

MANAGING MULTIMEDIA

by

Frank R. Theuerkorn

B.Sc., Pepperdine University, 1994

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF BUSINESS ADMINISTRATION

in the Faculty
of
Business Administration

© Frank R. Theuerkorn 1996

SIMON FRASER UNIVERSITY

November 1996

All rights reserved. This work may not be
reproduced in whole or in part, by photocopy
or other means, without permission of the author.

Approval

Name: Frank R. Theuerkorn
Degree: Master of Business Administration
Title of thesis: Managing Multimedia

Examining Committee:

Chair: Dr. C. Ernest (Ernie) Love

Dr. Drew Parker
Senior Supervisor
Faculty of Business Administration

Dr. Blaize Horner Reich
Assistant Professor
Faculty of Business Administration

Dr. Art Warburton
External Examiner
Faculty of Business Administration

Date Approved: 22 November 1996

PARTIAL COPYRIGHT LICENSE

I hereby grant to Simon Fraser University the right to lend my thesis, project or extended essay (the title of which is shown below) to users of the Simon Fraser University Library, and to make partial or single copies only for such users or in response to a request from the library of any other university, or other educational institution, on its own behalf or for one of its users. I further agree that permission for multiple copying of this work for scholarly purposes may be granted by me or the Dean of Graduate Studies. It is understood that copying or publication of this work for financial gain shall not be allowed without my written permission.

Title of Thesis/Project/Extended Essay

Managing Multimedia

Author: _____

(signature)

(name)

November 22, 1996

(date)

Abstract

Creating a successful multimedia project requires a diverse pool of talent sets that must be continually monitored and kept communicating among the various resources involved. This can prove to be an overwhelming task to manage and is the topic of this thesis. This work provides a model for multimedia development, discusses management issues for the stages within the model, and provides software to gain useful metrics in this environment.

Dedication

To my wife and best friend, Chris Hamm Theuerkorn. Her input and support made this thesis a reality for me.

Table of Contents

| | |
|---|-----|
| Approval..... | ii |
| Abstract | iii |
| Dedication..... | iv |
| List of Figures | x |
| List of Tables..... | xi |
| I. Preface | 1 |
| Summary of Chapters | 2 |
| Proposed Model..... | 4 |
| II. Multimedia Overview..... | 6 |
| Definition..... | 6 |
| History..... | 9 |
| Present | 14 |
| Future | 18 |
| III. Software Process Models | 24 |
| Traditional vs. Multimedia Software Processes | 24 |
| Industrial Revolution and Taylorism | 25 |
| Transactional System | 26 |
| Cost of Computing..... | 26 |
| Centralized Computing..... | 27 |
| Market Forces..... | 29 |
| Traditional Models | 29 |
| SDLC..... | 30 |
| Waterfall..... | 32 |
| Rapid Prototyping..... | 34 |
| Object Oriented | 35 |

| | |
|-------------------------------------|----|
| TQM | 36 |
| Other Industries | 37 |
| Architecture and Design Firms | 37 |
| Film Industry | 38 |
| Literature on Multimedia | 40 |
| IV. Multimedia Environment..... | 42 |
| External Components | 42 |
| Vendors..... | 43 |
| Changing Landscape | 44 |
| Stereotypes..... | 45 |
| Estimating..... | 45 |
| Design Revisions | 46 |
| Design Documentation | 47 |
| Internal Components | 48 |
| Content Experts | 48 |
| Programmers..... | 49 |
| Graphic Artists..... | 49 |
| Video Professionals..... | 50 |
| Audio | 51 |
| Management Approaches | 52 |
| Just Make It..... | 52 |
| Cecil B. DeMilles..... | 53 |
| Bureaucrat..... | 54 |
| Best Approach..... | 55 |
| V. Framework..... | 57 |
| Company Profile..... | 57 |

| | |
|---------------------------|----|
| Culture..... | 58 |
| The Creative | 59 |
| Role of the Manager | 60 |
| Development Phases | 61 |
| Pre-production | 61 |
| Production..... | 62 |
| Post-production | 62 |
| Models | 63 |
| Standard Models | 63 |
| Turbulent Pond | 64 |
| Process Flow..... | 64 |
| VI. Pre-production | 66 |
| Project Plan..... | 67 |
| Purpose | 67 |
| Audience | 68 |
| Environment | 69 |
| Delivery | 71 |
| Content | 72 |
| Resources..... | 73 |
| Storyboard..... | 74 |
| Non-digital..... | 74 |
| Collaboration..... | 75 |
| Prototype | 77 |
| Depth-First..... | 77 |
| Breath-First..... | 78 |
| Not the Final | 79 |
| VII. Production | 80 |

| | |
|-------------------------------|-----|
| Engine | 81 |
| Off-the-Shelf vs. Custom..... | 81 |
| Testing | 84 |
| Interface | 86 |
| Consistency..... | 86 |
| Proof of Concept | 87 |
| Elements | 88 |
| Element Dictionary..... | 89 |
| Rites of Passage..... | 90 |
| Storage..... | 91 |
| VIII. Post-Production | 93 |
| Testing..... | 93 |
| System Usability Scale..... | 94 |
| Packaging | 96 |
| Package Design | 96 |
| Documentation..... | 97 |
| Duplication..... | 98 |
| Delivery | 100 |
| Channel Marketing..... | 101 |
| Support..... | 102 |
| Follow-up | 102 |
| X. Software..... | 104 |
| Overview | 104 |
| Software Components | 106 |
| Project Definition | 106 |
| Task Screen..... | 107 |
| Task Reporting | 108 |

| | |
|---------------------------------------|-----|
| System Requirements | 109 |
| Software and Turbulent Pond..... | 110 |
| Appendix - Software Source Code | 112 |
| Index Page (index.html)..... | 112 |
| Navigation Frame (tasknav.html)..... | 112 |
| New Task Page (task.html) | 113 |
| Task Perl CGI (task.cgi)..... | 117 |
| References..... | 123 |

