the amount of horsepower a computer will need in order to "play" the project adequately. The horsepower of a computer is usually defined in terms of processor power, storage space and speed, RAM, and software components installed. Compression techniques usually rely on how fast is the processor; non-compressed imagery relies on how fast the CD-ROM can read the data; where as the number of objects that can be viewed at the same time relies on how much physical RAM exists in the computer. Different types of hardware may have different requirements due to the Operating System. Things such as file system differences and the unavailability of software development APIs can limit the target audience. The developer must take into account all of these possibilities and the risk of losing potential audience members during production.

Re-ordering / Reorganization

As projects get larger and more complex, the developer may need to re-order the events or reorganize the information of the product. This can slow down the release of the final product, but without it, the project may become so fragmented and disorganized that it is impossible to work with.

Unknown final product

When producing a product, the audience that must first be satisfied is the major stakeholders. There is always a risk for what they expect to get and what is produced. The project may be scrapped or completely revamped causing all previous work to be lost. The cost of the project increases and tensions rise as the projected completion date is pushed back.

Team Management

The last risk that a producer may have to face is common in any work environment. If the project is large enough to involve more than one person, then team management must addressed. Usually on these large multimedia projects, experts from separate fields of study are assembled.

New team members may be added, as they are needed. Trying to keep everyone on task and productive is always a challenge.

Typical production example.

Multimedia artifacts are usually assembled by a team with expertise in graphics, story-telling, and content material. Media is collected and the presentation is put together in accordance with a general content "script", but without any plan to cover material integration, introduce special digital effects, or make the presentation coherent and exciting. As the multimedia package begins to take shape, reordering, reorganization, and revision invariably occur. New experts may be gathered to add new content or ideas to the project. Feedback from preview audiences may indicate need for additional revision. When the project finally begins to take shape, optimization steps may be needed to enhance the electronic forum, or additional work has to be done to capture new material or recapture material in different forms. These are all time consuming steps that can delay the release of the final artifact and dramatically increase production costs.

Computer scientists have been dealing with a similar development process for years in the creation of software systems. Determining what clients think they require, what they really want, and what they will accept, has led to the science of software engineering. Software engineers have developed processes which consist of ordered steps used to determine software requirements, to develop, test, and evolve software, and to establish the transition criteria for proceeding from one stage to the next. Some software engineering processes have begun to be applied to multimedia applications. One such example is the WinWin Spiral Model project described by Boehm, (1997). This project will be a beginning in which to create the necessary steps in order to evolve the multimedia software project.

In the typical multimedia development process, the developer is only concerned with producing a media artifact to fit an immediate need, which generally implies creating or capturing component media objects. Individual media objects can be collected or created on the computer. Either way, the object is stored digitally such that it can be integrated with other objects using a set of tools on a development workstation. Usually, each media object is stored as a single file in a file system. The hierarchy of the file system defines where the objects are located. Each type of media is different in terms of how its objects are collected, digitized and organized. Thus appropriate hardware, software, and processing techniques must be defined specific for each type to allow the developer to most efficiently complete a project.

Delivery systems also impose unique requirements on each media type in terms of the effective presentation of the objects. Objects can be captured and stored in a variety of forms, including those that optimize capture or those that optimize delivery on particular devices. In many cases, it is impossible to recreate the original, general object from an optimized form, so deciding what forms to capture and save objects in is a critical decision. In general, for maximum flexibility, an object must be saved in high enough "resolution" (quality) to be useful for all present and probable future projects. What defines high enough resolution? Many times the nature of the media limits itself, sometimes the type of media is limited by the current acquisition hardware, and other times the limits come from the amount of available storage space and processing time. The greatest risk in not obtaining objects at a high resolution is that the media may have to be acquired each time it is to be used. This slows down development and increases cost. The goal to decrease this risk is to create techniques that define archival quality media which lowers the cost of a project that shifts delivery systems, chooses to use multiple delivery systems, or undergoes major changes in concept or design.

As discussed earlier, audiences expect Hollywood style presentations that utilize high quality media. When the audience is made to wait, they lose interest quickly. Target presentations must maintain an appropriate pace without long time delays. The audience expects information to be available instantaneously. For interactive presentations, the "presentation manager" must react to user input immediately. Presentations must operate properly on a wide range of end-user computers, with sound and color fidelity, and "smooth" movies. For example, an attempt to reach a generic WWW user with video clips, which require special computer hardware for viewing, eliminates a large portion of the intended audience. These are all risks that can be reduced by optimizing each media element.

Optimization implies selecting and successfully implementing the best alternative from a set of alternatives. There are different aspects to optimization. One technique that must be evaluated for all media types is file compression; however, certain compression techniques are inappropriate when the playback system is limited. Other optimum techniques are more media type dependent, such as color reduction, vector description, sprite animation, flicker reduction, and bit reduction. Special constraints in the delivery system often determine what optimization techniques are appropriate for a particular media. One such example is that images with very few colors to be displayed on web pages are best converted to an indexed color space. By proposing pipelines that optimize each media element as it is used, the final stage of product development will entail a much smaller overall optimization step followed by final media distribution, but there is also less risk of over optimizing, producing images of too low quality.

The creation of multimedia presentations is a labor-intensive task, but the risk of development can be reduced if specific methods are applied. Within this thesis, I will discuss how the WinWin Spiral Model, an archival media library and the final spiral iteration of media optimization can reduce the multimedia producer's risks.

Chapter 2

Background

To reduce risks in a multimedia production, multimedia producers must systematically apply a development process in the collection, organization, and optimization of media. Often, media are obtained by the easiest means possible. Gathering media content at or close to optimal forms, then organizing it appropriately will save time and energy. However, because of the limitations of technology, the producer of a multimedia production often chooses a development process based more on a set of production tools that are available to produce the specified artifact than on a coherent development plan. The next two sections provide information regarding the different forms of digital media and software development methodologies, as background prior to the discussion of development plans to organize this type of material.

Digital Media

Media is information that is distributed to an audience via one of the five senses: sight, sound, smell, taste or touch. A simple definition of *digital multimedia* is an artifact that integrates two or more media, using a digital device such as a computer to provide control and coordination. With today's technologies, many different media can be included in a multimedia environment, but the output environment is limited to the devices of projectors, monitors, printers and speakers. Therefore, media associated with sight and sound are the media most applicable in today's multimedia presentations.

As illustrated in Table 1, digital media can be classified naturally into six categories: text, two dimensional graphics, pseudo three-dimensional graphics, animation, video, and audio. All but one of these categories expands into sub-categories, explained below.

Categories	Subcategories
Text	
Two Dimensional Graphics	Raster Graphics
	Vector Graphics
Pseudo Three Dimensional Graphics	Visually Depicted
	Geometrically defined. - Non-uniform rational b-spline surfaces - Polygonal surfaces
Animation	Full Frame
	Moving Sprites
Video	Non-interlaced
	Interlaced
Audio	Speech
	Music samples
	Musical Instrument Digital Interface (MIDI)
	Synthesis

Table 1. Digital Media Categories

Two-dimensional graphics are visual images formed by photographing, painting, and drawing. In the realm of digital multimedia, this category expands into separate sub-categories: raster graphics and vector graphics. Raster graphics images are represented by a rectangular grid of picture elements (pixels), whereas vector graphics are represented by geometrical models that describe points, lines, curves, surfaces, etc.

Pseudo three-dimensional graphics is the art of depicting three-dimensional images in a two-dimensional viewable fashion. This category divides into three sub-categories based on surface description: visual depiction, non-uniform rational B-spline descriptions (NURBS), and polygonal based descriptions. Visually depicted three dimensional images are typically captured with special devices, such as fish-eye camera lenses. When displayed in a two-dimensional form, such images give the audience the sense that they see an entire three-dimensional view of a scene or object. Real three-dimensional images can be geometrically constructed, containing geometric information and surface colormap information that allows a computer to construct a viewable image of the object from a range of view points. There are two different ways to represent the geometric information. NURBS surfaces use mathematical models called B-splines to represent a smooth gridded surface, whereas polygon based descriptions combine sets of connected polygons to represent surfaces.

Animation is the art of constructing a sequence of images which when displayed in sequence portrays movement of one or more objects in the images. Digitally, animation divides into two categories: full frame and moving sprites. Full frame animation is similar to video in that the entire image is re-depicted from frame to frame. Sprite techniques use the fact that much of the image, such as the background, stays constant between images, so that only small parts of the image (sprites) need to change between frames.

A video clip is a digital version of film or video. Video clips divide into two categories: non-interlaced and interlaced. Interlacing is the technique devised for television broadcast to provide smooth movement to the human eye. This technique was developed to compensate for mechanical deficiencies in early television sets. With early television sets, it was not possible to display the 24 frames per second required so that the human eye does not pick up a distinct "flicker." To compensate, interlaced display techniques were used to display the even lines of a monitor in one pass, then the odd lines in the next pass. The phosphor in the tube of the television stays illuminated long enough to give the appearance that the entire screen displays in each pass. Because of the better mechanics behind today's computer monitors, there is no reason to interlace video to create smooth motion display. The standard for television has been set for a long period of time, so interlaced video is still the mainstay in television broadcasting; however, non-interlaced video is used in other applications, such as the computer screens in multimedia kiosks.

Digital audio is the last category, representing the digital reproduction of audible sound. This category is broken into four categories: speech, music samples, musical instrument digital interface (MIDI) sequences, and synthesis. The first two sub-categories are both digitized samples created from some non-digital sound using a digital sound converter, e.g. a microphone and digital audio tape (DAT) recorder. MIDI combines audio samples together in sequences, similar to a music score, or coordinates the playback of digital sound with other digitally controlled events. Audio synthesis is the creation of audio waveforms using techniques such as frequency modulation, wavetable synthesis, and physical modeling that modify and filter in various ways to produce different tones.

Within the class of multimedia lies a particular form of interactive, non-sequential information organization known as hypermedia. The essence of hypermedia is that chunks of information from one or more different media forms are linked together in a manner that permits a user to navigate a range of different paths through the information, yet still have media of different forms presented in a coordinated fashion. The interactively selected navigation path determines how the information is combined and presented. For example, a displayed word can be linked to a frame of video so that when the user selects the word, the corresponding video frame is presented.

In the United States today, both analog and digital storage systems are common. Storage systems vary widely from slides and audio tapes to computer formatted DVD-ROMs. The most common analog storage systems are VCR/TV for video, slide projection for images, overhead transparencies for images and text, cassette tapes for music and vocalization, and a human speaker for vocalization. The most common digital storage systems are CD-ROM (originally developed for audio), digital videotape (originally developed for video), and traditional computer magnetic media. Nowadays each of these digital forms can be used to accommodate most media.

All forms of media have a common characteristic, called resolution, which is a measure of its detail. In an image, there is a finite amount of information that can be captured, and there is a finite amount that can be displayed. Using a camera to capture a picture, then developing the film produces an analog photograph. Film is not a perfect continuous medium and magnification of a resultant image produced on paper shows separations between colors. This is the grain or resolution of the analog image. When converting an image to digital, a device known as a scanner collects a set or sample of the colors from the photograph for storage. This set of stored colors defines the digital image's resolution. The higher the number of colors stored for an image, the higher the resolution. It is possible to capture the image at a resolution close to that of the analog grain resolution; however, the resolution of the display device for which the image will be shown is much lower. Resolution applies similarly with other media types.

Five of the six media categories are visual media that can use color. Few current color input and output technologies reproduce all human visible colors, so that digital color involves some inherent compromise. Within the human eye, rods and cones use photosensitive chemicals to relate color information to the brain. The rods use *rhodospin*, which is most sensitive to blue light and provides us with our *night vision*, whereas, cones are sensitive to short, medium, and long wavelengths of light. Traditionally these wavelengths have been classified by color and the cones are referred to as red, green and blue cones characterized by separate RGB wavelength values. As illustrated in Figure 1, each cone type transmits information for a range of colors. About 64% of all cones in the eye are red, 32% are green, and 2 % are blue.

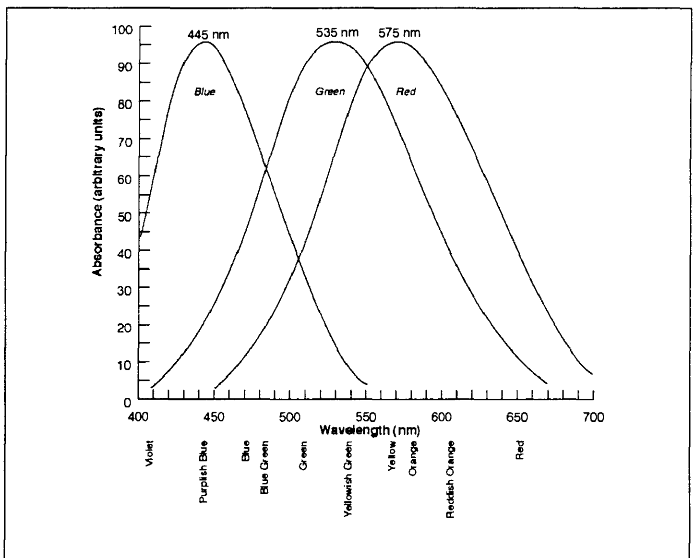


Figure 1. Arbitrary absorption of light curves of the cone system

Due to the fact that there is overlap in the light ranges that different cones perceive, humans typically don't think in RGB terms but think in terms of hue, saturation, and intensity (HSI). Hue is the frequency (color) of the light, saturation is the amount of different frequencies mixed in with a hue, and intensity is the perception of brightness and reflectivity.

The Commission Internationale De l'Éclairage (CIE) defined the total range of light that the majority of humans can perceive mathematically in 1931, which produces the CIE diagram, shown in Figure 2. This entire range of colors is referred to as color gamut. Each color output device may display a different subset of colors in the human color spectrum, thus each may have a different gamut as illustrated in Figure 3. For example, different RGB monitors may have different focal points for red, green and blue primaries, and thus provide different gamuts. This raises serious consistency concerns in the acquisition and display of color images.

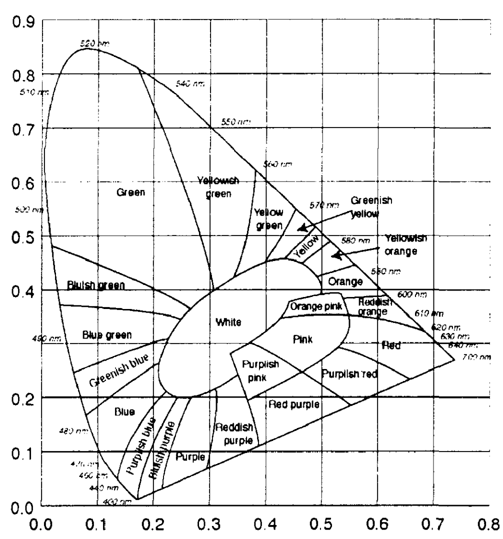


Figure 2. CIE 1931 Diagram.

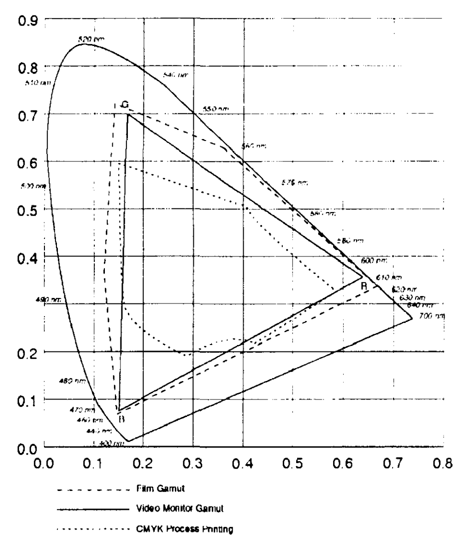


Figure 3. Color Gamuts on the CIE diagram

White as perceived by the human eye is quite variable, as shown by the large area in the CIE diagram. Two different colors within the white area are obviously different, but generally, each is perceived as white as the other. This is because the optical neural system adjusts to different white points. By heating a black body at different temperatures, different wavelengths of light produce an arc through the white section of the CIE diagram called the color temperature scale. Light producing objects such as incandescent bulbs are measured on this scale and produce a color temperature near 2854 K, whereas indirect sunlight is much bluer near 6774 K. Devices such as monitors can have their white point adjusted, but there is no single *correct* or *standard* white point.