List of Figures

| | |
|---|-----|
| Figure 1 - Turbulent Pond Model | 4 |
| Figure 2 - SDLC Model | 31 |
| Figure 3 - Waterfall Model | 33 |
| Figure 4 - Turbulent Pond Model | 65 |
| Figure 5 - Pre-Production Phase of Turbulent Pond Model | 66 |
| Figure 6 - Production Phase of Turbulent Pond Model..... | 80 |
| Figure 7 - Post-production Phase of Turbulent Pond Model..... | 93 |
| Figure 8 - Project Definition Screen | 106 |
| Figure 9 - Task Screen | 107 |
| Figure 10 - Task Reporting Screen..... | 108 |
| Figure 11 - Export Task Items | 109 |

List of Tables

| | |
|--|-----|
| Table 1 - Factors influencing user environments..... | 82 |
| Table 2 - Sample bug types..... | 84 |
| Table 3 - Sample naming conventions..... | 90 |
| Table 4 - Task Screen Field Definitions..... | 108 |

Chapter I

Preface

Recently, there has been a renewed interest in multimedia as computers capable of providing full screen video and sound are within the reach of many people's budgets. Also, as a society we have come to expect information to be more entertaining as well as informative. We are constantly bombarded by the media trying to vie for our attention to buy or subscribe to their latest products. In a sense, we have become more desensitized to the increase of information as we enter into the 21st century -- where information is power.

Technology will continue to improve and soon everyone will have access to information anywhere in the world (likely via the Internet). Barriers such as cost, connectivity, and technical literacy will continue to decrease and hence make attractive information more valuable.

To create a successful multimedia project requires a diverse pool of talent sets that must be continually monitored and kept communicating among the various resources involved. This can prove to be an overwhelming task to manage and is the topic of this thesis. This thesis provides a model for multimedia development, discusses managerial issues for the stages within this model and presents software to gain useful metrics in this environment.

Summary of Chapters

This chapter gives an overview of the thesis and proposes a model to formalize the process of multimedia development called the Turbulent Pond model. This model is introduced in the last section of this chapter.

The second chapter provides an overview of multimedia. A working definition of multimedia used in the context of this research is given and compared to the traditional forms of information delivery. Also, a look into the past, present, and future of multimedia provides a prospective of this industry.

Chapter three covers traditional approaches to software development and how they differ from multimedia. Models are investigated with a discussion of why they are not appropriate for multimedia. Next, the design firm and film industry are examined, since they have similar processes. Finally, the available literature on the subject of managing the multimedia process is reviewed.

A look into the multimedia components is the focus of chapter four. The problems from external components faced by this industry is addressed in the first section. The second section describes the internal components and the profile of the various resources involved in multimedia. The last section outlines the current approaches used by management in this industry.

Chapter five lays the framework for the process of developing multimedia. The multimedia company profile is defined in the first section. The phases of pre-production, production, and post-production that defines the development process are introduced. Finally, a model for this development process called the Turbulent Pond is presented.

The root of the pre-production phase of development is detailed in chapter six. The chapter discusses the planning of the project plan to develop the concept along with the storyboard and prototype that comprise this pre-production phase.

Production, the heart of development, is the focus of chapter seven. The development of the engine, interface, and elements that encompass the production phase are presented. The interrelation of these development processes is also examined.

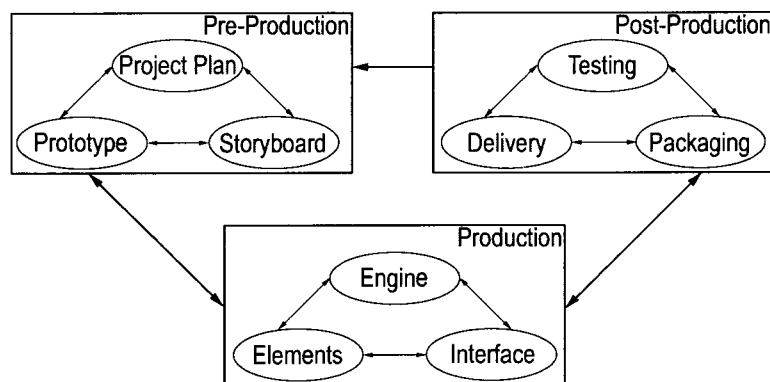
The final post-production phase is expanded in chapter eight. This is the critical phase that produces the final product. How the product reaches the intended audience is also addressed.

The last chapter gives an overview of the software that is provided that works in conjunction with the Turbulent Pond model to help manage the multimedia process. How to utilize and manage the usage of this software is presented. The source code for the software can be found in the appendix of this thesis.

Proposed Model

The Turbulent Pond model illustrated in figure 1 has three distinctive phases of development. The phases of pre-production, production, and post-production take the process of multimedia software from concept through development to the final deliverable product. The model illustrates that the process involves several iterations between these phases.

Figure 1 - Turbulent Pond Model



Within these phases exist stages that are also interrelated. The pre-production phase involves the formulation of a concept or idea into a perceptible product. The stages of project planning, storyboarding, and prototyping are iterated until a feasible idea is created.

The production phase takes the feasible idea from the pre-production phase and tries to build the product. Production is broken into the stages of the engine, interface, and elements of the multimedia piece. These stages are interdependent and require the careful coordination

between them. Often in the production phase it becomes evident that the pre-production phase must be reexamined as assumptions made in the concept are not actually feasible.

Finally, the post-production of the model is concerned with getting the final product to the targeted audience. This phase starts before the production phase has actually been completed and some processes begin in the pre-production phase. The majority of the effort in this stage is done when the previous phases have been completed.