Human perception of brightness is not directly proportional to light intensity. Instead it is directly proportional to the logarithm of the light intensity. Certain colors such as blue are also perceived to be less intense than those in either the green or red spectrum. Due to this fact, images stored digitally may need an adjustment known as gamma correction which applies a weighted logarithmic algorithm to the color values in order to adjust the brightness. Different operating systems have built-in color lookup tables that always perform an amount of correction. This can cause varying problems during development of multimedia projects. Most Microsoft® Windows® operating systems provide no built-in gamma correction, whereas the Apple MacOS operating system's gamma correction can be set between none (1.0), standard (1.8), and NTSC (2.2). Some workstations have a default gamma correction factor as high as that of NTSC, e.g., those from Silicon Graphics.

The one media that is not visual is perhaps the most important media category. Studies have shown that poor visual media can be offset with high quality audio. For example in experimental studies with High Definition Television (HDTV), Neuman improved the perceived quality of video by increasing the audio quality only (1987) and Sasse concluded that low video quality is considerably less disturbing to an audience than bad audio (1994).

From an analog-presenter's standpoint, trying to organize multimedia without any *method* can be disastrous. The presenter must gather up the media to be presented, such as transparencies, audio tape, and slides, then manually attempt to coordinate the presentation of particular pieces of each media type and encapsulate them all into a coherent presentation. However, having to run a slide projector, control a cassette recorder, and flip transparencies on an overhead, while talking makes the whole effort something of a circus act. Trying to accomplish all of the appropriate button pushes at the right time is a scheduling nightmare. On the other hand, a computer, a digital device that can accommodate all of these media, can relatively easily be programmed to coordinate the display of all these types of information at appropriate times. If the appropriate scheduling framework is setup ahead of time, the presenter can interact with the computer through simple events such as key presses, pointer actions, or absolute timing events, without having to interact directly with individual display devices.

Digital multimedia presentations are transmitted to an audience through several types of delivery systems. Stored digital information can be retrieved for delivery in two distinct modes: data stored locally on either optical or magnetic media, or stored and accessed remotely over a network by another computer-controlled device. Whichever delivery technique the multimedia producer chooses, resolution is a major obstacle.

Looking first at the simple storage issue, computers store media as files of binary information. These files can be quite large depending on their resolution and type. The basic rule is that the higher the resolution, the larger the media data set. Magnetic media in the form of floppy disks are cheap and effective in storing media as long as the presentation is not too large; however, only 10 seconds of a very small movie clip can be stored on a 1.4-megabyte floppy. Today's 630-megabyte CD-ROMs can store approximately 50 minutes of small frame, but eye and ear pleasing digital audio with video. The physical cost of producing a CD-ROM has decreased dramatically over the past five years, making CD-ROMs a viable solution to data storage and delivery. A new standard for Digital Video CD-ROMs is available that allows full screen, full length Hollywood movies to be stored on a single disc. Nevertheless, no matter how much information can be stored, accessing the information fast enough to present it is a problem. That is, no matter how much information is stored on a disk or CD, media must transfer at very high speed to the actual presentation devices to provide a high-quality multimedia product.

Networked information storage and presentation is quite popular today, because it allows for easier centralized information updates and for dissemination to a large audience quickly. However, network bandwidth limits the amount of information that can be presented in real time. Though in theory the bandwidth is large, the limitations of current networking create significant delays when large amounts of data transmit. This means it is usually not feasible to try to deliver presentation material in real time over a network; instead, the user must download the material, save it to the local host as a "batch," and later replay it. Even so, a one-minute movie clip may take twenty minutes of "download" time on a saturated network, before the user may view it.

Computer controlled multimedia presentations must organize the information in either sequential or hyperlinked fashion. In either case, the producer applies some time base to synchronize the user's requests for information with the computer's delivery of that information. *Events* are the actions that signal the multimedia device to present information in one or more media forms. *Timing events* are synchronous time based events created by the launch of an application, a start button push from a running application, or a signal from an internal timer. *Explicit events* are those linked to explicit user generated (asynchronous) signals, such as mouse or keyboard actions. Using these two types of events, a designer has the ability to allow interaction, create a strict linear time-based presentation, or combine the two.

The information in each media category must be stored and manipulated in an appropriate digital format. Within each media category, software and hardware manufacturers have developed various hardware and software combinations to operate on datasets of specific types. Producers need to digitize datasets or create them initially in digital form. Software has now evolved to work with a variety of different media datasets. The size of datasets and the need to meet real-time constraints in data delivery (e.g. for sound and animation) makes the combination of hardware and software critical to the presentation. With the wide range of software and hardware combinations available, making educated decisions on which equipment to purchase is difficult but essential.

Once the hardware and software are chosen to provide a platform for digital multimedia, the challenge in building multimedia material is to accumulate base information and package it in a *presentation* that defines, synchronizes, and links appropriate media clips and events. The speed of the computer hardware, the size and transfer rate of hard drives, speed of networks, and resolution of display all impose limitations on the level of media that can be delivered or depicted in real time. The need to consider these practical limits plus the need for creative and artistic expression of content makes labor costs by far the most important factor in the construction of the multimedia presentation.

Software Development

Since the introduction of computers, scientists have written software to accomplish meaningful tasks. As the tasks grew in complexity, the software also grew. It took more than one developer to write a single program. Out of the need to coordinate the complex activities of a number of developers grew the science of software engineering. Software engineering processes aid in the creation of large software projects. They utilize *document-driven*, *code-driven*, *risk-driven*, or other specific models. These processes consist of a set of ordered steps to define, develop, and maintain software. The processes also establish the transition criteria for moving from one stage to the next, defining completion criteria for the current stage, plus decision criteria and start criteria for the next stage.

The most basic development methodology is the informal *code-and-fix model*. This model involves two steps: write code, then fix the problems in the code. There are inherent problems with this technique; code becomes unstructured, the software doesn't match what the user required, and the code becomes expensive to fix. Even the best developer using this model begins programming almost immediately after being given a brief description of the problem. The stakeholders have little or no involvement until a final product is produced. The result is usually not what was expected. Stakeholders often pull out of the project or demand expensive redevelopment. The *evolutionary-development model* (McCracken, 1982) has expanded on the code and fix model but is still a code-driven process. It has the similar downfalls of producing highly unstructured code and lacks support for project planning.

With the problems of the code-driven processes, computer scientists developed the *stagewise model* to involve the stakeholders immediately in the project. The model produced a set of successive stages that software engineers follow, including planning, writing specifications, testing, and evaluating. This was eventually refined into the *waterfall model* (Royce, 1987). The waterfall model, a document-driven process, involves building the product through a set of careful product specification and development steps designed to guarantee extensive documentation of the final product. Of course, complex system specifications are very difficult to define. This leads to a further refinement of the development process called the *transform model*. The transform model expands on the waterfall model to allow the documentation to change as needed.

To address the problems of previously mentioned software development models, focus shifted towards an emphasis in identifying project risks, and using development models that allowed early identification and resolution of risks (Boehm, 1988). The resulting risk-driven process is the *spiral model*. This model can accommodate most previous models as special cases, but specifically takes into account the amount of project risk involved at each stage of development.

Multimedia Development

Most multimedia currently produced goes through a document-driven process, such as storyboarding or scripting. Because details of the final product are impossible to define at the project's start, the specifications change as needed. The problem with this approach is just like with software: each specification implies changes in the needs for the underlying media. If too many changes occur, the developer may have to scrap the initially developed media components, with the entire development starting over. A risk-driven process evaluates the amount of risk involved at each stage of product development and focuses resources on resolving the most risky aspects first in order to avoid catastrophic redesign or failure. The spiral model is more adaptable to the full range of software project situations than the primarily document-driven or code-driven approaches. The same methods that apply to software engineering can apply in developing multimedia titles with further extensions. The WinWin Spiral Model is an extension of the spiral model that has proven to be effective in producing multimedia applications (Boehm, 1997).

The WinWin spiral model uses a cyclic approach to develop increasingly detailed designs of the multimedia application. The cyclic model involves seven main activities (Boehm, 1997).

- Identify the system or subsystem's key stakeholders.
- Identify the stakeholders' win conditions for the system or subsystem.
- Negotiate win-win reconciliations of the stakeholders' win conditions.
- Elaborate the system or subsystem's product and process objectives, constraints, and alternatives.
- Evaluate the alternatives with respect to the objectives and constraints; identify and resolve major sources of product and process risk.

- Elaborate the definition of the product and process.
- Plan the next cycle and update the life-cycle plan, including partition of the system into subsystems to be addressed in parallel cycles; this can include a plan to terminate the project if it is too risky or infeasible; this must include a plan to secure the management's commitment to proceed as planned.

Boehm applies this model to multimedia software development and illustrates that it can produce wins for the stakeholders. Boehm approaches such projects from a software-engineering standpoint in which his students create a piece of multimedia software to meet the requirements of a specific subset of an entire multimedia platform, such as a video viewer or hypertext display engine. A producer could use the WinWin model to produce a multimedia title; however, the approach may be different. Boehm's projects concentrated on designing the tools for a multimedia presentation. Since a large number of high-level multimedia development tools already exist, a typical multimedia development project does not involve a large number of programmers. Instead, a team including artists, storytellers, and programmers work together in production. The goal of the development team is to produce a clean, exciting, and fast paced multimedia application for which an audience stays interested. The final product must have very little *wait time;* thus, once the prototype reaches a final revision the team must optimize the data and multimedia project.

For each iteration of the spiral model, a set of objectives, constraints, and alternatives arise, the producer evaluates the prototype, and the next level of product evolves. If the stakeholders decide to change or add another delivery system, the risk for failure is huge; the project will most likely have to start over, and nobody wins. A new model that extends the WinWin model defines the product as not only an application but as an application with corresponding digital media. The guidelines for the new model define media collection and organization for possible delivery system migration; the multimedia product can be a success even if the stakeholders decide against the original application. The new model allows more flexibility to reach a success. Since no stakeholder at any one time knows enough to define a finished product, an organized library of high-quality archived digital media is still a success.

Chapter 3

Hardware and Software for Digital Media

Although different media types have typically been treated differently as though there were distinct lines separating the types of information they contain, once media items are captured and stored digitally the data from different media are actually quite similar. What has historically defined the lines that keep these media separate starts with the fact that humans think of their five senses as separate, so they wish to retain the same separation in working with media. Other separations are in place due to the fact that most technologies were developed to digitize one, or at most two, types of media. Separation also occurs when a media is bound for a specific media distribution.

As the process of digitizing media becomes simpler, many of the lines separating the media begin to blur. Since modern hardware and software can handle the complexities of media with more ease, many hardware and software solutions now encapsulate multiple media. As the software in particular becomes able to handle nontrivial operations on multiple media data types, the complexity of using multiple software packages and possibility for mistakes in media handling decreases. As the use of multiple media software continues to grow, it blurs the lines between media distinctions.

The section that follows describes media in terms of classical, distinct media types. Each media category typically has three phases in its *digital lifetime*. These are acquisition/input, storage/display/output, and synthesis/transformation. In discussing these phases, I assume that the multimedia producer wants to capture and store data at the highest archival quality possible, using subsequent optimization and data reduction at a latter stage of processing to reduce resolution for particular applications.

Text Processing

Storage/Display/Output

The data of text media are character strings. To display text media, the multimedia producer provides the character strings, along with specifications for the font, font size, weight, character spacing, and placement.

Acquisition/Input

There are three ways to acquire text as digital media: keyboard entry, optical character recognition, and document imaging.

Keyboard entry is the process whereby text is entered by users with a hardware device such as a keyboard. Special data entry devices for motor impaired users are also included in this group. The text is typically collected using software such as a text editor or word processor. A large number of different editors and word processors are available (Thompson, 2000), each with different capabilities, such as spell checking, availability on different hosts, etc. For a large amount of digitizing, this option is the slowest.

Document imaging is the process of scanning documents into a computer and storing them as the graphic file. This is by far the fastest technique and least expensive of the three methods but the file size of each digitized document can be as much as 15 times larger than the strict text version. The captured text cannot be edited except as a picture. However, this can be the best technique if the primary goal is only to display the text. A large variety of scanners are available (Thompson, 2000) with much of the variety related to how the paper or sensing elements are moved, e.g., sheet feed scanners, flatbed scanners, and drum scanners. Each scanner has benefits over the other. Drum scanners are typically too expensive for use in simple document imaging, thus they will not be discussed in detail here. Sheet feed scanners work in an environment where only a small amount of personal document imaging is required. For any bulk document imaging, a good flatbed scanner of at least 300 dots per inch (dpi) with an automatic sheet feeder attached is a necessity.

Optical character recognition (OCR) is a two-pass process that starts with scanning a document into a digital image, then using specialized software to recognize characters in the image to form a comparable text file. Each OCR application has different capabilities. The competition in this market niche over the years has weeded out the poorer performing products. An example high quality system is ScanSoft's OmniPage Pro®, which handles the character recognition with great accuracy and provides a simple interface for correcting mistakes. For large amounts of digitizing text, OCR is faster than keyboarding, but is not as fast as document imaging. OCR requires user input to make corrections when text is unrecognizable.

A new hybrid technology developed by Adobe Systems, Inc. uses both document imaging and

OCR to produce a well-known proprietary document format named portable document format (PDF). The software uses a scanner to create a document image. Next, it uses OCR techniques to recognize as much text as possible in the image. Finally, all information that cannot be recognized as text is saved as smaller graphic images. The resulting document can be displayed with free viewer software, provided for most operating systems from Adobe. This technique provides much of the file size savings gained by OCR, but removes the high cost of user interaction due to the fact that unrecognizable text can be saved as images. Only when the text is to be edited does the user need to resolve those words originally saved as images.

Synthesis/Transformation

The advantages of transforming images to text are obvious: text is editable and can be highly compressed for storage without any data loss.

Two Dimensional Graphics

Storage/Display/Output

The data for two-dimensional graphics describe some type of image. The difference in how the image is stored digitally determines how the acquisition should be performed. Raster graphics are images depicted by a fixed number of picture elements (pixels) on a rectangular grid, whereas vector graphics are images formed by geometrical models that describe lines, points, etc. The quality of raster graphics is resolution-dependent. In particular, they can appear jagged and lose detail if they are scanned or created at a low resolution, then printed at a high resolution. In contrast, vector graphics are based on geometric relationships that will always produce crisp, clear lines no matter how the image is scaled. Vector graphics are best suited to logos and bold graphics, whereas, raster graphics are best for photographs.

Raster graphics use a rectangular grid divided into equal size pixels. Each pixel contains one or more data values e.g., describing what color should be displayed. Additional information describes aspects of image resolution, such as the physical size of the image, the number of pixels in the horizontal and vertical direction, and the number and size of the colors.

The information stored in each pixel of a raster graphics file is usually determined by how the image will be depicted. The primary output devices for raster graphics images are monitors and printers, which have different requirements. Monitors use three separate values to control the intensity of three separate *guns* to produce the amount of red, green, and blue mixed on the display surface to form a single color. Color printing processes actually mix cyan, magenta, yellow, and black inks in order to produce colors. These two different color mix processes determine how a given image will be represented. The different output devices require different information. Typically, the color information needed for different reproductions of an image is called a *color table*. Not all graphic output and input devices use the same color table. The values that represent colors in a file are typically set by a number of bits per pixel per color. An eight bit per color, red, green, blue (RGB) graphic or (24 bit RGB image, sometimes referred to as true-color) has a color table with 255 different shades for each color, thus providing a composite of 2^{24} different color possibilities. This is thought of as almost good enough to fool the human eye. An eight bit per color, cyan, magenta, yellow, and black (CMYK) image is referred to as a 32 bit CMYK image and can produce 2^{32} different colors.

The size of the rectangular grid for a raster graphic is limited only by the processes used to capture and save the image. A higher number of pixels results in a higher quality or resolution raster graphic. The ratio between the number of pixel elements and the physical dimensions of the object being imaged are typically used to describe the resolution of an image. For example, if the rectangular grid were measured in inches then the ratio would be given in pixels per inch (ppi) (also known as dots per inch or dpi). Images are typically printed at high resolution, from 150 to 11,000 dpi, compared with images displayed on a screen at 72 to 150 dpi.

The file formats for storing raster graphic images are pretty much the same. The files store the raster information as a block of data with a description of the data accompanying. Many different formats exist that support different bit depths, description information, and using a variety of compression techniques. For vector graphics, there are very few standard file formats. Most vector graphics software uses its own proprietary format. The most widely used vector graphics format is encapsulated postscript (EPS), and since Adobe Illustrator was one of the first vector drawing packages, the Illustrator format (AI) is also widely used. A new standard defined by the W3C organization called scalable vector graphics (SVG) looks very promising.

Acquisition/Input

The most common way to capture raster graphic images for small objects is with a scanner. There are different types of scanners commonly used for documents: flatbed, sheet feed, and drum scanners. Flatbed scanners capture the image by moving a bank of sensors along a document placed flat on a glass surface. With a sheet feed scanner, the document is moved past stationary sensors. With a drum scanner, the document is attached to a drum and scanned by stationary sensors while being spun at high revolutions.

Scanners work by using sensors arranged in a rectangular grid to capture the amount of light reflected from or passed through the surface of a document, divided in a comparable grid. Moving the document or the sensors allows the whole document to be imaged with a relatively small bank of sensors. The digital device used for sensing light is a charged-couple device (CCD), a solid state component that changes a quantity of light into a corresponding electrical charge. By itself, a CCD can only measure light intensity, and not the color of the light. Thus, using just CCDs in a scanner produces only a grayscale image, where black surfaces reflect no light and white surfaces reflect all of the light. Each CCD converts the light intensity at its pixel to an electrical signal strength that is quantified as a numeric (digit) value. The electrical signal strength can be quantified at different resolutions, or bit depths, ranging from 4 bit to 14 bit (16384 distinct values).

Color can be acquired with a CCD by placing light filters in front of the CCD to eliminate all but certain type of light, which can then be measured as described above. For example, using a red filter in front of a CCD allows the intensity of red light to be measured. A color image can be formed by using multiple filters to separately scan different color channels, then combine the channel values together. An alternative is to divide the light by using a prism to redirect light frequencies to separate CCD elements, e. g., placed at the red, green, and blue focal points. Prism based systems allow multiple colors to be measured at a single time, but require more CCD elements. Filter based systems typically make multiple passes to complete the scan, using the same bank of CCDs but changing the filters for each pass. Though three-color scanners are the norm, some scanners capture more than three colors over a larger portion of the color spectrum, then use mathematical transformations to create the desired colors description data (RGB or CMYK). These scanners usually produce an image with higher color fidelity, i.e. better match to colors as perceived by the human eye.

For flatbed, passthrough (sheet feed), and film scanners, the horizontal resolution of the scanner is determined by the physical placement of the bank of CCDs, which is referred to as the scan line. The vertical resolution of the scanner is determined by the quality of the stepper motors used to move the document or sensors. Though flatbed scanners move the scan line whereas sheet feed scanners move the document past the scan line, the quality of the movement is comparable, so vertical resolution is the same. Film scanners are similar to sheet feed scanners, but use much finer steps to produce higher resolution images. During the drum scanning process, the original document (transparency or reflective artwork) is attached to a glass drum, which is then spun at high speed while a single beam of light is passed through (or reflected off) the original. This process provides a large number of passes to be made very rapidly, which allows a greater range of adjustments to be made during scanning, resulting in a greater tonal range and thus sharper images.

There is little reason to pay more for a scanner with higher resolution capability than will ever be used for the images that are captured. For example, there is no reason to purchase a drum scanner for images that will be only used on web pages. However, if the scanned images are going to be used for multiple purposes, it is a good idea to use a scanner that will capture images at the highest resolution needed. The following table lists scanner specifications dependent on image use.

	Optical Resolution	Color Depth	Bits per Color	Type of scanner
Images bound for display on a monitor or low quality print				
Minimum Specs	300 dpi	30 bit color	10 bit per color	Flatbed, Sheet feed
Maximum Specs	600 dpi	36 bit color	12 bit per color	Flatbed
Images bound for standard print				
Minimum Specs	600 dpi	30 bit color	10 bit per color	Flatbed
Maximum Specs	2000 dpi	36 bit color	12 bit per color	Flatbed
Images bound for prepress or photo quality print				
Minimum Specs	600 dpi	36 bit color	12 bit per color	Flatbed
Maximum Specs	5600 dpi	48 bit color	16 bit per color	Flatbed, Drum scanner

Table 2. Scanner Specifications

Not all scanners have equal capabilities, so users should be sure to read the fine print when getting a scanner. Many scanners list resolutions with digital interpolation. Optical resolution is the amount of information that the scanner gathers using its CCDs and lenses for magnification. The higher the resolution means that the scanner has better optics. Digital interpolation is a way to increase the number of pixels captured beyond the true optical resolution, but image quality is largely determined by optical resolution, not interpolated resolution. Other scanners include unique technologies to cut down on moving parts and increase the quality of scans.

The number of colors a scanner captures also plays a role with the quality of a scan. Just as optical resolution plays a factor in how many pixels the scanner captures, the number of colors and the color bit depth affects the scanner's resultant quality. Even though the final destination of an image may be twenty-four bits, capturing an image at 36 bits and using mathematical calculations based on human perception of color can produce better images. Be aware that some scanners specifications quote they are 36 bit, but this value may be derived mathematically rather than from direct scan values.

Acquiring the best color possible initially reduces the amount of post acquisition work and results in the smallest amount of data loss. The color variation between different images is so complex that the acquisition process for high-quality scans is probably impossible to automate. The entire scanning interface (hardware and software) should offer the user options to adjust the color levels such as the black point, white point, and color balance adjustments during the scanning process. Many step-by-step scanning procedures can assist with this task (Rodney, 1998a).

If the acquisition interface does not provide directly for color adjustment, the adjustment must be performed within the raster image editing software. A good example of this is the imagery acquired with current digital cameras. Acquiring the image with the highest possible bit depth increases the amount of data available and decreases data loss. Adobe® Photoshop® allows the user to perform color level corrections at bit depths of 12 bits per color channel. Converting the image to 8 bits per color channel after the color correction provides the highest quality acquisition. Some devices do not offer higher than 8 bits per color channel, but to improve the image quality, color corrections should still be done even at the cost of lost information (Rodney, 1998b).

Printed material such as magazine prints and newspaper images are lower resolution than photographs, somewhere between 85 lines per inch (lpi) and 144 lpi. They use a printing technique called halftone printing. Halftone technology uses a lattice of uniform squares with a dot printed in each lattice cell. By using dots of graduated size, a desired fraction of each lattice cell may be colored (the ink color) and the remainder of the cell may be left white (the paper color). The illusion of a continuous variation of gray or color tones occurs because the eye averages features that are not clearly seen as object details. In newspaper photographs the halftone lattice is coarse enough to be seen easily. In glossy magazine printing, the lattice is much finer and therefore more difficult to detect. The scanning interface can be used to descreen halftone-printed documents, which will acquire a higher quality image, by setting the scan resolution to match the printed resolution as closely as possible and selecting the descreen option.

If the final destination for an image is in a color model such as CMYK, scanning an image with only red, green, and blue values limits the quality of the result. With a larger color model, more information can be gathered and stored. The same type of mathematical calculation used to reduce the bit depth of an image can derive higher quality images with scanners that capture more than three color channels. Typically in pre-press applications where the image is scanned from a photo and bound for press, the number of colors needed is higher than what the RGB color model can reproduce. Certain scanners include options for scanning with more colors.

The software that accompanies the scanner can also make a big difference. Usually a plug-in for Photoshop® or other comparable software packages is provided to drive the scanner and image acquisition process, but the features that can manipulate the scanner can make a big difference. For example, color sync technology, a way to match color to print based on device-independent CIE color spaces, can make a big difference in successful scanning.

Digital Cameras and Video Cameras

Digital and video cameras work on principles similar to those of scanners, but typically use a rectangular array of CCDs instead of a single row of CCDs. Because they capture sequences of images in real-time, cameras cannot use multiple passes. The resolution of individual images is limited by the resolution of the CCD array. Three chip CCD cameras split the light into three separate color images that each project onto its own CCD array. Single chip CCD cameras split a single array of pixels to gather three separate color channels. For example, the first three lines of an array would be used to capture red channel on the first line, the green channel on the second line, and the blue channel on the third line. This is repeated over the entire array in order to capture an entire image.

Most of the consumer versions of digital cameras use a single CCD system and only capture enough pixels for screen quality or low quality print images. In order to get images with high enough quality for standard print, a camera should support at least 1152x864 pixels. For high quality print the camera should support at least 3000x2000 pixels, which is roughly comparable to the resolution of traditional photographic film.

Another option for shooting pictures digitally is to shoot standard film, then use a high-resolution scan process. The Kodak Photo CD is a *packaged* process that combines high-end film scanning with traditional film printing (Eastman Kodak Company, 2000).

Vector Graphics– acquiring

Vector graphics file formats store information about the mathematical objects occurring in the image. Typical objects of interest are points, lines, circles, rectangles, and Bezier curves. Each of these objects can contain information about size, location, color, stroke width, color gradient direction, and more. Because this information is defined mathematically, the images are resolution independent.

Though vector graphics are most often used in computer generated images, it is also possible to acquire vector graphics images. The process to digitize a vector graphic begins with scanning the image as a raster graphic. This raster is then analyzed by software that identifies objects, creates the object description, and thus converts from raster to vector. A second, much more labor intensive option is to place an image on a digitizing tablet and have a user simply trace the objects to create the vector descriptions.

Synthesis/Transformation

Once digital images have been captured, a wide variety of transforms can be used to synthesize additional images. Adobe Photoshop® is the premiere raster graphics image editor on the market today, but there are many other software packages (Thompson, 2000). Input devices such as a mouse or graphics tablet allow the artist to interact with the software and move tools across a digital canvas.

The input tools for manipulating and synthesizing images consist of mice and other pointing devices. A digitizing tablet is a much more specialized device consisting of an electronic tablet and a cursor or pen. A cursor (also called a puck) is similar to a mouse, except that it has a window with cross hairs for pinpoint placement, and it can have as many as 16 buttons. A pen (also called a stylus) looks like a simple ballpoint pen but uses an electronic head instead of ink. The tablet contains electronics that enable it to detect movement of the cursor or pen and translate the movements into digital signals that it sends to the computer. For digitizing tablets, each point on the tablet represents a point on the display screen in a fixed manner. This differs from a mouse, in which all movement is relative to the current cursor position. Certain features, such as pressure sensitivity that allow an artist to apply more pressure on the tablet to apply more ink, are available in specific brands.

There are several software packages for editing vector graphics (Thompson, 2000). Computer Aided Design (CAD) graphics packages are typically based on vector graphics with features selected to support the manipulation of traditional "mechanical drawing" operations so they are included in this list as well. Some software packages combine the ability to manipulate both raster graphics and vector graphics. A prominent example is Corel® Painter™, a package recommended for any artist who will be creating work from scratch.

Three Dimensional Graphics

Storage/Display/Output

Currently, there is a distinct line that divides computer 3-D graphics into two categories. The first category includes those graphics that are actually 2-D images (snapshots) of 3-D objects presented to the end user in a manner such that the 2-D image depicts a 3-D object or scene. We call these visually depicted 3-D graphics. The second category is actual 3-D objects stored with appropriate 3-D constructive geometry, along with additional data such as surface colormaps that can be used to present an appropriate depiction from an arbitrary viewpoint. The only commonality between these two categories is the presentation of the material seen by the viewer. With either option, a two-dimensional image displayed on a video monitor is the primary output. With the 2-D snapshot, there is exactly one view available, whereas the 3-D object can be rendered from a range of viewpoints to produce a range of views. In either case, coloring, shading, and perspective are adjusted to present a more *photo realistic* look. Other techniques have also been developed to try to trick the sight sense into believing it's actually seeing three-dimensions on the flat display by showing slightly offset images to each eye. A technique such as wearing the funny 1950's red-blue glasses has existed for a long time and there is some renewed interest in this technology using video interlacing on fast video displays. One advantage the geometrically defined category has is that it is possible to actually construct solid models from the defined objects. Since the geometry specifies the actual shape of the objects, facilities such as SanDiego Super Computer's TeleManufacturing Facility http://www.sdsc.edu/tmf can create physical 3-D objects.

There are typically two different types of visually depicted 3-D graphics, a panorama and a visually depicted 3-D object. Panoramas are images containing all of the visual information available in a 360-degree view from one single viewpoint. For depiction, the image is mapped to a virtual cylinder, cube, or sphere, using a view port to show only part of the image at a time, giving the end-user controls to move around, and establishing the 3-D feel. Visually depicted objects are images of an object taken at a sequence of predetermined viewpoints (typically occurring as a camera is rotated around the object), then the rotations are combined together into a sequence. Multimedia software gives users controls over the objects so that they feel they are rotating the object. Initially, all of the imagery to construct these graphics comes from 2-D graphics. Currently there are several proprietary formats used to store the finished visually depicted 3-D graphics. The chosen final format determines what software and hardware is needed in order to produce a viewable presentation.

Geometrically defined 3-D models extend 2-D vector graphics into 3-D equivalents. Components of a 3-D model define a mathematical 3-D space and often add information describing the type of interconnections between components in the space. Geometric models include engineering and architectural structures, molecules and other chemical structures, geographic structures, and other man made structural objects. Additional information can be added to each component to describe arbitrary data values or color/texture information. Geometric defined models typically use components such as points, lines, surfaces, and volumes in a hierarchical fashion to systematically construct much more complex scenes. Most simple 3-D software uses points, lines, and interconnections to define simple planar polygons. A group of connected polygons, forming a polygon mesh, can describe the surface or surfaces of a solid, thus representing the simplest geometrically defined model. However, to represent smooth curved surfaces, such as the fender on an automobile, polygon meshes fall short. In such cases cubic spline surfaces constructed using a series of polynomial equations are more appropriate. The most commonly used cubic curve for this purpose is a non-uniform rational B-spline (NURBS).

Depicting a 3-D geometric model on a 2-D display screen is more complicated than strictly displaying 2-D graphics. The typical approach borrows terminology and methodology from photography. To depict a 3-D scene, a camera is placed at a viewpoint, then oriented to capture the desired view of the scene. This small amount of information allows the 3-D object to be projected onto a 2-D projection plane, which in turn is mapped into a viewport for display. In general, projections transform points in a coordinate system of dimension n into points in a coordinate system of dimension less than n. The change due to projection is also referred to as rendering. Thus the resultant 2-D image is rendered from the 3-D object, based on viewpoint and orientation.

In rendering, it is important to distinguish the difference between 3-D objects based on polygon mesh versus NURBS surfaces. NURBS surface processing is more computationally intensive, since it must describe and store the NURBS data, and also construct the smooth curves. Many different 3-D rendering techniques exist, both to construct surfaces and to adjust color, shading, and other aspects of depiction to provide more photo-realism. Most 3-D software packages include, in order of complexity, point-cloud, wire-frame, flat, Gouraud shaded, Phong shaded, ray-tracing, and radiosity rendering techniques. Typically a rendering algorithm creates a 2-D bitmap image. However, since 3-D models are geometrically defined, several rendering algorithms exist that produce 2-D vector images.