Through the post-production phase the project is wrapped up or needed modifications identified in this phase are made. This will entail going back to the production or even the pre-production phase. Even when a product has been finished and delivered to the audience the project team must now focus on the pre-production phase to build a new product or develop an upgrade.

Each cycle through the model improves the process for a multimedia team, since previous iterations provide useful methodologies for future iterations. These methodologies are unique to each team as culture, skill sets and the nature of each product influence the development cycle. This model and the factors that influencing multimedia are examined in later chapters.

Chapter II

Multimedia Overview

This chapter begins by identifying the term multimedia and various definitions from several sources. Next, the history of multimedia is unfolded as the forms of hypertext and hypermedia build the foundations for the current industry. This leads to the next section that outlines the current state of multimedia and the recent technological advances that have fueled this industry. The last section forecasts the future of multimedia and how its importance will further increase as technology improves and market demands increase.

Definition

A recent report¹ from McKinsey declared that "Multimedia and the Information Superhighway are terms used so broadly that they have come to mean absolutely everything and, as a result, are beginning to mean virtually nothing". The same report goes on to warn that, although a lot of money will be made, "the potential for massive value destruction is painfully real". The indications are that with so many people talking, as it were on crossed wires (or even maybe "crossed fibers"), attempts by applications providers to bring the technology closer to potential users and their real needs is being thwarted by this simple lack of clarity in the scope of multimedia.

¹ Beardsley, S., Warwick, B., and Rooijen, M., "The Great European Multimedia Gamble", The McKinsey Quarterly, 3 (1) pp. 178-195

Here are a few of the popular definitions for the term multimedia:

Main Entry: mul·ti·me·dia

Function: adjective

Date: 1962

: using, involving, or encompassing several media <a multimedia approach to learning>

- multimedia noun

Merriam-Webster Dictionary

Interactive multimedia, any computer-delivered electronic system that allows the user to control, combine, and manipulate different types of media, such as text, sound, video, computer graphics, and animation. Interactive multimedia integrate computer, memory storage, digital (binary) data, telephone, television, and other information technologies. Their most common applications include training programs, video games, electronic encyclopaedias, and travel guides. Interactive multimedia shift the user's role from observer to participant and are considered the next generation of electronic information systems.

1996 Encyclopaedia Britannica, Inc.

Human-computer interaction involving text, graphics, voice, and video

Dictionary of Computing (<http://wombat.doc.ic.ac.uk/>)

In its most basic definition multimedia can be thought of as applications that bring together multiple types of media: text, illustrations, photos, sounds, voice, animation, and video. A combination of three or more of these with some measure of user interactivity is usually thought of as multimedia computing.

Apple Computer, Inc. (Demystifying Multimedia)

All the above are good definitions of multimedia of the past, however as technology evolves the form of multimedia will become integral into many facets and forms of delivery and not limited to the computer. Television sets are increasingly gaining computational power to provide better screen resolution and users can interact via the remote control.

Today's television sets have more computing power than the early military computers providing circuitry to improve poor signals and provide Dolby Surround Sound. Recently, the ability to retrieve data from the Internet through the television has been introduced with Sony's new WebTV². The unit is compliant with Picture-in-picture features, giving people the opportunity to "surf the Net" while watching their favorite television shows.

The traditional method of information delivery has been typically through linear mediums, like books, radio, and television which requires the participant to extract the information in a predetermined path. Information can become incoherent in the traditional delivery systems if not viewed from start to finish. This passive form of information retrieval does not lend itself well to multimedia systems and hence translation of the traditional forms of media needs to be redefined to successfully communicate to the user by hypertext and ultimately hypermedia.

² Sony Corporation, "Now Everyone Can Experience the Internet", 1996
<http://www.sel.sony.com/SEL/webtv/index.html>

The term hypertext describes an electronic text composed of nodes (blocks of text) which may be linked together non-sequentially. The World Wide Web is an example of a hypertext system. Here, each web page is a node, and links may be made to other pages, either at the same site or one on the other side of the globe. When the nodes contain elements of a literary work, hypertext becomes a site for artistic creation.

It is instructive to define hypertext not by packaging or technological features, but rather by the experience of the author and reader.

Hypertext provides for multiple authorship, a blurring of the author and reader functions, multiple reading paths, and extended works with diffused boundaries. The inclusion of sound, graphics, video, and other media as nodes (referred to as hypermedia) will expand the world available to a user.

Multimedia is information delivered using multiple formats (e.g. text, audio, video, etc.) stimulating two or more senses of a user. However, this information is not likely to be delivered in a passive format.

Successful multimedia will allow the user to interactively extract information that is of interest and can communicate this content in a variety of ways through the use of hypermedia.

History

The idea of non-linear nodes or hypertext is not a recent form of communication, but can be traced back to ancient times. A good example of this is the Bible. Indeed, with its many interlocking parts (the Synoptic

Gospels, for example, tell many of the same stories from different points of view) very few readers actually read the scriptures from beginning to end. Many find it just as useful to open the text to any page and read what ever chance/divine guidance shows them. Its history of multiple interpretations and re-writings (William Blake, for example, believed Satan was the hero of Genesis) makes the Bible the earliest model of hypertext.

In every Book, one might say, is a hypertext struggling to get out and vice versa. This should not, however, obscure from us the fact that the Bible continues to fascinate, and to inspire its believers precisely because it holds out the possibility of the divinely ordained word.

It was not until after World War II, that the first multimedia system was proposed. Vannevar Bush (1890-1974) is the pivotal figure in hypertext research. He conceived the idea of an easily accessible, individually configurable storehouse of knowledge that he called the memex.

Vannevar Bush first wrote of the memex early in the 1930s. However, it was not until 1945 that his essay "As We May Think" was published in the *Atlantic Monthly*. The frequency with which this article has been cited in hypertext research attests to its importance.