Typically, the files that store geometrically defined 3-D graphics use proprietary formats, defined by the software used. One of the first proprietary formats defined by AutoDesk® for its CAD software was the DXF format. The format is an ASCII based format that is now documented to allow developers to read and write it easily. However, it only supports polygon meshes, and has many other limitations with regards to colors and texture maps. To correct these limitations, a consortium (www.vrml.org) was formed to create a new specification, which became known as the Virtual Reality Markup Language (VRML). Work progressed among the consortium, but as competing technologies were introduced and large companies became involved, the pace of progress almost stopped. VRML2 was introduced in 1997, but it has yet to catch on as a standard. The consortium, now named webl3D, has subsequently produced a new format named X3D, based on the extensible markup language (XML). The consortium is providing full specifications to the format as well as software development kits for incorporating functionality, so there is still hope that this format will emerge as a non-proprietary standard.

Acquisition/Input

Unfortunately, the techniques, software, and hardware for the categories of visually depicted 3-D graphics and geometrically defined 3-D graphics have virtually nothing in common. The most popular form of visually depicted 3-D graphics is Apple's QuicktimeVR; however, others exist. Most of the hardware to create panoramas is similar among the competing technologies. First the developer must capture a panoramic image from the desired location. Very expensive special cameras and lenses exist for this purpose. The cheapest panoramic solution involves a camera tripod, special panoramic tripod head configurations, and a camera (digital or film based) with manual light exposure control. The camera is used on the tripod head to successively capture fractions of the entire 360-degree view until the entire scene has been stored. Using wider-angle camera lenses, such as 18mm, can increase the amount of information captured in one image and reduce the number of images required. Once captured, all of the images from one viewpoint must be *stitched* together into a single panoramic image, as shown in Figure 4.

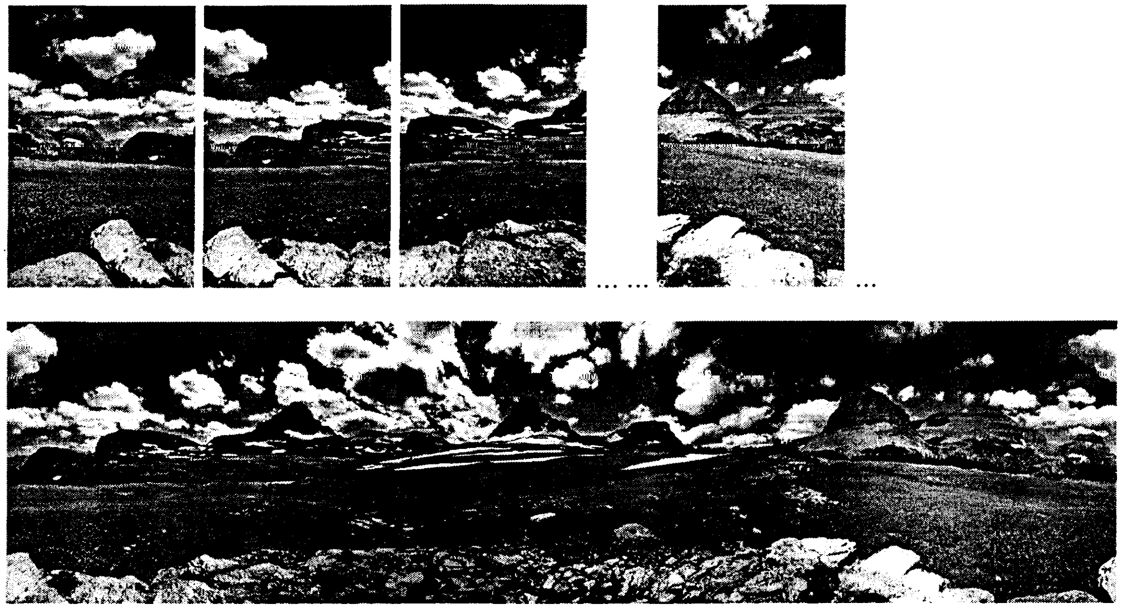


Figure 4. Stitched Panorama

For visually depicted 3-D object movies, QuicktimeVR is the most popular current format. Capturing all of the images for a single object movie requires special camera tripods, a camera (digital or film based) with manual light exposure control and a turntable. Fair results can be achieved by capturing an object from different viewpoints without special equipment, as illustrated in Figure 5. Companies such as Kaidan have created special object rigs that offer more flexibility. Once all of the images are captured, the software can then create one object movie. Software for making visually depicted 3-D graphics includes Apple's QuicktimeVR Authoring Studio, software from LivePicture, and others (Thompson, 2000).

Figure 5. Example Object Frames

Capturing 3-D geometrically defined graphics is not trivial, and effective, affordable capture techniques and/or hardware are scarce. One technique uses laser measuring devices to capture the geometry, and CCD technology to capture color at points along a surface, thus providing surface constructs as well as colormaps needed for definition. Another technique uses 2-D graphics capture techniques such as video and performs triangulation to determine the 3-D geometry. Similar techniques are used in very expensive scientific instruments, such as MRI CT scans, to reconstruct 3-D geometry (Thompson, 2000).

Synthesis/Transformation

Since visually depicted 3-D models are actually 2-D images imitating 3-D, the developer can use 2-D graphics tools to create or modify initial imagery. The manipulation of 3-D geometrically defined graphics requires techniques that can transform the geometry, color maps, texture maps, and other data. Depending on the origin or final purpose for the geometry and the specialty of the 3-D software market, the developer might have to use several different tools to accomplish a given task. For example, computer aided design (CAD), 3-D animation, data visualization, and geographic information systems (GIS) all implement a variety of different transformations. Even Web-oriented tools have begun to add specific 3-D geometrically defined transformations for rapidly streaming objects from disk to rendered form for display. Refer to the hardware and software references for multimedia for a list of particular 3-D software packages (Thompson, 2000).

Video

Storage/Display/Output

The output of video is film that is displayed with a projector on a screen, or video signals that are displayed directly on a display device such as a monitor. The display requirements differ for both of these outputs. Film typically displays 24 entire still frames per second. Each standard frame of a 35mm film is comparable to 3656 pixels by 2664 pixels, but can be as high as 6144 pixels by 4096 pixels for formats like Vista Vision. Standard television video signals (NTSC) display interlaced 59.94 fields per second or 29.97 frames per second at resolutions of 640 pixels by 480 pixels. High definition television uses progressive or interlaced scan rates of 24 fps or 30 fps and is able to display 1920 pixels by 1020 pixels.

When frames are displayed at a rate slower than 24 frames per second, the human eye detects flicker. However, because of the large amount of data needed to produce video at this rate, the developer may have to drop the frame rate to something lower. The slower the frame-rate the smaller the amount of data that must be moved or generated in real time. Hence, slower frame rates (with comparably lower video quality) are often used across the Internet or even for multimedia kiosks.

Video bound for display to film, non-interlaced high definition television, and computer screens are all fairly similar. The primary differences are resolution and the number of frames per second. However, video bound for television in a format like NTSC has specifications that require special handling. Video for NTSC does not use square pixels and must present two frames, each as a field, in one frame. The total resolution at the International Radio Consultative Committee (CCIR) 601 standard quality for NTSC displays 720 pixels by 486 pixels, i.e. at a 4:3 aspect ratio.

Whatever format is used, the amount of storage for video can be tremendous. For example, one second of a possible NTSC video signal requires:

$$30\,frames \times (720 \times 486)\frac{pixels}{frame} \times 3\frac{bytes}{pixel} \approx 30\text{megabytes}.$$

With this amount of information, the developer can only work on a small amount of video at any time. Hardware and software exist that can compress the information to allow the developer to be more productive; however, compression can cause problems of its own.

Acquisition/Input

Techniques used to acquire video for use in the multimedia system depend on the source. For standard NTSC video signal, as regular video, S-Video, or professional component video, the developer must use an analog to digital scan converter. A variety of these exist at different levels depending on the resultant quality (Thompson, 2000). Depending on the scan converter's level of compression, it may be necessary for the capturing station to use extremely fast secondary storage to capture the data in real time. Redundant arrays of independent disks (RAID) level devices are typically used for this type of application. RAID devices are rated by performance at various levels. Level 0 stripes together multiple drives in parallel as a single drive. When data needs to be accessed or stored. The RAID unit does n operations at once, where n is the number of drives striped.

Assuming disks fast enough to capture the information exist, the architecture used to move data from the scan converter through the computer bus to the drive can become the bottleneck. Typically, a RAID is connected through the small computer system interface (SCSI). However, standard SCSI-1 and SCSI-2 are generally too limiting. To capture high bandwidth video in real time, a modern workstation requires a SCSI card that supports at least a 40 MB/sec transfer rate, such as a single channel ultra wide SCSI. An example of a scan converter that does uncompressed component video, the Aurora Video Systems' Uncompressed Igniter™ Video Capture Card, needs a sustained throughput of read and write of 13.3 MB/sec.

With the ever-changing computer technology field, a new interface that looks promising is the IEEE 1394 standard, or *Firewire*, designed for connecting devices such as hard-drives, video cameras, etc. that transfer data digitally. Firewire currently supports 50 MB/sec transfers with an expected 100 MB/sec transfer rate within the next year. It uses a peer to peer technology, which means data can be transferred from one device to another without having to be routed through the host computer's central CPU and bus. Sony offers camcorders and an interface that can convert analog NTSC signals directly to the DV (digital video) format that feeds directly to other Firewire devices. Many companies now offer camcorders that record directly in the DV format that can hook up to Firewire for transfer to a computer.

Converting film to digital has two extremes. A very low cost process involves using a camcorder and recording the projected film. The aspect ratios are typically not the same, the colors that can be depicted by both film and video are different, and various environmental conditions cannot be controlled while recording, so the information loss can be great. A much more expensive process involves using a high-resolution digital film scanner that scans every frame of the film and stores the result as a digital video sequence.

Synthesis/Transformation

Video is typically captured and edited with software that cuts it apart, splices it together and adds a small amount of titling. Many of the companies that produce hardware for digitizing video also create video manipulation software, such as Avid® and Media 100 Inc. Other companies have created the software to perform video editing that can work with any digitized video sample. Most of these work with the Quicktime format that allows for frame level access of the video regardless of the file storage and compression method. Apple Computer Inc., Adobe Systems Inc., and others have products that work with Quicktime video (Thompson, 2000).

Because of the interlaced nature of NTSC video, it is difficult to present an image that looks *clean* when presented on the progressively scanned medium of film or non-interlaced monitors. There are two de-interlacing techniques from which to choose. One is to blend the two fields together, which preserves motion better and produces sharper images in some cases. However, individual frames are composed of a composite of two fields, so a paused movie will generally produce some sort of motion blur / double image in areas of motion. The other option to de-interlace a movie is to throw away the even or odd fields. This voids the "motion blur" in still frames, but when the result is replayed normally the motion may not be as smooth.

Since the frame rates of NTSC video and film are different, film is typically transferred to video using a Telecine system, which uses the 3:2 pulldown method to convert 24 fps film to 30 fps (i.e., 60 fields per second) video. "3:2" refers to the method where the first film frame is "pulled down" into the first three video fields -- both fields of video frame one and the first field of video frame two. The second film frame is "pulled down" into the next two video fields -- the second field of video frame two and the first field of video frame three. The cycle repeats for additional frames.

The staggered nature of 3:2 pulldown creates a repeating pattern of five video frames. Three of the five video frames are whole-field frames (they contain two fields from one film frame) and two are split-frames (they contain fields from different film frames). If source files contain 3:2 pulldown, software such as Adobe® AfterEffects® can remove the 3:2 pulldown and then render the movie back to the original 24 fps.

The input and output devices for video may use different color spaces. NTSC video devices work in the luminance and chrominance (YIQ, YUV, YPbPr, or YCbCr) color space whereas computer monitors work with RGB. The luminance/chrominance color space is a transformed RGB color space. Since the human eye perceives luminance (brightness of an image) as the most important feature in an image, the creators of the NTSC standard dedicated more bandwidth to the luminance component. The chrominance component is thus the color (hue) that is added to the luminance. The hardware and software being used usually do the transformation between the two color spaces.

Animation

Storage/Display/Output

Typically animation refers to artistically created image sequences. Animation ranges in complexity from small black and white line drawings printed on the bottom corner of a flip-book to feature length 3-D animated films such as "A Bug's Life". As a media object, animation is basically the same as video, except that it is "generated" in some way rather than "captured" by a camera. The production techniques and production tools used in generation divide animation into two separate categories. The two categories are cel animation and sprite animation. Basically, cel animation output is the same as video, where each frame of the animation is a complete image. The color and shapes change from frame to frame, and motion is created by showing the frames at a given rate. For sprite animation, unique objects called sprites are created and placed in a background animation scene. The background stays the same in most frames, but the sprites can be transformed in a variety of ways, such as changing shape, position, color or a combination. Depending on the tools, a sprite can be a single 2-D raster graphic, a sequence of 2-D raster graphics, a 2-D vector graphic, or even a 3-D geometric object.

Artists have been drawing cel-based animations since the beginning of motion pictures. To animate a man waving his arm, the production process for cel-based animation is to first print a background on paper. The artist draws the entire character's figure except the moving arm on a thin transparent plastic sheet called a cel. The arm is drawn on a separate cel. The drawings are layered with the background on the bottom, then the character's body cel, and finally the arm cel at the top. The entire composite image is then recorded onto film. Next, a new cel with the arm slightly moved is drawn, and replaces the earlier arm cel in the composition. The new composition is recorded as the next frame in the animation and so forth. Artists use the previous cel as a base reference for the following cel. Commonly the artist will lay a blank cel over the previous cel and then begin drawing the slightly moved version. This technique is referred to as onion-skinning. Even with the cel and onion-skinning techniques, the process is extremely time consuming even for professional artists.

The number of frames drawn for one second of animation varies. Due to the amount of effort it takes to produce a cel-based animation, many artists limit the number of frames too as little as 6 per second. However, large producers of animation that require high quality results, such as feature length Disney animations, use 24 frames per second.

Computer generated animation has helped revolutionize the production of high quality animations. Artists can now use mathematically depicted objects to help produce animation. Computer based techniques are used to composite graphics together, create graphic movement from one location in the scene to another, and create in-between frames to create smoother motion. For example, if the artist places an object in the scene at location $<x_1,y_1>$ then 60 frames later requires the object to be at location $<x_2,y_2>$, an algorithm can be used to create the other 58 frames moving the object from point 1 to point 2 smoothly through what is commonly known as tweening.

Artists can also use full 3-D objects to produce animation. The systems used to work with 3-D graphics have a multitude of options for creating animation, though, the result is still a sequence of rendered 2-D image frames.

Acquisition/Input

If animation were done using traditional means with paper and transparent cels, the animation would need to be digitized using the 2-D graphics acquisition techniques described earlier. Software such as SoftImage® Toonz has been developed specifically to support scanning each cel with a scanner, registering the cels to a common origin, and performing cleanup such as handling shadows cast on the back cel.

Synthesis/Transformation

The tools used to support computer-based animation are typically separated into categories that allow an artist to work the way he/she feels most comfortable. Software such as The Animation Stand works like traditional cel based animation. Backgrounds are drawn and layers (virtual cels) allow cel objects to be drawn. Features are typically added to speed up production, such as multiplane camera control, automatic cel painting, 3-D shading, and audio editing. Traditional 2-D graphics tools can also be applied to cel animation, but, they lack support for the traditional animation workflow found in the more specialized tools.

Further specialized cel based tools exist that allow the artist to paint directly on traditional video frames. Creating special effects such as laser blasts, lightning strikes, and performing rotoscope operations working frame by frame are laborious. Software such as Commotion from Puffin Designs allows the artist to draw on the video as it is playing at a selected speed. While the video is in slow motion, and artist can paint a single brush stroke that will look like a laser blast. without the artist having to edit the video frame by frame.

Sprite animation tools came into increased use with the advent of the computer-based tools, particularly through the use of algorithms that perform tweening. Programmers have enhanced sprite based software by adding the capability to "tween" more than just sprite locations. Sprites can change location, shape, color, can be image sequences, and can even be special effects such as particles and light flares. Two of the largest software packages for creating sprite animations are Adobe® AfterEffects® and Macromedia Director. However, as the focus on delivery changes from video to the web, newer packages that focus on vector sprites, such as Macromedia Flash, are gaining popularity.

Whether working with cel based animation tools or sprite based animation tools, the input devices are the same as those for 2-D graphics. Typically an artist uses a pointing device, such as a mouse or graphics tablet, and the computer keyboard. Typically most 3-D graphics tools allow the artist to manipulate parts of a scene over time, which produces a sequence of rendered frames as animation. Products such as Strata Studio 3D are used to animate strutting peacocks for NBC commercials. They provide timelines for placing events, like camera movement or object movement.

Audio

Storage/Display/Output

Audio media objects are completely different from the aforementioned media types. Audio also works on a different sense, hearing, than those objects oriented toward sight. In some ways there is much more variety with audio input and output than with other objects. For example, there are noticeable differences between spending $30, $300, or $3000 for a pair of external speakers, with almost continuous quality differences at points in between. Similar variety occurs with audio acquisition.

The four categories of audio are speech, music samples, musical instrument digital interface (MIDI) sequences, and synthesis. Speech and music samples are digitized analog audio, which we refer to as samples. Like graphics, audio samples also have resolution, and the higher the resolution, the higher the perceived quality of the digital sample. As technology has progressed, the level of resolution for audio has increased. Early digital audio samples used 8 bits of data and an 11 kHz sampling rate to capture audio. What this means is that a microphone first converted the audio wave to an electronic wave, then an analog to digital converter sampled the electronic waveform to produce an 8 bit data value (256 possible sample positions) 11,000 times per second. Higher resolution sampling is obtained by increasing the size of the data range, the number of samples per second, or both.

Humans can detect audio frequencies in the range of 20 Hz to 20 kHz. Low resolution audio samples can seem muddy or muffled (missing the high frequency tones), so it is necessary to sample audio at high resolutions. For example, 16 bit 44 kHz sampling is the standard for audio CDs. Many of today's analog to digital interface cards are capable of capturing 24 bits of data at a rate of 96 kHz. This digital representation of the waveform is sufficient to accurately reproduce any audio wave that humans can hear. Thus, whether the sound is acquired or generated in some fashion, this establishes the size of the data stream required to describe high quality audio.

Simple digital audio is typically just a type of data stream stored in either a proprietary file format or one of several standard formats such as the audio interchange file format (AIFF). MIDI sequences are data streams that combine audio samples and control events together to create digital sound and control external devices such as synthesizers. MIDI software packages tend to use proprietary formats for their control sequences, but there is a standard MIDI file format that allows some exchange of information. Audio synthesizers are specialized audio equipment. They can produce a wide range of audio waveforms programmatically from a variety of digital inputs, rather than just doing a digital to analog conversion to produce sound from a digital waveform. Examples include devices such as those from Symbolic Sound or as software programs like nord modular from clavia (Thompson, 2000). Sounds from synthesizers can be made into audio objects in two ways: they can be "played" and captured as audio samples to be used directly, or they can be used as a MIDI controlled sound bank.

Acquisition/Input

The equipment to capture audio waveforms varies. First the audio waveform is captured as an electric waveform. The most common way to do this is with a microphone, which has a diaphragm and a transducer to convert mechanical energy into electrical energy. This electrical waveform is then converted to a digital waveform using either an analog to digital computer interface card or some other analog to digital device such as a Digital Audio Tape (DAT) recorder. It may be necessary to capture audio waves from several microphones and other electrical waveform devices as one electric waveform. A mixing board is used to combine several electrical signals at the same time, allowing an operator to adjust each signal's amplitude separately to achieve a desired balance between signals. MIDI devices are typically capable of both input and output (I/O) so they can send events to the controlling computer as well as receive commands. Some MIDI software can automatically digitize an audio signal and produce comparable MIDI information.

Since audio quality is more important than video quality, high quality audio acquisition is a higher priority than high quality video acquisition. Thus, labs should provide high quality audio equipment, such as high quality microphones, audio compressors, limiters, DAT machines, and mixing equipment, all connected with quality cables.

Synthesis/Transformation

Once audio samples are captured, the multimedia author can use software tools such as BIAS Peak to cut them into smaller, more manageable pieces and perform clean up work. Often only portions of a full sample are needed thus the need for segmentation. Clean up can vary from making sure that all of the samples are at appropriate volumes, mixing together multiple samples, fading out samples, down-sampling (reducing the sample resolution or rate) or even removing ambient noise.

The manipulation of MIDI sequences requires software that can control other devices and play back samples at the same time. Newer MIDI packages such as Opcode Studio Vision Pro allows video to be synced to a MIDI audio sequence, allowing a musician to compose works specifically to match a particular video object. Once a MIDI piece is complete, all of the audio producing devices (synthesizers, computer audio cards, samplers) must feed their waveforms into a mixing board where the levels of each device can be adjusted. The output of the mixing board is then a single waveform that can be acquired as discussed earlier. It is also possible to capture the waveform of each device and then use multitrack editing software such as BIAS Deck to combine the samples together, allowing the musician to use a virtual mixer.

Multimedia

Storage/Display/Output

Bringing all of the aforementioned media together results in a multimedia object. A multimedia presentation results from depicting the components of a multimedia object simultaneously. Typically such a presentation presents only a display on a video monitor and audio on a set of speakers, because these are the only sensory output devices commonly available.

In order to distribute a multimedia presentation, media large enough to hold a combination of all the required data must be used. The options currently used today are magnetic media, optical media, and network transmission. DVD-ROM can hold 2.4 GB of data on each side of a disc, CD-ROM can hold 640 MB of data on one side of a disc, and magnetic media such as the Castlewood Orb can store 2 GB on each disk. Network transmission of a multimedia project is typically limited to the amount of space available on the presentation system and by the network transmission throughput.

The typical multimedia presentation contains user control or interaction points, so a data format must permit something more complex than simply streaming to output. There are basically three multimedia encapsulation specifications for which presentation viewers exist on most platforms. They are Portable Document Format (PDF), Quicktime and Synchronized Multimedia Integration Language (SMIL). There are other proprietary run-time engines for HyperText Markup Language (HTML) to encapsulate multimedia, but it requires that extensions be added as plug-ins to a web browser to manage video, sound, and animation.

Synthesis/Transformation

A plenitude of multimedia authoring software packages exists, but they tend to be platform specific. If the authoring environment is the Apple Computer Mac OS or Microsoft® Windows, the multimedia author can choose the software package that best fits the required output and needed options (Thompson, 2000). However, very few exist that allow authoring on computers running any variant of a Unix operating system.

The requirements of the multimedia project may not be met by any existing multimedia authoring software. In such a case, the developer has to program the end product in a computer language such as Sun's Java. Remember though that the cost of custom development adds to the overall cost of the multimedia system and also increases the risk to the project.

Chapter 4

Archival Quality, Organizational Techniques and Optimization of Digital Media

Following the WinWin Spiral Model to develop multimedia projects, we seek to first identify elements of risk and second, minimize the risks in these elements. All multimedia systems rely on high quality digital media to produce a high quality presentation. During multimedia project development, the authors must acquire and store media for use. A primary element of risk is that authors will capture and/or archive media objects in a manner that limits their quality. Hence, some element of project success can be attained simply by constructing a well-organized archive of high quality digital media assets. Regardless of whether the initial multimedia presentation is completed to the key stakeholders' vision, the archived digital media library is invaluable in supporting the refinement of the presentation and production of other related presentations.

The ideal archival library provides a developer access to high quality digital media elements on demand and in a form that is independent of any target presentation. The developer may need to transform a media element into a different format for use in a specific project, but should not have to reacquire the element. To create such a library, media elements must be routinely acquired at archival quality and systematically added to the library. The resultant library must be viewed as a primary result of the multimedia project so that the acquisition or synthesis of individual media objects always adds value to the overall media environment.

A digital media library archive requires reliable, fast long-term storage, high quality acquisition techniques, organizational standards that permit easy object access, and a wide range of format conversion techniques that allow objects to be converted from archival format to the format required in a particular presentation. Digital media objects can be large, so techniques such as feature reduction, file compression, and Society of Motion Picture and Television Engineers (SMPTE) time encoding are also important. Archival quality objects often can be used directly in the multimedia production. However, since archival media is stored at the highest quality level achievable, the objects will often need to be converted to lower resolution prior to use. Each delivery system for multimedia presentations has different requirements and limitations, so techniques to produce media elements optimized for each delivery system are crucial.

Current technology to provide large, fast, reliable, and easily accessible storage is limited to hard-disk technology. Most operating systems can now address terabytes worth of secondary storage. Hard-disks are fairly reliable with quoted mean time between failure (MTBF) ranging up to hundreds of thousands of hours. However, individual hard-disks are not fault-tolerant, so RAID technologies (hardware or software based) can be employed to provide fault-tolerances, with various RAID levels indicating the complexity of the scheme involved. Level 1, commonly known as *mirroring*, protects against loss of data by storing redundant data on two drives. Level 3 implements a scheme that uses multiple striped drives to store the user data and a dedicated parity drive for securing the data. Level 5 stripes user data across the physical disk array and implements a scheme for storing parity evenly across all drives. If a drive failure occurs using Level 3 or 5, the missing data can be reconstructed from the remaining active drives. Having fault-tolerant technology does not remove the need for backup systems. Backups should still be performed on a regular basis and due to the possible large archive that must be backed up, only very high-capacity tape backup systems are applicable (Thompson, 2000).

Since the media archive is invaluable as a resource, maintaining objects in storage for extended periods is important. Current options for providing large reliable and easily accessible long-term storage are limited to optical media. The current leader in this storage medium is the 5.2 GB digital versatile disk (DVD). There are two varieties of writable DVDs, DVD-R and DVD-RAM. DVD-R is only writable once, whereas DVD-RAM provides the developer with multiple read and write access. Current studies predict that under normal storage conditions in an office or home environment the lifetime for today's optical media will be 100 years or more (Stinson, 1995).

Perhaps the most important aspect of the digital media library is its organization. Once the media object is digitized at archival quality, and stored reliably, its usability hinges on whether another author can locate it and use it; if not, the object will more than likely be reacquired. A traditional organizational scheme is simply to use a hierarchical computer file system style of organization. While the hierarchical file system approach provides hierarchical organization, it provides little or no meaningful information about the origin, format, usage, costs, or content of the media. Adding this metadata can significantly enhance the usability of the data. Companies such as Canto have created media specific databases that extend the organization of media assets to include this information (Thompson, 2000).

Regardless which organizational method the project uses, archival media must be stored in accessible file formats. For historical reasons, many different file formats for each media exist. As technology progresses, more sophisticated data types require creating new file formats. Data files are structured according to specific format conventions, which are (or should be) open and well-documented. Designers of an archive must choose a format for the objects stored in the library that is easily accessible and amenable for use with common multimedia development tools. Certain formats may provide more data flexibility, but may be supported only in a few, proprietary authoring environments.

The amount of storage digital media requires can be large; therefore, data compression is typically used to reduce the physical size of each object. Algorithms to compress and decompress data are often called codecs. Compression algorithms encode an object into a more compact representation conveying the same or at least similar information. A decompression algorithm then reconstructs the original (or similar) object from the compressed file. Codecs can be divided into two categories: *symmetric* and *asymmetric*. A symmetric codec requires approximately the same amount of work to compress data as it does to decompress data. Asymmetric codecs require substantially more work to either compress or decompress data. Compression schemes can also be classified as: *lossless* versus *lossy*. Lossless compression refers to algorithms in which the compression/decompression process produces the original information; in lossy compression/decompression some data is lost. Lossy techniques thus sacrifice quality in order to achieve higher compression ratios.

Each media type stores inherently different information. Therefore archival quality acquisition, file formats, compression, and delivery optimization techniques are different between media types. For some types, such as text, there are few variables, so the primary factors in archival are acquisition and storage. However, other media types, present a wide range of advanced acquisition, compression, and delivery optimization options.

Since the goal of an effective multimedia production is to keep an audience interested, delivery of a *smooth* output stream is an essential element in a final multimedia product. The idea is to collect and coordinate all the necessary media elements for presentation on the output devices. Delivery optimization techniques that minimize the time between user requests and information presentation are critical. Once a user requests information, the multimedia application must fetch all the required data elements, perform any necessary transformations such as decompression or format conversion, then present the entire collection of multimedia coordinating the frames of various media types. Wait time is dominated by the time it takes to fetch the data and perform any transformations; time required for event coordination can be assumed minimal. To minimize the wait time, the developer must match storage and compression techniques to the speeds of fetch and decompression, keeping in mind requirements on the quality of the media elements.

Text media consist primarily of character strings, along with metadata that may specify text formatting such as font, font size, weight, character spacing, and placement. There are several common formats used for text. The simplest is as an ASCII character data stream, usually referred to as a text file, which contains no metadata. Other file formats can be used to add metadata, such as hypertext markup language (HTML) and rich text formatted (RTF). As the main format for all web pages the well-publicized HTML format provides the longevity and rich set of tools needed by an archive. The RTF format is also well specified and has already shown that it can withstand time. Most word processors can read and write RTF documents. All multimedia-authoring environments work directly with some form of text media. The environment the author works in determines how the final project stores and delivers text. Whether or not the media is compressed for delivery is dependent upon the authoring environment; in many cases due to their relatively small size text objects are not compressed. Lossy compression is pointless for text data. Many authoring environments provide text data compression natively, but for archival data storage a public compression format (such as Gzip, bzip2, Stuffit, and PKZIP) is preferred.

The ideal format for archival quality image storage is a lossless compressed file format. One of the most encompassing file formats is the tagged image file format (TIFF), which is supported by most multimedia and graphics software, provides RLE and LZW lossless file compression, supports multiple color models and multiple color depths, supports alpha channels and multiple images per file, and can handle large resolutions. The TIFF format is also an open specification with implementation source code and computer utilities freely available.

The resolution to store archival quality raster graphics must match or exceed the highest quality input or output resolution. With the knowledge that film is one of the highest resolution media, the KODAK PhotoCD format resolutions should be high enough to encompass most 2-D raster graphics. The PhotoCD format provides multiple resolutions that are based on the 2:3 aspect ratio of film with the highest quality level storing 4096 pixels by 6144 pixels and the standard quality level storing 2048 pixels by 3072 pixels. Archiving all graphics at the same size provides consistency and predictability of the archival media. Synthesized raster graphics should either be stored at the resolution they are created or at the highest resolution needed for print, typically similar to resolutions mentioned above.

It is obvious that most archival quality raster graphics can not be used directly in the final multimedia project due to their size. The graphics must be transformed to fit the multimedia production. A typical workflow to insert a graphic from the archive into the media production is as follows:

1. crop the image to size;
2. resample the image to the desired display resolution;
3. use sharpening and gamma correction to improve the display;
4. determine the quality of color needed and convert accordingly;
5. determine if the graphic could be converted to a vector graphic;
6. choose the presentation file format and compression.

Cropping an image is the process of removing unwanted portions. In page layout software used to publish printed material, images are rarely cropped. Instead the viewable portion of the image may be resized, but all of the original information exists. Since we need to reduce the size of the file that will be transmitted during the multimedia presentation, unseen portions of the original image must be physically removed.

If a graphic is larger than what is needed for display in the multimedia presentation, resample the image to reduce its size. The multimedia presentation output device (computer monitors and graphics cards) predetermines display resolutions. Traditionally, computer monitors are listed as

72, 75, or 100 dots per inch, thus multimedia producers usually work in terms of the number of pixels to display not inches or centimeters. The entire presentation display area on the video monitor is deemed the canvas, which can vary in size from 640 x 480 pixels to 1600 x 1200 pixels. Knowing how large a graphic should be with respect to the size of the canvas should give a rough estimate to the size in pixels of the optimized graphic. The smaller the graphic, the smaller the size of the file and the more responsive the multimedia production can be. Since graphics are two dimensional, as the size increases linearly, the file size increases by a power of 2; thus a reduction in the image size by 75% reduces the file size by 56.25%. Using this fact, the author can store the image at a size slightly smaller than needed for the multimedia presentation and have the authoring environment scale the image as needed. The tradeoff may be a slight quality loss.