The memex is "a device in which an individual stores all his books, records, and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility". A memex resembled a desk with two pen-ready touch screen monitors and a scanner surface.

Within would lie several gigabytes (if not more) of storage space, filled with textual and graphic information, and indexed according to a universal scheme. All of this seems quite visionary for the early 1930s, but Bush himself viewed it as "conventional".

Bush saw the ability to navigate the enormous data store as a more important development than the futuristic hardware. Here he describes building a path to connect information of interest:

When the user is building a trail, he names it, inserts the name in his code book, and taps it out on his keyboard. Before him are the two items to be joined, projected onto adjacent viewing positions. At the bottom of each there are a number of blank code spaces, and a pointer is set to indicate one of these on each item. The user taps a single key, and the items are permanently joined [...]

Thereafter, at any time, when one of these items is in view, the other can be instantly recalled merely by tapping a button below the corresponding code space. Moreover, when numerous items have been thus joined together to form a trail, they can be reviewed in turn, rapidly or slowly, by deflecting a lever like that used for turning the pages of a book. It is exactly as though the physical items had been gathered together from widely separated sources and bound together to form a new book.

This passage is an apt description of the process of forming a link between nodes in today's hypertext packages.

In the late 1940s, Douglas Engelbart was stationed in the Philippines when he read Vannevar Bush's "As We May Think" in a Red Cross library. He became an early believer in Bush's idea of a machine that would aid human cognition. Later, he worked at Ames aeronautical lab,

and developed the idea that would form the basis of today's computer interfaces.

In the early 1960s, Engelbart began the Augmentation Research Centre (ARC), a development environment at the Stanford Research Institute. Here, he and his colleagues (William K. English and John F. Rulifson) created the On-Line System (NLS), the world's first implementation of what was to be called hypertext. As he states in "Working Together", Engelbart was particularly concerned with "asynchronous collaboration among teams distributed geographically". This endeavor is part of the study of Computer Supported Co-operative Work (CSCW); software which supports this goal is often called groupware.

"Augmentation not automation" was the slogan, the goal being the enhancement of human abilities through computer technology. The key tools that NLS provided were:

- outline editors for idea development
- hypertext linking
- teleconferencing
- word processing
- e-mail
- user configurability and programmability

The development of these required the creation of:

- the mouse pointing device for on-screen selection
- a one-hand chording device for keyboard entry
- a full windowing software environment
- on-line help systems
- the concept of consistency in user interfaces

Itemizing these accomplishments using today's terminology emphasizes their detachment from one another. However, NLS was an integrated environment for natural idea processing. The emphasis was on a visual environment, a revolutionary idea at a time when most people (even programmers) had no direct contact with a computer. Computers at the time consisted of data input by punched cards and output by paper tape.

Engelbart's work directly influenced the research at Xerox. In 1969, Xerox created its Palo Alto Research Center (Xerox PARC). Its mission was to explore the "architecture of information." In 1973, the Xerox Altos was the first true multimedia capable computer complete with a mouse and graphical user interface using icons.

The first multimedia capable computer available as a consumer product began with the Apple II, Tandy's TRS-80 and Commodore's Pet. These home computers did offer some colour graphics and multiple windowing capabilities, but the screen resolution was quite low to support a true multimedia system.

Apple introduced the Lisa in 1983 and later the Macintosh in 1984, that were directly influenced by the Xerox PARC project developed a decade earlier. In 1984, Telos introduces Filevision, a hypermedia database for the Macintosh. Later OWL introduced GUIDE, a hypermedia document browser in 1986 that allow user's to create their own hypermedia system. Here's the ending remark of a review on this new authoring system in *Byte*³ magazine:

Guide's innovative capabilities easily outweigh the current minor flaws in its user interface. The product points the way to the future "hypermedia" systems that will link animated video and sound with massive text and graphics files.

It was not until one year later, when Apple created its own hypermedia authoring system called HyperCard. The first widely available personal hypermedia authoring system that was bundled with every Macintosh computer. It was at this point that multimedia become a mainstream form of information delivery. However, it was not till five years later that QuickTime was developed by Apple to bring video to the desktop and to realize the future quoted in the above article by *Byte* magazine.

Present

Current desktop computers available today come standard with multimedia capabilities built-in. This is only a recent phenomenon as the

³ Hershey, W., "Guide", *Byte*, 12 (11), pp. 244-246

prices for audio and video support have come down making these features affordable to the consumer market. Even the current crop of laptops boast an impressive support for multimedia and have computational power that rivals the first CRAY super computers at less than 1% of the cost.

Desktop systems today combine the core computer and consumer electronics technologies to deliver multimedia systems complete with TV, radio, and advanced telephony functionality. These systems are capable of displaying full-screen, full-motion video, and audio systems that boast Dolby Surround Sound. Additionally, a built-in microphone can turn a standard home computer into a full-duplex speaker phone, with telephone answering system capability. Put that together with a high-speed modem connection and a low cost video camera, then video conferencing becomes an affordable and attractive alternative to long distance telephone service. As these capabilities become more mainstream, users will expect most forms of information to be multimedia rich to match their computer systems.

The Internet has recently seen phenomenal growth through the World Wide Web (WWW). The WWW project, started by Tim Berners-Lee while at CERN (the European Laboratory for Particle Physics), who sought to build a "distributed hypermedia system." In practice, the web is a vast collection of interconnected documents, spanning the world. He wrote the first Web clients and server in late 1990 and defined the URL, HTTP and HTML specifications on which the web depends while working at CERN.

In 1994, Netscape Corporation released its first commercial browser (Navigator) and its popularity gave rise to the recent growth of the WWW. The original browser developed had limited graphic and text layout capabilities. The current crop of internet browsers currently offer fully integrated email, newsgroups, video, audio, 3D, and telephone communications capabilities. It is estimated that the growth rate of the WWW is at 100 percent every 9 months (PCS, 1996, Hoffman & Novak, 1994⁴). New HTTP servers delivering multimedia content on the web have increased sixfold during the last year as seen in Webcrawler's⁵ commercial index server.

Netscape's browser (Navigator) is now supported on 16 different platforms bringing a new generation of multimedia creation for one-time development on cross-platform delivery. The rise of open standards based computing has as much to do with business as technology. During the mainframe and desktop PC eras, vendors' proprietary technology not only diminished competition, but locked in customers to a particular vendor. Open standards, in contrast, shifts the balance of power from vendors to customers. Furthermore, open standards enables

⁴ Hoffman, D. and Novak, T., "Internet and Web Use in the United States: Baselines for Commercial Development", Vanderbilt Univ., 1996
<http://www2000.ogsm.vanderbilt.edu/>

⁵ America Online, Inc., "WebCrawler's Web Size", 1996
<http://webcrawler.com/WebCrawler/Facts/Size.html>

interoperability, which promotes competition, innovation, and better avenues for rich multimedia delivery.

The adoption of new multimedia technologies have gained an amazing momentum in the software industry. Computer games introduced in 1994 (only one year after Apple released QuickTime) used small video clips to heighten the user's experience as in the popular game like *Myst*. The following year in 1995, games offered full screen movie-like productions as in Origin's title, *Wing Commander IV*. New software and hardware technologies are implemented immediately into multimedia titles, so as to remain competitive in this consumer market.

CD technology will soon gain a new format this year called DVD (Digital Video Disk). A single DVD can play up to 133 minutes of a full featured Hollywood film (about 92% of all movies ever made) on a single side. Picture quality is at nearly three times more resolution than VHS, demonstrably better than Laserdisc. DVD also offers up to 8 different sound tracks, and 32 subtitles, all with the push of a button. Imagine the possibilities for learning a new language while entertaining yourself!

DVD-ROM has the capability to store as much as 13 times the data of a traditional CD-ROM. DVD also has the unique capability of offering movies in multiple formats on a single disc. Like interactivity and multiple story lines that will allow you to watch the rating version that you prefer. An instructional DVD allowing you to analyze a golf swing from up to nine different camera angles.

Multimedia has arrived and the giant corporations are battling over the rights to serve this new technology into the household. This will only mean that those able to tame this new medium will be successful as their competition is left behind. To manage today's technology and plan for tomorrow's growth should be a key critical success factor for most businesses.

Future

Alan Kay, inventor of modern object-oriented programming once stated that to successfully predict the future can be achieved by several methods. The first way to predict the future is to look into the past and how the industry has been shaped. This method is seen in Moore's Law that states the processing power of computers will double every 1-2 years. Indeed next year promises to provide the MMX architecture to Intel's Pentium chip gaining an increase of performance of 50 to 400 percent, which does not include the increase in clock speed for the Pentium and for systems beginning to support multiple CPUs. The doubling of processing power is likely to be a conservative estimate for next 1-2 years.

Another method to predict the future, as stated by Kay, is to note what type of research is being done in the research labs. Today's researchers are building the prototypes of tomorrow's technology. Video-on-demand seems to be the Holy Grail for many companies and much of today's research is gearing for this goal.

The bandwidth of broadcast TV is about 27 Mbps (megabits per second), which makes the compression necessary over a 28.8 Kbps modem in the range of 7,500:1. Currently, the consumer hardware/software based compressors can achieve rates of around 200:1 which would require connection rates at 1 Mbps to realize broadcast TV over the Internet.

The reality of broadcast TV today requires a T-1 connection to the Internet costing over \$1000 per month. Additionally, the hardware to compress the video over this type of connection is close to \$10,000. This is hardly within the reach of the average consumer today. A final problem is the traffic load on the Internet itself to sustain the increased bandwidth required for broadcast TV. Current network service providers (NSPs) that provide the backbone link of the Internet to local ISPs currently average connection speeds of 10 Mbps, which could support only 10 users at a time.

Access to the Internet by the end-user will see some phenomenal increases as well. Two years ago, Internet World advised readers that “a 14.4 Kbps (kilobits per second) connection to the Internet provides enough bandwidth to handle the needs of the average individual user”. Even today’s 28.8 Kbps modems seem unbearably slow and next year's modems that support the 56 Kbps chipset from Rockwell will only temporarily satisfy the consumer market.

Integrated Services Digital Network (ISDN), which is the phone company’s digital service available in most areas now can provide throughput rates ranging from 64 Kbps to 2.048 Mbps. Unfortunately,

this requires additional hardware and expensive connection charges from the phone companies and ISPs. ISDN does have the capacity to serve video-on-demand to many consumers.

The phone companies are not the only players in this field as satellite and cable companies are planning to provide online connections to the Internet. At present the only satellite service is Hughes Network Systems' DirecPC that is providing 400 Kbps to 3 Mbps speeds with 10 Mbps soon. Cable modems being testing in various communities are already providing 500 Kbps to 10 Mbps speeds with the promise of 30 Mbps and beyond.

To answer the threat of the satellite and cable companies, AT&T Paradyne has announced a high-speed transmission technology, called GlobeSpan, that adds 6 Mbps of bandwidth to a standard phone line. If all goes smoothly, AT&T officials expect telcos, service providers, and applications providers to go online with GlobeSpan in the coming year. Beth Gage, broadband consultant for Verona, N.J.-based TeleChoice, said that by the year 2000 the total revenue for interactive video, data, and broadcast services will total \$350 billion.

This competition by vendors to provide the best bandwidth versus price will benefit the consumer and provide the means for broadcast TV. Advances in processing power will help compress the video bandwidth required and costs for the hardware will decrease. The final solution to the broadcast TV problem faced today is the current network support provided by the NSPs.