Most algorithms that resample an image average the colors of neighboring pixels. This can cause an image to be slightly blurred. Algorithms that increase the contrast of adjacent pixels sharpen resampled images. Most raster graphics image editors name these filters Sharpen, Sharpen edges, and Unsharp masking. After sharpening an image, perform any color corrections and decide on final gamma levels. Depending on the target display, determine what the correct gamma should be. Images that look perfect on a Microsoft® Windows® computer will seem washed out on MacOS computer; images that look perfect on the MacOS will appear too dark on a Windows® computer. Each image, depending on the range of colors, should be corrected to provide average results.

Limiting the total number of colors to as few needed for a suitable quality graphic can substantially reduce the file size. Also, some multimedia authoring environments limit the color lookup tables used for image display. Algorithms such as dithering (approximate a color with a mixture of other colors) or replacing pixel color with the closest matching available color can reduce the number of colors needed for an image. If a small indexed color table (less than 64 colors) can store all of the colors represented and only the indexes to the colors are stored for each pixel, the file size can be reduced. Since smooth gradients, such as those produced in many photographs, require a large number of colors for display they cannot be color reduced without a large amount of data loss. If the image can be reduced to a very small number of colors (less than 32 colors) without dithering, it may be beneficial to see if it can be converted to a more compact vector graphic. Reducing the number of colors in an image narrows the file formats that must be used, since some formats do not support color lookup tables.

There are primarily five major file formats used to store graphics placed into multimedia projects. They are the Macintosh picture (PICT), PC paintbrush file format (PCX), graphics interchange format (GIF), portable network graphic format (PNG), and JPEG file interchange format (JFIF/JPEG). The first two, PICT and PCX, get their notoriety by being the main formats used by the two popular operating systems MacOS and Windows®. Many of the authoring packages use these formats for internal and external storage. Both formats support a lossless run length encoded (RLE) compression; but only the PICT format supports JPEG compression. A GIF file only supports indexed color images with color tables with 256 or fewer colors. It uses a lossless LZW algorithm for compression and decompression that is very fast. An extension to this format was added in 1989 to use Control Extension data that allows both textual and bitmap-based graphical data to be displayed, overlaid and deleted. This adds animation capabilities and supports use of a rendering technique named interlacing (sometimes referred to as progressive display). Although different than NTSC interlacing, it overlays progressively finer resolution images to give the viewer immediate feedback. The PNG format supports both indexed color images and up to 48 bit color images. It uses an efficient variant on the lossless LZ77 compression algorithm and supports progressive display. PNG can also store some color matching information such as chromaticity of the RGB data or stored gamma values. The last format JFIF, typically called JPEG due to its compression algorithm, uses a lossy algorithm to compress continuous tone images with ratios up to 40:1, although ratios of 20:1 have a much smaller amount of data loss. The JPEG format supports progressive display but should not be used for images with a small number of colors (use either PNG or GIF).

The JPEG compression algorithm in JFIF files is such that an end user can "tune" the quality of the encoder using a parameter sometimes called a *quality setting* or a *Q factor*. The algorithm divides the image into blocks of pixels to remove redundant image data. Companies such as Adobe® and BoxTop Software, Inc. improve on the JPEG algorithm by filtering the image to remove hard to compress pixels. BoxTop Software's ProJPEG even adds variable compression that can compress the image's background information higher than the foreground (in focus) information.

Since 3-D visually depicted panoramas are basically a set of 2-D graphics, very similar techniques and methodologies apply to acquisition, archiving, and optimization. It may be useful to archive all of the original imagery used to produce the visually depicted panoramas for use as 2-D imagery. The two primary differences between 3-D panoramas and 2-D graphics are that special acquisition parameters must be met and the final optimized result is stored in a format native to the presentation system. Multiple images that will be *stitched* together into a single image panorama require that there be minimal lighting variation between frames, that the evenly spaced images be aligned vertically, and that the focal point of each image be directly at the center of rotation.

Because 3-D visually depicted objects are a sequence of frames and not a single image, the techniques and methodologies for acquisition, archiving, and optimization are very similar to digital video. In fact often acquisition of objects is done with a video camera. Just like panoramas, the final optimized result is stored in the format native to the presentation system.

Geometrically defined 3-D graphics are best stored in the format native to the application used to construct the object. However, as the webl3D consortium begins to finalize its format, X3D may become a viable 3-D archival file format. Since 3-D geometrically defined objects can contain surface maps as large as some 2-D graphics, any format can be large. The speed of rendering 3-D objects can also be very compute intensive and tax even high-end computers. Two different companies, MetaStream and Cycore, have developed software that allows 3-D geometrically constructed objects to be rendered as the information is available, similar to 2-D graphics progressive display. As more and more information is available, more of the finished 3-D scene is rendered. If the audience of the prospective 3-D geometric media does not have the computer hardware to support 3-D rendering, the author can render the image to 3-D visually depicted graphics, 2-D raster graphics, or with software like Vecta3D even 2-D vector graphics.

Full frame video and animated objects are extremely large. It may be infeasible to store video as a file within the archive library. However, videotape that already meets the need for storage is appropriate. To catalog the information stored on videotape, use the Society of Motion Picture and Television Engineers (SMPTE) time code along with tape cataloging information, such as tape name and number. SMPTE time codes use the designation of hours, minutes, seconds, and frames to mark start and stop points on tape. When the multimedia author needs a segment of video, acquisition, editing, and optimization can be done in a single step. By using the original videotape as the archival storage, generation loss, the process of copying video signals over and over, is avoided.

If video must be edited, having time coded tapes allows for using an offline (low-resolution) model of editing, which takes up far less disk space. The creative work can be done with smaller resolution clips. Then at the final production step, the high-resolution clips can be batch digitized from the archival tapes and assembled. The final production can then be transferred back to a new archival tape for storage.

Video is a visual medium similar to 2-D graphics, thus has color as an important variable to reproduce. Traditionally, video display systems such as televisions have used a luminance and chrominance color system. There are a few variations on this color system, such as YUV, YIQ, YCbCr, and YPbPr, depending on which standard is followed or what equipment is used. With the YUV standard, the Y value, known as luminance, is the intensity and saturation of each pixel value. Black and white televisions only use the luminance for display. The U and V signals are the chrominance values, or hue information. U values produce the colors ranging from green to magenta where as V values produce the colors of blue through yellow. Television broadcasters gave the luminance more importance, focusing more data (and more bandwidth) on its description. Consequently the bandwidth provides an NTSC resolution of 720 active pixels for each scanline with 486 active scanlines. A triple pair of numbers is used to describe how much information is provided by different equipment for the U and V signals. In high-end video equipment, 4:2:2, signals provide luminance on every pixel and U and V values on every other pixel per scanline. Most of the new digital video (DV) equipment uses a 4:1:1 signal, which provides luminance over the entire image but U and V values on every fourth pixel per scanline.

Even a 4:1:1 YUV video signal requires a large amount of storage space. Uncompressed, it requires 15 MB/s. With the right equipment, this is achievable. However, for a multimedia project, archiving video at this level is unreasonable. Just as with 2-D graphics, individual frames of video can be compressed to save storage space. If even a 5:1 compression ratio can be attained, 3 MB/sec is much more realistic for dealing with video. However, most compression requires enough processing time that the computer's processor is not fast enough to capture compressed digital video in real-time. Most video capture boards convert the analog signal to digital, convert the color space from luma/chroma to RGB and perform the compression in special hardware. The most used compression is motion-JPEG (MJPEG). Primarily this codec compresses each frame as a separate JPEG image. The new DV hardware, such as camcorders, uses a variation of the MJPEG format that compresses each field in the 4:1:1 YUV color space. This data can be transmitted from the DV devices directly to the computer hardware without further compression.

When synthesizing video, such as animation sequences, picking the proper video codec requires determining what the output device will be. If the final destination is for a video signal, then use a codec that matches that in the video hardware (such as MJPEG or DV). If the final destination is for multiple purposes use an uncompressed codec or lossless compressed codec (assuming sufficient storage space is available).

Videotape shelf life is known to be 10-30 years in controlled conditions. For extended storage of video, the Moving Picture Experts Group (MPEG) has developed a compression algorithm specially designed for video at high quality that allows video to be archived onto DVD media. The MPEG-2 codec specially designed for CCIR 601 video and high-quality audio achieves compression ratios from 8:1 to 30:1. Not only does it use compression technology like MJPEG within a frame (inter-frame) but also applies compression between frames (intra-frame). MPEG-2 allows the multimedia author to archive digital video at archival quality; but do to the nature of intra-frame compression, it is not suited as a video editing codec. Technologies such as DVD and digital cable television use MPEG-2 technology.

The size of video makes its delivery over limited bandwidth difficult. IP networks do not guarantee continuous information at a given rate; slow CD-ROMs may transfer as little as 350 KB/sec. Technologies called *streaming* and *fast-start* try to decrease the time delay between information download on a network and presentation. Streaming uses a protocol (RTSP) on top of IP networks to increase control over the information sent and received. The Fast-start technology uses standard protocols such as HTTP and orders the information in video files so that when enough information is downloaded the video begins playing before the entire file has completely downloaded. Even with these improved presentation technologies, the rate of information transfer is limited. Sophisticated compression algorithms must be used to limit the data load while maintaining reasonable quality. Since complicated compression algorithms take more processing time, the presentation system must have a processor capable of decompressing the video as it displays.

The authors must decide what to use as the minimum specifications (data transfer rate, video hardware acceleration, processor speed) for the presentation system. Once these decisions are made, decisions about which codec and the size of the resultant video can be made. Currently, there are four video frameworks for which video can be presented, as shown in Table 3. Each framework stores files in its own format and is compatible with certain operating systems. Frameworks such as Apple Computer's Quicktime™ support a variety of codecs for both audio and video. Developers created different codecs to meet different needs. As development progresses, quality increases, data rates decrease, but computer-processing tasks increase. Terran Interactive, Inc. has a software program dedicated to video and audio compression named Media Cleaner Pro. The software can compress to various frameworks using a variety of codecs and the company provides a large amount of information about different codecs (Terran Interactive, Inc., 2000). A good start for understanding the differences between codecs is with Ben Waggoner's comparison of many different codecs for use with multimedia video (Waggoner, 1998).

Video Framework	MacOS	Windows	Unix (some variant)
Apple Quicktime™	•	•	•
Real Systems	•	•	•
Microsoft Windows Media		•	
MPEG-1 or 2	•	•	•

Table 3. Video Framework Availability

Although today's codecs provide improved quality with lower data rates, the following workflow can provide the highest quality presentation video:

1. capture the original video at the highest quality possible;
2. de-interlace the video if needed;
3. reverse the Intelecine encoding if needed;
4. crop the video removing unneeded information;
5. scale (resample) the video to the appropriate size with square pixels keeping the pixel dimensions divisible by a factor of 4, 8, and 16;
6. apply noise reduction filters to remove hard to compress stray pixels;
7. restore black levels and increase contrast slightly;
8. choose an appropriate frame rate, a factor of the original such as 12 or 24 for film and 15 or 30 for video;
9. if the codec provides fine tuning, limit the data rate to known presentation system's usable data rate;
10. create multiple versions at different quality levels for different presentations;
11. compress audio as discussed in the audio section of this chapter.

The old adage garbage in – garbage out couldn't apply more to any other media than video. It is imperative that before trying to produce high quality multimedia video that the source video be as good as possible. Poor video will create even poorer multimedia presentation video. It is imperative to capture the best source material possible with the cleanest input signals. Noise introduced by cables and generation loss will be amplified during the optimization process.

Useless information such as the black bars needed to display wide screen formatted video on NTSC monitors should be cropped off of the video. This information will increase the file size but is irrelevant when working with multimedia video. Just like 2-D graphics, as the frame size is decreased, the file size is reduced even more. NTSC video also has a certain amount of information that is never seen around its borders with the standard television monitor. Knowing this, the author may wish to further crop the source video.

Computer monitors are different than typical video monitors since they display square pixels, not rectangular pixels. On NTSC monitors, 720 x 480 pixels represent the 4:3 aspect ratio, whereas a computer monitor uses 640 x 480 pixels. Thus the author should adjust the scale of the video to convert the rectangular pixels. The final resolution chosen by the author can play a huge factor in how much compression is performed. The resolution of the presentation video can be half the size of what is actually displayed to reduce the data rate. It is also important to keep in mind that most of the compression algorithms compress much higher if the resolution dimensions are divisible by 4, 8, and 16.

Color differences between different presentation systems require that multimedia presentation video may need adjustment from its original captured source. Adjusting the black level (what color is truly black), increasing contrast, and lowering the gamma level provides a more impressive final quality multimedia video.

The high number of frames per second needed to reduce flicker may not be achievable with lower data transfer rates. Also, lower-end presentation systems may not be able to decompress the information fast enough resulting in video that looks worse. The easiest way to reduce the data rate is to reduce the number of frames delivered per second. Make sure that the new frame rate is division of the original frame rate to keep motion as smooth as possible. For example, if the source material is video at 30 frames per second, the final video should be a subset at an equal division of 30; take every 5th frame to produce 6 frames per second, etc.

Many of the video codecs have options available for fine tuning similar to the Q-factor of the JPEG algorithm. These options allow the author to choose what amount and type of compression to perform in order to reach a final goal. Some of these options can be programmatically determined by just setting bounds such as what data rate the final movie should be.

The multimedia author may wish to provide multiple levels of the same video clip so that different presentation systems use different clips. Some of the authoring environments such as Quicktime allow the author to store multiple versions of a single video at several different quality levels. At presentation time, the video framework chooses which level of video to play determined by the system's specifications.

Synthesized animation is usually produced digitally, and should be stored in a production environment format so that further editing can be performed if necessary. To optimize the format for animation, it may be necessary to save the animation as video, then optimize as described in the previous section; however, several other options exist for sprite animations, which can significantly reduce the storage required and thus give more smooth playback.

Traditionally, audio file formats existed as elements of specific software packages. However, as operating systems matured, developers often added specific other general-purpose audio file formats. The most common formats in use today for multiple resolution audio samples are the Audio Interchange File Format (AIFF) from the MacOS, Waveform Audio File Format (WAV) from Windows®, and Sound Designer II (SDII) used with most DigiDesign audio hardware. Any of these would be appropriate to use as an archival format. Multimedia developers should probably use the one that is most compatible with their developmental environment.

Audio sample files can be quite large. For example, one minute of uncompressed CD quality stereo audio occupies approximately 10 MB. Archival quality audio should be stored at resolutions at or above CD quality audio (16 bit, 44.1 kHz), because anything less will introduces quality issues during editing. If storage size is a concern, make sure that audio captured from sources that used monaural inputs such as single microphones are not captured and stored as stereo files. Compression is limited for high quality audio, only a few lossless audio sample compression algorithms exist as utilities, such as Waves TrackPak (which can reduce a file size by 33% on average).

Most multimedia presentation systems include audio by importing audio files in formats such as WAV and AIFF. However, some audio environments are sophisticated enough that they require their own special software for inclusion and playback. The specialty software typically is provided as a plug-in to the multimedia authoring environment. The most common plug-in audio delivery systems are Quicktime, RealAudio, Shockwave and Beatnik.

Since archived audio often has data rates too large for most delivery systems and is typically intended to be used with a comparably sized video stream, the following workflow can be used to provide audio optimization.

1. Determine if a MIDI sequence would be more appropriate.
2. Trim all silence off the beginning and end.
3. Silence audio that should be silent (i.e. eliminate background noise).
4. Perform any necessary audio filtering.
5. Mix multiple audio tracks into one.
6. Reduce the file to monaural from stereo.
7. Normalize the audio samples.
8. Reduce the number of samples per second.
9. Re-filter to brighten the sound.
10. Reduce the number of bits per sample.
11. Choose an audio file compression technique based on the source material.

Audio descriptions based on MIDI and similar technologies, such as Beatnik from Beatnik, Inc., do not require as much storage as comparable digital audio samples, since they store a palette of sounds that can be used over and over to reduce the amount of information that must be stored.

Often, audio that is assembled using sequencing software can be saved into such a format and played back within the multimedia environment. The quality of the audio reproduced can be as good or better than a sampled version of the same material.

For audio samples, removing useless information reduces the file size. In uncompressed files, silence requires as much storage as non-silence. Use an audio editor to trim off all silence from the beginning and the end of the waveform. Compressed audio files can compress silence much higher than non-silence, so it is also important to use an audio editor to "zero" the sound in other silent areas. Audio editor plug-ins exist that can automatically silence audio under a threshold level.

Regardless of the audio equipment quality, some audio may need a level of adjustment. For speech samples, an audio compressor/limiter makes the sound level constant. The quality of poorly sampled sound can be improved by using filters to de-ess (remove "s" sounds), remove noise such as hums and fans, remove clicks and pops, and equalize sound levels.

If possible, mix all individual tracks into a single audio track. Each track requires approximately an equal amount of storage. It is convenient when producing multimedia to have underlying music tracks initially separate from the speech track. However, storing multiple tracks stores more information. For example, three tracks of audio requires three times the amount of storage as one mixed track of audio. Since stereo files are basically two track audio files where one track plays to the left channel and one track plays to the right channel, mixing the two tracks to monaural requires half the storage.

One of the most significant steps to provide quality audio is to bring the volume of all samples to a similar level. A process called audio normalization allows the developer to optimize the volume of a selection or an entire audio document so that it is at its maximum without clipping. The normalize function is very useful for boosting the loudness of material that was recorded at too low a level, or for making sure that the amplitude of each of the documents is uniform.

The number of audio samples per second is directly proportional to the size of the file. Thus, if an audio file is reduced from sampling at 44 kHz to 22 kHz, the audio sample file size is reduced by half. Quality loss from reduced sampling may be small compared to the substantial storage and data transfer rate savings. Remember that the human frequency range is not much higher than 20 kHz, and due to the dynamic range, voice-overs can often be reduced to as low as 11 kHz without significant quality loss. Note that reducing the audio sample rate may remove some of the higher frequencies, which makes the audio sample sound "flat". However, audio sample filters and equalizers can be used to amplify the higher frequencies, boost the sound and "brighten" the audio file.

As a final step in reducing file size prior to file compression, the developer may wish to reduce the number of bits used to store each sample. Converting a 16-bit sample to an 8-bit sample reduces the file size by 50%. However, recognize that the quality loss may be much greater than comparable file size reductions based on reducing the sample rate. A developer needs to be sure to evaluate each sample transformation for the right combination of size reductions.

Most audio file compressions are lossy compressions, which offer compression levels between

176:1 and 10:1, in exchange for loss of quality. The most popular compressors are IMA/ADPCM, MPEG II Layer III audio, Microlaw, QDesign, and Qualcomm Purevoice. Codecs with higher compression rates take more processing time to decompress, so compression must always be "engineered" to match other processing requirements and available processing power.

To allow final production, a developer typically must assume some type of base line presentation system that has specific data rates and processing speed for each media type. Systems that meet or exceed the minimum system should have no problems providing a smooth responsive multimedia delivery. The primary decisions that must be made are minimum canvas display size, minimum data transfer rates, minimum processor speeds, and any specially required hardware.

Given guidelines for these parameters, the developer can predict performance of a minimum system and engineer the media accordingly. However, it is always best to perform testing on a minimally configured system at various stages in development to avoid any surprises with the finished product.

For determining the minimum canvas display size and minimum data transfer rate, the information provided in Tables 4-6 can be helpful.

Screen Size	Typical Canvas Size
14”	640 x 480 pixels
15”	800 x 600 pixels
17”	1024 x 768 pixels
19”	1200 x 1024 pixels

Table 4. Canvas Display Sizes

	Data Transfer Rates	
CDROM Speed	Typical Sustained	Theoretical Maximum
2X	200 KB/sec	300 KB/sec
4X	400 KB/sec	600 KB/sec
8X	800 KB/sec	1200 KB/sec
16X	1600 KB/sec	2400 KB/sec
24X	2400 KB/sec	3600 KB/sec

Table 5. CDROM Transfer Rates

	Data Transfer Rates	
Network Connection	**Typical Sustained**	**Theoretical Maximum**
28.8 baud modem	2.8 KB/sec	3.51 KB/sec
56.6 baud modem	4.7 KB/sec	5.86 KB/sec
ISDN	6.3 or 12.5 KB/sec	7.81 or 15.63 KB/sec
ADSL modem	37.5 or 75 KB/sec	46.88 or 93.75 KB/sec
10 Base T Networking	1 MB/sec	1.25 MB/sec
100 Base T Networking	10 MB/sec	12.5 MB/sec

Table 6. Network Transfer Rates

System processor speed provides guidance on what audio and video codecs are appropriate. However, the developer should also refer to the documentation for specific codecs to see if there are other concerns. Special hardware may be needed in order to decompress audio/video in real time (such as an MPEG decompressor card).

Chapter 5

Bringing it all together, an example

Project Background

The example discussed below involves multimedia development for several specific "artifacts" developed for a petroleum reservoir characterization project. The objective of the overall project was twofold: to develop petroleum reservoir characteristic and modeling techniques, and to develop a cross-trained reservoir characterization team of more integrated earth scientists. The research team was formed from members with expertise in geology, geophysics, applied mathematics, petroleum engineering, and computer visualization. Thus, the purposes of the multimedia development included the illustration of scientific results, promoting cross training for project participants, and promoting the integrated interdisciplinary approach to the "public" at large.

The petroleum reservoir selected for the initial characterization project was the Rabbit Hills oil field in north central Montana. The researchers started with the assumption that maximum understanding of reservoir behavior would be gained through integration of 3-dimensional seismic data, geologic and engineering models, mathematical models that accommodate variations of physical scale and relative emphasis of data, and 3-dimensional visualization of the integrated model set. As noted above, one of the key artifacts to be produced in the project was a multimedia CD-ROM that could be used to promote petroleum science to the public. The decision was made to target this at an elementary/secondary education level, explaining how the project scientists worked together and what results were obtained. What follows is a description of how multimedia development processes were used on one portion of the development of the multimedia CD-ROM.

Chosen Subsection

In what follows, I focus on a particular subsection of the multimedia project in which the content includes a variety of media types, organized in terms of the seven inter-linked *content-components* shown in Figure 6.

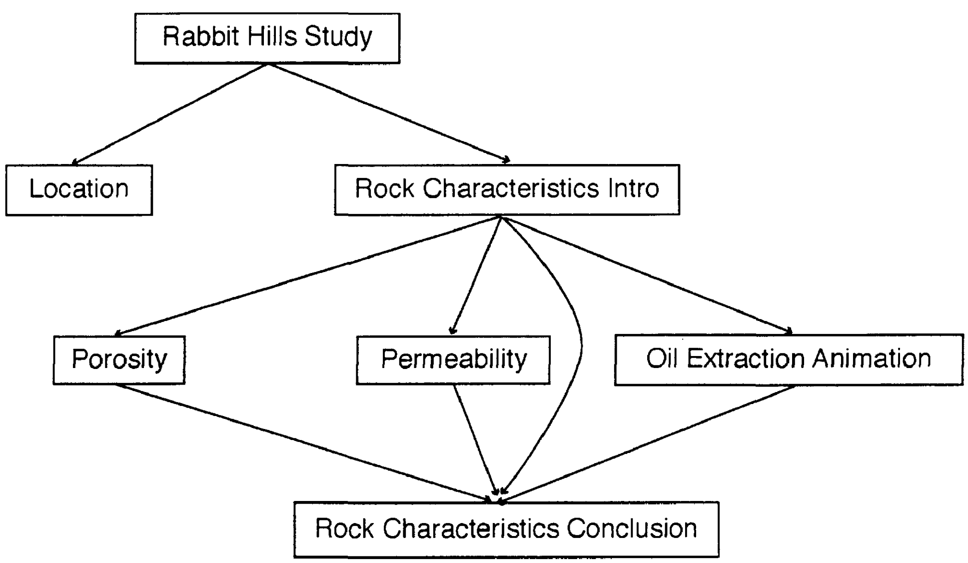


Figure 6. Subsection Organization

The development is described using Boehm's WinWin multimedia development process, I start the description with the data collection and organization steps.

Data Collection and Organization

The key stakeholders of the project provided much of the media as 35mm slides and letter-sized diagrams. I scanned the slides using a 35mm slide scanner at 1200 dpi (its maximum optical resolution). I used a Hewlett-Packard flatbed scanner at 600 dpi to scan the hard copy diagrams. After cropping and resizing all of the images to 2048 x 3072, I did color correction using Adobe® Photoshop®. Finally, I saved the results as "base imagery", giving each image a name that fit its pictorial scene and placing the whole collection in a file system directory named "images".

Other members of the multimedia team synthesized additional imagery, in the form of backgrounds and buttons. I collected these into the "images" directory. Several video clips were also to be used. These had been captured using a Hi8 camcorder with an external Lavalier microphone. After the specific in/out SMPTE times of each clip were decided I initiated the process of converting the original video into digital objects. I first used digitizing software to capture each clip at full screen (640 x 480 pixels) 30 fps, with each clip saved as a separate movie file. The digitizer was a Radius Video Vision Studio that stored the movie clips with a proprietary codec as a variety of MJPEG. I stored the first generation digitized video objects in a directory named "video".

With the target audience specified as children, the team decided to demonstrate several complex concepts as animation. One of the graphic designers took charge and used Macromedia Director to develop six animations describing oil extraction and seismography. In order to provide a more widespread animation format, I converted the animations to Macromedia Flash. I stored the completed animation files into a folder named "animations" within the archive.

Sound production started with the whole team writing the "voice-over" script. I then acted as the director/producer, directing various team members to record the scripted voice-over segments. The audio clips were captured on DAT tape, which the team's audio specialist then digitized at 44.1 kHz, 16 bit using Digidesign Audio Media hardware. The digital audio clips were stored separately as named files. I collected all of these files into a directory named "voice overs", and added this to the archive. The audio specialist scored original background music using Opcode's Studio Vision and gave me the completed work as a Vision file. I added this to the archive in a directory named "music". Lastly, I collected small amounts of text from printed materials into a text file, saved in the archive in a directory named "text".

The resulting archive, shown in Figure 7, contained the majority of elements needed for this iteration of the multimedia project. This collection of media objects was suitable to permit the production of a number of multimedia artifacts. Also, it was flexible enough so that any additional iteration of development would permit the addition of more media elements and more types to the library.

animation	–	folder
primary extraction	–	folder
primary flash	160 K	Macromedia Flash movie
primary.pct	404 K	Photoshop® PICT file
primary.psd	792 K	Photoshop® file
images	–	folder
background.pict	496 K	Photoshop® PICT file
braided-model.tif	1.2 MB	Photoshop® TIFF file
core002.tif	7.2 MB	Photoshop® TIFF file
Oilfield015.tif	11.1 MB	Photoshop® TIFF file
Oilfield016.tif	9.7 MB	Photoshop® TIFF file
oilsponge.pict	84 K	Photoshop® PICT file
perm-button.pict	12 K	Photoshop® PICT file
porosity-button.pict	12 K	Photoshop® PICT file
porosity.pict	228 K	Photoshop® PICT file
primary-extract.pict	92 K	Photoshop® PICT file
project study diagram.tiff	224 K	Photoshop® TIFF file
rock-char-banner.pict	48 K	Photoshop® PICT file
thinsact slide.tif	9.3 MB	Photoshop® TIFF file
movies	–	folder
porter-perm.mov	5.9 MB	MoviePlayer movie
porter-porosity.mov	82.3 MB	MoviePleyer movie
music	–	folder
rain dance.aiff	58.1 MB	MoviePlayer document
rain dance.svp	76 K	
seismic.aiff	9.1 MB	MoviePlayer document
text	–	folder
all.txt	4 K	Microsoft Word text document
voice overs	–	folder
2 important rock characteristic	424 K	MoviePlayer document
and flow is permeability	428 K	MoviePlayer document
find out more about 2 character	348 K	MoviePlayer document
goto kp on permeability	712 K	MoviePlayer document
goto kp on porosity	716 K	MoviePlayer document
known as porosity	352 K	MoviePlayer document

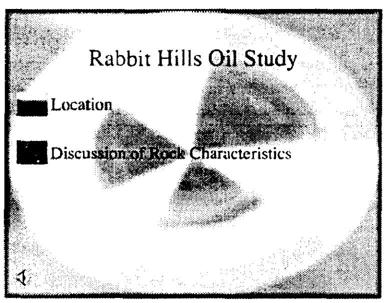
Figure 7. Archive Library

Design Decisions

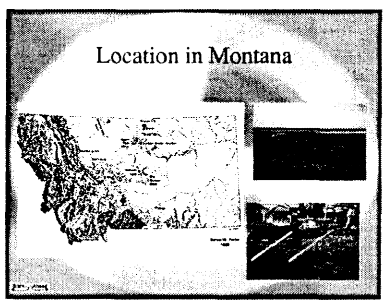
The next key decision required prior to the production of any specific artifact was that of the minimum target presentation system capabilities. Since the project plan called for both CD-ROM and Internet delivery of material, I needed to consider the requirements for both. Since many education institutions have lower end computers, I set the canvas display size at 800 pixels by 600 pixels. Similarly, low-end PCs imply a minimum CD-ROM transfer rate of 4X, which limits the sustained data transfer rate to 400 KB/second. Assuming that a minimum Internet connection is an ISDN connection, the on-line data transfer rate is limited to 12.5 KB/second. Planning to include (most) computers purchased within the last four years set the minimum processor speeds to 233 MHz for a Windows system (Intel® Pentium®) and 200/300 MHz for a Macintosh (PowerPC 604e/603). For Internet distribution, I planned to use standard HTML as the distribution format, using Netscape Navigator 4.x or Internet Explorer 5.x with appropriate plug-ins as the presentation software. For CD-ROM distribution, I chose to use Macromedia Director as the development platform. This software produces executables that run on each platform, masking specific formatting details from the end user.

Design the initial prototype

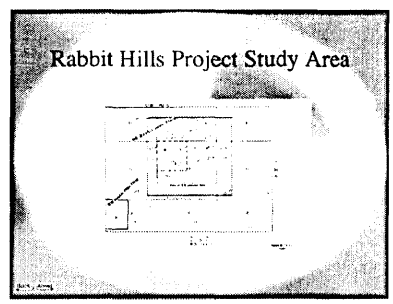
Using the archival quality materials, I constructed rough initial designs of each content-component using various rapid prototyping tools. Most HTML tools are restrictive in the file formats they can accept, which in turn forces a developer to perform a lot of file manipulation and format conversion. Macromedia Director, due to its steep learning curve, can be slow to prototype with, so I chose Microsoft® PowerPoint® as the tool for prototyping initial content design. The result is illustrated in the following 8 images.