Network service providers are upgrading the backbone link of the Internet to connection speeds of up to 155.52 Mbps. By the end of the year, backbone speed will approach between 622 Mbps and 655 Mbps. In May 1996, MCI introduced the next generation transmission rate of 10 Gbps (gigabits per second) with future plans for 40 Gbps by deploying Four-Wavelength Wave Division Multiplexing (Quad-WDM). This would mean an approximate increase of 26,000% on the traffic load for the Internet within the next 2-4 years! This will amply provide the required bandwidth for broadcast TV before the year 2000.

CD-ROM technology will not disappear due to the emergence of the Internet as gains in higher storage capacity will be achieved by such technologies as blue light lasers. These lasers can read information in denser packets, allowing for 10 times the storage rate. Additionally, CD-ROM's with multiple layers can increase the storage capacity an additional 5-10 times so storage capacities of a single disk could easily reach over 30 Gigabytes without any compression format.

The mass production of these new formats will be based on the current technology, so the manufacturing industry will not have to make large investments to retool duplication plants. This will mean that ultimately, these higher storage CD-ROMs capable of containing entire libraries online could be manufactured for less than one dollar per CD-ROM. This will again increase the demand by the consumer for more rich multimedia content in the near future.

Display technology is also improving. This summer, Matsushita Electric Industrial introduced 'PlasmaView,' a 26-inch 16:9 aspect ratio wide-screen plasma display. The plasma display is just one-sixth the depth (only 8.5cm) and approximately half the weight (17.5kg) of equivalent screen-size cathode ray tube (CRT) displays and can easily be incorporated into a wall-hanging TV. Offering 16.77 million displayable colours, this deck has the versatility to become an important multimedia interface device in the home, as well as in such public places as schools, businesses, hotels, and airport lobbies. Estimates are that plasma displays will be capable of displaying 1000x800 pixel resolutions by years end on laptop systems.

In the future, all electronic devices will be connected and they will be sharing a user's personal information. Almost every object will have embedded intelligence, as a little bit of semiconductor content with a little bit of smarts. Homes of the future will be intelligent - knowing what room you are in and whether you are sitting, sleeping, exercising or eating. The home will adjust room temperature and lighting based on your activity. Additionally, it can be retrieving and sifting through information for it's occupant.

Delivering information in this new era will provide a challenge to most businesses that today are struggling to create very simple multimedia projects through corporate presentations, web based ad campaigns or even product support CD-ROMs. The user of the future will be more savvy on how information is delivered and presented. Multimedia will

make up the majority of information systems and companies need to begin to invest on how to deliver this content now and in the future.

Chapter III

Software Process Models

Every industry requires a model to allow management the ability to control the processes within a company. Models provide a means to measure the company's success in terms of quality, profit, and production. Multimedia is a relatively new industry and little has been developed to describe this process. Traditional software models differ from multimedia and do not provide an appropriate approach to this type of development.

The first section of this chapter explains why traditional software development differs from multimedia. The following section outlines the various software models of the industry and why they are not suitable for multimedia. The third section examines the design firm and film production. These industries share common characteristics of multimedia and have been used by some multimedia productions, but the methodologies of these industries do not adapt well to multimedia. The final section reviews the available literature on this topic.

Traditional vs. Multimedia Software Processes

The models for the software development process used in the computer industry are poor fits for multimedia development. No longer does the development of software involve the resources of just programmers and end-users in a set environment. Multimedia involves a whole range of talent resources within a project to deliver a product to a rapidly

changing environment. Listed are five reasons why there is a gap between traditional software and multimedia development.

1. Industrial Revolution and Taylorism
2. Transactional System
3. Cost of Computing
4. Centralized Computing
5. Market Forces

Industrial Revolution and Taylorism

Work simplification has been the trademark of industrial engineering since the industrial revolution. The assembly-line was introduced to allow for increases in production to a hungry market. Departments and job specifications were placed to help optimize the efficiency of this era. Frederick Taylor helped pave the model of management to remove all the influence of an individual worker and placed the controls into the hands of management and the technostructure. This is still prevalent in today's business culture as most businesses still rely on the manager for many of the business decisions and gone are the traditional craftsmen of the past.

Most businesses have become very bureaucratic and slow to external change or internal criticism, since these technostructures take many years to build and hence make them almost impossible to adapt in today's rapidly changing environment. Consumers expect technology leaps and the demands of the market reflect this acceptance. Consumers

buying a computer system today realize that their purchase will become obsolete in several years. This is not the same environment of the past and it would be in poor judgment to blindly believe in old management principles to deliver a product to this market.

Transactional System

Many of the systems developed have been to replicate the assembly line systems of our business processes influence by Taylorism. The computer was better at handling repetitive tasks at a higher transactional rate possible by an individual. Building transactional systems was a quantifiable task, since the system is well laid out and the computer need only replicate the process faster and more accurately.

These types of systems were easily justified through cost-benefit analysis as the input and output of the system is well understood. The flow of information is linear as data is inputted, processed and reported. Multimedia systems are non-linear by nature, information is usually delivered in an undetermined sequence as the user is free to roam through the system. Benefits are hard to identify and the cost difficult to estimate, since this is a new field. This probably explains why many companies choose not to invest into multimedia projects.

Cost of Computing

The cost of computers has dramatically decreased in the last half century. In the beginning, computing was affordable to only the world governments funded as research projects. Later, computing research

allowed advances to reduce the cost so companies could use this technology as a strategic tool. However, the cost of computing was significant and this caused a tight control of how the computer was being utilized within a company.

Today the cost of a computer is affordable to most households and it is estimated⁶ that 37% of the households in the United States own a computer and that by 1999 almost half will own a computer. Having a computer for a business is no longer a strategic advantage, but rather a cost of doing business. The focus has shifted from computing resources to how information (a product of today's market) is delivered through the value chain defined by Michael Porter⁷.

Historically, most software projects were code-centric that optimized business processes with the existing computer architecture. Multimedia is content-centric and the hardware barriers are dropping as features and prices make a computer a consumer product. This effect is making the message more important than the messenger.

Centralized Computing

Initially, the high cost of computing required a company to closely monitor its resource allocation and was typically best served in a

⁶ e-land, "The e-stats", 1996 <http://www.e-land.com/>

⁷ Porter, M., *Competitive Advantage: Creating and Sustaining Superior Performance*, The Free Press, New York, 1985

centralized system. The usual scenario was a main frame or central computer system and a department that controlled the usage of the computing facilities. As departments became more reliant upon the computer system and change requests were slow to be implemented (if they were approved) causing a frustration level within the company. Departments competed for needed enhancements and management policies were vital to monitor this effect.

As the price for personal computers dropped, departments were able to budget computing resources for their own use. The reliance of the centralized computing resources was decreased. Most companies had no formal plan for the integration of the personal computer and hence little or no managerial policies were in place to monitor these new systems. A small revolution was taking place in the corporate landscape and little was contributed in managing the process of developing a system to manage the disarray of information bits across these personal computer systems.

Multimedia is fundamentally developed around a decentralized system of personal computers. The lack of management systems developed historically on this type of information architecture leads to lack of a model to control the development of the architecture of a multimedia software development house.

Market Forces

Traditional models of software development had long development cycles, since the need for a complex system to replicate a business was involved. Most importantly great care was taken to ensure that the system performed to expectation or more time was given to reach this goal. Project development of traditional software could be measured in years and the software process models reflected this fact.

The current market for multimedia is changing and a single year of development can have a significant impact on the delivery of multimedia. As outlined in chapter two, the technological advances in computer technology are moving at a staggering pace and leading edge development can become obsolete or, even worse, passé in the following year.

Traditional Models

Software process models allow a multifaceted project like software development to be managed when more than one individual is involved in a work environment. Models allow managers to control these processes by placing metrics to measure such items as quality, budgets, and delivery of a finished product. Without any type of system a project will quickly run out-of-control and be over-budget, of poor quality, and miss on promised ship dates.

SDLC

The system development life cycle (SDLC) is well defined in the textbook, "Foundations of Business Systems"⁸. There are many derivations of this model, but this textbook defines the process in great detail and how the different implementation of each phase has been used for this model in the computer industry.

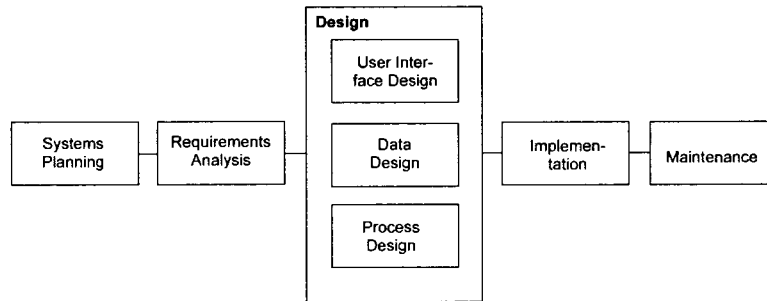
SDLC models divide the life of a project into phases. SDLC begins with some planning of the project to identify the resources needed.

Requirement analysis, the next phase determines the scope and success factors needed for the plan from the previous phase. Next is the actual design phase using a variety of diagramming methods to represent the various aspects of design, i.e. Data Flow diagrams and ER diagrams.

Once the design phase is complete the implementation phase is entered to create the new system and finally the maintenance phase is reached once the system is in place and running. Figure 2 illustrates the SDLC model.

⁸ Anderson Consulting, *Foundations of Business Systems*, The Dryan Press, Fort Worth, Texas 1991

Figure 2 - SDLC Model



Let's examine what this model is telling the potential manager of a multimedia project. First, the manager must gather the resources necessary for the project. Next, the manager must identify the scope of the project and determine what measures of success to use and have the design team work to develop model(s) for this system. Once the system has been designed, the programming team puts the project together and the system is then tested. Finally, the delivered product needs some mechanism for support.

The simplicity of the SDLC model is its greatest fault. It leads one to believe that the phases have a linear path, that once the project has been planned and designed the programming will just fall into place and any bugs will be handled through maintenance. In the textbook, "Foundations of Business Systems" the first 3 phases of the SDLC approach account for over 600 pages, but implementation including programming and testing is covered in less than 70 pages and finally maintenance is given a mere 13 pages.

The emphasis is on designing the structure of the project controlled by management and left to the employees to create this dream.