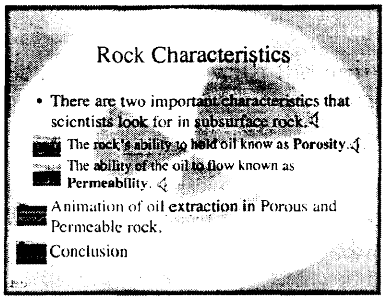
Component 1



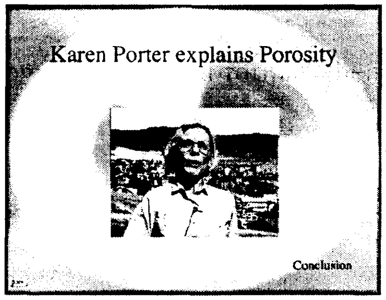
Component 2



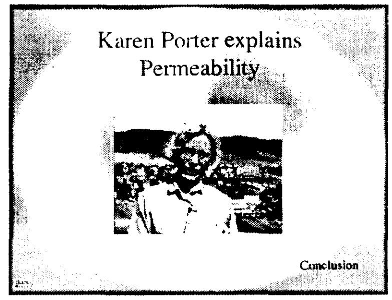
Component 3



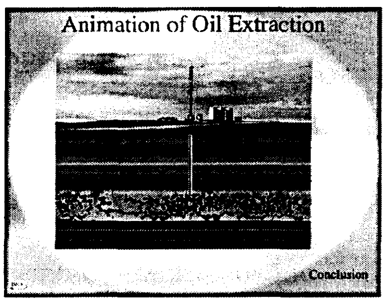
Component 4



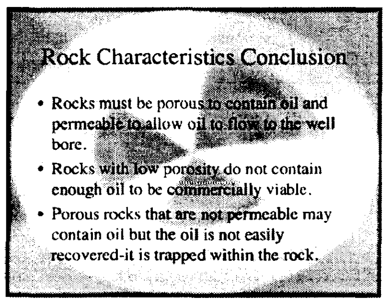
Component 5



Component 6



Component 7



Component 8

Figure 8. Prototyped Subsection Snapshots

A minimal amount of time was spent on designing this initial prototype, which represents a rough *storyboard* for the presentation. Acquisition and data conversion costs were also minimized. The result was sufficiently illustrative and detailed that it allowed the key stakeholders to review the prototype and provide feedback, which allowed me to plan the next development cycle, recognizing that further development could include re-engineering the flow, revising the individual components, or acquiring new material. Note that because audio elements are difficult to show within PowerPoint as individual slides, I represented them as speaker icons.

The idea was to use this form of prototype to get customer feedback. Once development satisfied the key stakeholders and they made the decision to develop the product, I then had to actually create each content-component with its optimized media, then link them together via sequential and hyper-linked events to form a complete object customized for the particular required delivery systems.

Prototype accepted; begin optimization spiral.

I next followed a sequential basic development process, optimizing the content objects one by one. Since many of the optimization steps had to be repeated several times for different elements, I show only the optimization process for the first unique element within the project. As the elements were optimized, I placed them into an optimized library and inserted them into the two multimedia development systems (Adobe® GoLive Studio® and Macromedia Director).

First Components's Unique Elements: Text, 2-D Graphics, Audio (music)

The textual elements in a presentation are typically so small that it is immaterial to attempt to optimize their format. Thus most of my attention was on imagery and audio.

The first image to optimize was the background image. This was an artistically created image. It needed no cropping, and was already at the desired 800 by 600 display resolution. It was developed on a Macintosh, so I had recognized that the artist's intended color might be too dark when displayed on other platforms. I increased the gamma by 1.5 to lighten the image, using the Levels adjustment in Adobe® Photoshop®. Since this image had a large gradient nature, the most appropriate file compression was JPEG. For the web image, I saved the image as a JPEG

(JFIF) file. For the CD-ROM image, since the version of Macromedia Director I was using couldn't read JPEG or GIF files, I had to save the image as a PICT file with JPEG compression enabled. I experimented with the JPEG compression to find the lowest Q-factor value (highest compression) that would retain reasonable image quality. The archival image took 664 KB of storage. Choosing a Q-factor of 10 for the web image resulted in a file size of 8.8 KB, whereas a Q-factor of "low" for the PICT produced a 28.2 KB file.

The two little icons on the first component are images from the archive. The first step was to resample each image to the appropriate icon size. After experimentation, I set the size for each icon to 59 pixels wide and 44 pixels high. I resampled the image to this size, applied a small amount of sharpening and adjusted the gamma slightly. Each icon had smooth gradients similar to that of the background image; however, due to the small number of pixels stored, a GIF file limited to 32 perceptually chosen colors produced a better image than an equal sized JPEG. The original image file size was 9.7 MB for each icon. The new icon size was less than 1 KB as a GIF for the web, and 16 KB as a PICT for use with Director.

The last piece of media on the first component was a music track that was to be played throughout the presentation. The original was stored in the archival library as a 5 minute 45 second, 58.1 MB AIFF file. File size made this completely inappropriate for delivery over the Internet and questionable for low-end CD-ROM players. Since I had the original music sequence, I was able to create a general MIDI file within the Quicktime media environment. This MIDI file was only 1.5 KB, yet retained fair audio quality. This was acceptable for web delivery, but since the data rate could be much higher with a CD-ROM I created a higher quality audio component for the CD, a new AIFF from the original music file. I used the Quicktime media environment to store the final audio file in Director. I used the QDesign music codec to compress the 44 kHz 16 bit stereo audio file at a data rate of 6 KB/sec, yielding a 1.9 MB audio file.

At completion, the total size of the first presentation page was 13.5 KB, which could download in a little over 1 second with an ISDN line. The size of the first Macromedia Director content-component is less than 2 MB, which could display from a 4X CD-ROM in less than 1 second.

Second Component's Unique Elements: 2-D Graphics

I produced the five new images in Component Two following the same image optimization process, though the compressions for each object were different. Since the large Montana map is primarily a black and white image without continuous gradient tones, I first reduced the number of colors used to portray it to four distinct grays, then saved the image as a GIF file. The two color pictures needed to be reduced in size (reducing the number of pixels), then compressed as JPEG images. Finally, I had to prepare the small back and forward button icons. I decided that each could be an object with a transparent background, so just the word and arrow would be displayed. This required me to use the GIF or PNG image file formats which both support transparency. PNG-24 (the 24 bit version) even supports a transparency mask known as an alpha channel, so I chose the clean look of the PNG-24 alpha channel over the smaller size of the GIF transparency. For the CD-ROM, I saved the image with its alpha channel as a PICT file, which also stores transparency information.

Third Component's Unique Elements: 2-D Vector Graphics

Within Component Three, the graphic showing the location of the project conformed nicely to a vector graphic. Since typical web delivery systems provide direct support for vector graphics via Scalable Vector Graphics (SVG), Macromedia Flash via a plug-in, and Quicktime via a plug-in, I converted the original raster graphic to vector. Director supported only vector graphics in Quicktime format, so I stored the final result in that format. Using vector graphics reduced the file size and allowed for device dependent resolution, i.e. permitting the graphics to be scaled with no quality loss. To convert this graphic, I imported the raster graphic into the vector drawing program Adobe® Illustrator® and then traced it. Once completed, I saved the new version as an archival quality diagram in the library and exported as a Macromedia Flash document. Using Apple's Quicktime Player, I converted the Flash document to the Quicktime vector graphics format. I then put this image directly into both presentations.

Fourth Component's Unique Elements: Audio (voice)

Within Component Four, the new elements that had yet to be optimized were the audio voice-overs. These audio samples were recorded within a sound studio, so the original quality was exceptional. The first task I performed was to trim off excess silence from both the beginning and end of each sequence. The second step was to go through and silence (literally zero all of the audio data) in the small gaps between each phrase. The final step before compressing each file was to normalize the audio to 100% to bring the audio sound level up to its peak.

For very high vocal compression, Qualcomm's PureVoice codec produced the smallest file sizes; however, the quality was quite poor, similar to that of poor analog cell phone reception. Instead, I used the QDesign Music 2 codec and compressed at 22 kHz 2.5 KB/sec. This doubled the file size, but also doubled the quality of the audio. Since the rest of the page's elements were quite small, it was reasonable to allow the audio to be slightly larger. The total size of 18 seconds of audio was 48 KB. For the CD-ROM, file size wasn't much of an issue. Keeping the quality very high, I saved the samples as AIFF files with IMA 4:1 compression, which Director can read natively. The resultant file size was 440 KB.

Fifth Component's Unique Elements: Video

Within Component Five, I needed to optimize the video elements. I captured each of the video samples stored within the archive with a Radius Video Vision Studio analog-to-digital capture board. This board converted the video to an RGB full screen video sample with 44.1 kHz stereo audio at a sample rate of 4.5 MB/second. Using Media100's Media Cleaner Pro, I adjusted the brightness, contrast, and black point, cropped the border of each video by 10%, re-sampled the image to the final resolution, and performed pixel noise reduction for each clip as the video was compressed. For web delivery, I set the resolution to 240 x 180 pixels at 10 frames/second, chose the Qualcomm Purevoice codec with 16 bit 11 kHz at 1.2 KB/sec for the audio, and used the Sorenson video codec set at 8 KB/sec with a high-quality first frame. The resultant clip was 8.8 KB/sec. For CD-ROM, faster transfer rates allowed the quality of video and audio to be improved. I set the size to 320 by 240 pixels at 15 frames per second, chose the IMA 4:1 compression 16 bit 22 kHz for the audio, and used the Sorenson video codec set at 70 KB/sec with a high-quality first frame.

Seventh Screen's Unique Elements: Animation

The last primary element I had to add to each presentation was the animation of Component Seven. I started with the animation in its original format, then exported it as a Quicktime sprite movie, which produced a 40 KB 640 x 480 animation. This sprite movie gave the highest quality with the smallest file size possible. For example, if I had to export the same animation as a frame by frame movie at half its size, it would have been 170 KB.

Presentation Optimization

Special techniques were applied to any image that could be used multiple times, such as the background, "back", and "ahead" buttons. The idea was to reduce display time by loading each into memory only once. A similar multimedia technique, known as pre-loading, reduced the display time by downloading images before they are required at times when nothing else is transferring. The entire presentation optimized for the web was less than 1 MB, which easily met the goal of minimal wait time when downloaded on an ISDN line. The higher quality elements for CD-ROM added up to just a little under 9 MB, which met the goal of minimal wait time for data transfer from a 4X CD-ROM. Optimizing each of the elements to reduce the wait time and putting them together into the multimedia presentation did not complete the development spiral. The last task was to test the presentation on "typical" (minimal) workstations. Though analysis indicated that the playback should be "smooth", problems could still arise. For example, some codecs can be extremely processor intensive, and do not playback smoothly on lower end computers. If playback of video or audio elements would have been found to be too choppy in this testing, I would have had to go back and experiment with compression using other codecs which would have increased data transfer but decreased processing.

Chapter 6

Conclusions

Conclusions

There are various risks in developing multimedia presentations related to team management, project organization, and product specification. This thesis explains how organizing an archival quality library of separate media can reduce the risk even if original product specifications change or multiple delivery systems are to be employed. Gathering digital media at archival quality starts with a basic understanding of each media and the equipment available to create or acquire objects of each media type. Optimization procedures used to convert archival quality media to components engineered into specific presentations tailored for specific presentation devices follow from that.

The understanding of the different media and acquisition knowledge have been used to develop a multimedia lab at The University of Montana. Within this lab, which became known as the Information Technology Resource Center (ITRC), the methods of acquisition and optimization have been applied to numerous projects such as UM ITRC Demo CD-ROM (Armetta, Holbrook, and Thompson, 1995), America Wars in Asia (Burgess, 1996), and Oil the Hidden Resource (Hughes, 1997). Detailed development examples from Oil the Hidden Resource are given in a previous chapter.

What does the future hold

It is not possible to predict what new delivery systems will be developed in the future; however, distributed information seems to be the most prevalent. Regardless, development of multimedia independent of the delivery system will continue to grow in fields such as education and entertainment. What was once thought impossible to deliver on any medium other than CD-ROM will be delivered through the airwaves directly into high definition televisions. In order to reduce multimedia development risks further, other topics such as multimedia aesthetics and graphic design, human computer interaction (HCI), and multimedia team management must be investigated.

Using multimedia optimization methods requires a fair amount of human qualitative analysis. At each media compression step, the designer must balance the aesthetic quality level against the physical object size, which determines transfer time. With enough experience, designers can develop heuristics that can automatically make these determinations without human intervention.

Current evidence suggests that although CD-ROMs offer longer-term storage than magnetic tape for archives, the length of time is not as long as originally predicted. Technologies will advance and longer term storage will become available providing authors with cheap options for storing media digitally.

More research is being done in compression techniques and in the future technologies such as MPEG4, Scalable Vector Graphics, and universal 3-D formats will provide better storage and delivery options for multimedia authors.

An authoring system that allows the developer to create a multimedia presentation with the archival media and then create an optimized version based on a set of criteria would solidify the archival library methodology. Currently no such software exists. As shown in the thesis example, multiple tools must be used for prototyping and final production.

Often, the primary stakeholders have little knowledge of the amount of time and work that is involved in producing a multimedia artifact. A methodology to analyze the risk and costs of developing multimedia products needs to be developed to help stakeholders understand the complexity of any proposed project.

Works Cited

Armetta, J. F., Holbrook, S. L., and Thompson, D. L., (1995, July) UM ITRC Demo CD-ROM [Computer Multimedia Program] Missoula, MT: The University of Montana Information Technology Resource Center (Dr. Lynn Churchill, church@selway.umt.edu)

Boehm, B., et al., (1988, May) A spiral model of software development and enhancement. IEEE, 61-72.

Boehm, B., et al., (1997) Developing multimedia applications with the WinWin spiral model. Lectures in Computer Science, 1301, 20-39.

Braham, R. (1995, July) The digital backlot, IEEE Spectrum, 51-63.

Burger, J. (1993) The desktop multimedia bible. Reading, MA: Addison-Wesley Pub. Co.

Burgess, Amy, (1996, August) America Wars in Asia [Computer Multimedia Program] Missoula, MT: The University of Montana Information Technology Resource Center (Dr. Lynn Churchill, church@selway.umt.edu).

Essex, J. (1996) Multimedia sound and music studio. New York: Random House.

Eastman Kodak Company, (2000, May) Photo CD products and features. (Available from [http://www.kodak.com/global/en/professional/products/storage/pcdMaster/aboutPCD.shtml])

Fortner, B. and Meyer, T. E., (1997) Number by colors: A guide to using color to understand technical data. New York: Springer-Verlag

Hughes, Rick, (1997, September) Oil the Hidden Resource [Computer Multimedia Program] Missoula, MT: The University of Montana Distributed Applications and System Lab (Dr. Ray Ford, ford@cs.umt.edu).

McCracken, D. D., and Jackson M. A., (1982, April) Life-Cycle concept considered harmful, ACM Software Engineering Notes, 29-32.

Murray, J. D., and VanRyper W., (1996, April) Graphics file formats (2nd ed.). Sebastopol, CA: O'Reilly and Associates, Inc.

Neuman, W. R., Crigler, A., Schneider, S. M., O'Donnel, S., and Reynolds, M. (1987). The television sound study. Unpublished manuscript. Massachusetts Institute of Technology, Media Laboratory, Advanced Television Research Program, Audience Research Facility: Cambridge, MA.

Rodney, A. (1998, June) Step-by-step scanning, Photo Electronic Imaging, 40-47.

Rodney, A. (1998, October) Can digital cameras be calibrated, Photo Electronic Imaging, 48-52.

Royce, W. W., (1987, April) Managing the development of large software systems: Concepts and techniques, Proceedings ICSE 9, 39-45.

Russ, J. C., (1995) The image processing handbook (2nd ed.). Boca Raton, FL: CRC Press

Sasse, M.A., Bilting, U., Schulz, C-D. and Turletti, T. (1994): Remote Seminars through Multimedia Conferencing: Experiences from the MICE project. Proceedings of INET'94/JENC5.

Stinson, D., Ameli, F., and Zaino, N. (1995). Lifetime of KODAK writable CD and Photo CD media. (Available from [http://www.cd-info.com/CDIC/Technology/CD-R/Media/Kodak.html])

Terran Interactive, Inc. (2000) Codec Central (Available from [http://www.terran.com/CodecCentral/index.html])

Thompson, D. L., (2000, May) Hardware and software references for digital multimedia. (Available from [http://www.cs.umt.edu/DASL/theses/multimedia-refs.html])

Waggoner, B. (1998, June) Making a good first compression: Comparing codecs for CD-ROM and similar uses. DV Magazine [http://www.dv.com/magazine/1998/0698/0698compress.html]

Waggoner, B. (1999, October) Making great video: Techniques to ensure your streaming Web video look as good as possible. DV Magazine [http://www.dv.com/magazine/1999/1099/index.html]