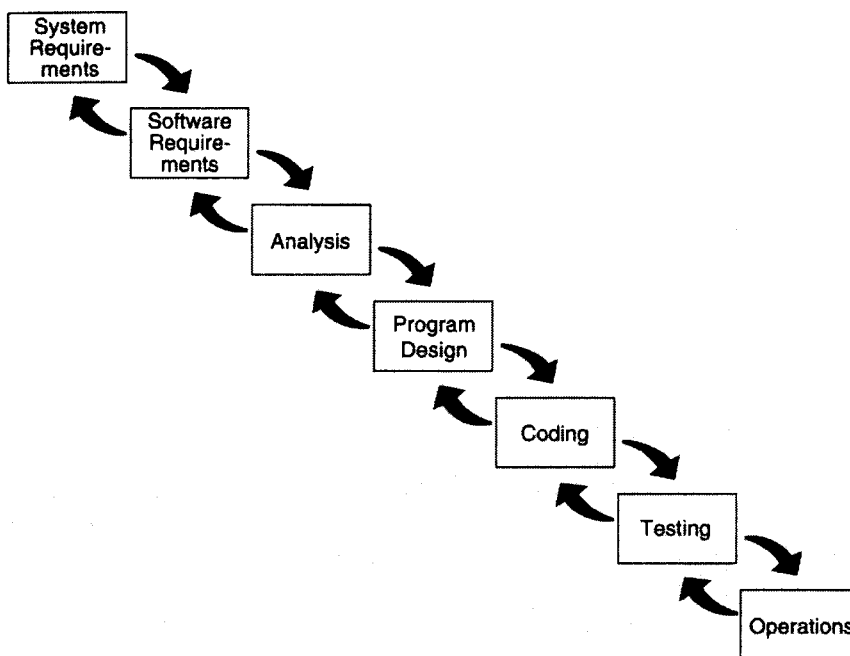
Unfortunately, real life does not reflect this Tayloristic utopia, but rather once development is underway the project's design may change due to many factors of the dynamics of multimedia, i.e. delivery limitations, unforeseen design flaws, etc. Multimedia is fundamentally a creative product, so this approach would be like telling an artist what brush, colour, and position to begin painting on the canvas.

Waterfall

The Waterfall method was first described by Dr. Winston W. Royce⁹. This method improves upon the SDLC approach where each stage goes through an approval process by management and can be moved back a stage if needed. A generic model of this is represented in figure 3.

⁹ Royce, W.W., "Managing the Development of Large Software Systems: Concepts and Techniques", Proceedings of IEEE WESCON, 1970

Figure 3 - Waterfall Model



The Waterfall is a potentially good model for multimedia. Everything is carefully designed, analyzed, and documented before a single pixel is generated, or a line of code is written. The separation of phases allows for a maximum efficiency of people's skills. Writers and designers come in and do their creation thing, then their results flow down into the actual building phase to the programmers and artists. By the time programming begins, the designers are onto another project.

Multimedia developers may embrace this structured control promised by the Waterfall, but the software engineering community is abandoning this model. The Waterfall model was based on a flawed notion that a development team can figure out the details in advance. The name of the

Waterfall model represents what was originally considered its defining strength, but is now viewed as its liability. There is no real mechanism to go all the way back upstream, except to the previous stages. Although in theory, it seems reasonably virtuous to work out every last detail and define every last bitmap in advance, the inability to respond to user feedback after the project has moved down the development stages can be fatal.

The Waterfall does have a nice big "test" section at the end, but that's only testing for bugs and functionality. In other words, the test confirms only that the program performs according to the design documents built at the beginning. If users think the game is too hard or they cannot figure out the tutorial interface, there is no going back.

Rapid Prototyping

Prototyping or rapid prototyping is a new idea to the software industry, but has been used for centuries by architects building scaled-down models of buildings. The prototype methodology involves immediately starting to build a demonstration of the proposed system rather than modeling it on paper.

This approach is very different from the SDLC and Waterfall models described previously. The benefit to the development software process is that it avoids creating a product that is not possible to build for the programming team. Also, this process helps gauge how much effort will be required to complete the project, since most major hurdles will be

worked out through the demonstration models. Additionally, it helps bridge the gap between user's expectations and development efforts.

The major drawback to this approach is that the programming team will dictate the design of the final product, which will be limited to the tools familiar with the team. This will create a series of static products and not push the limits of the programming team to build new and innovative projects based on the content to deliver. Additionally, resources are being wasted as the team spends time to develop these mini applications rather than building specifications to streamline their development process.

Object Oriented

Object oriented design tries to model the real world closer than traditional approaches. In the SDLC design, data and processes are modeled separately. Object oriented design is based on object oriented programming that all data elements (objects) have processes (methods) that are fundamental in their design and should not be separate. This approach brings easier modeling of real world systems and a closer integration of the programming to the model.

This approach however is very closely tied to object oriented programming and can be overwhelming for those unfamiliar with the terminology. This would lead to a project lead by the programming staff and their new tool called, "Object Oriented Programming". Time will tell if this approach will mature into an approach that more people could

embrace. However, object oriented design is more about a programming style rather than a management technique.

TQM

Total quality management (TQM) is the application of quality principles for the integration of all functions and processes of the organization.

Quality is defined as fitness for use by the customers (Juran, 1974). The development emphasis is on customer-oriented products. This is accomplished by using customer feedback in the development cycle or predetermining the needs by identifying who the customer is for the product.

Building quality into a product can easily result in spending too much time to finish a product. Markets are a moving target and delaying a product by continually adding quality may have the product miss the market. Also, quality is sometimes confused with adding features and known in the software industry as “creeping featurism”. This occurs when a product’s completion is constantly being delayed as more features are added to the product.

The bottom-line for any multimedia project should be to get a quality product with the original specifications to market as quickly as possible. A great product is of no value if it is not in the hands of the customer.

Other Industries

Some of the processes involved in developing multimedia are shared in other industries. The task of managing creative resources in a high production environment has been the concern for design firms. The film industry has a long history of creating a final product utilizing a diverse pool of talent with each production a unique process. Both of the industries approaches are examined to identify useful methodologies and why the models for these industries do not provide a good fit for multimedia.

Architecture and Design Firms

Architectural and design firms have been developing products from a team of creative employees for several centuries. Over this time a management system has developed to control this process. A survey¹⁰ done by the Organization for Economic Co-operation and Development (OECD) found that a calculation of resources on design work is not possible. However, design firms can make estimates based on past experience of individual draftsmen and place this into an overall work flow framework to ascertain the total time to completion.

Management received weekly reports from each of the designers that compiled their time spent on each order. Each project was broken down

¹⁰ OECD, "Design departments; a survey of the role, organisation and functioning of design departments and drawing offices in European engineering firms", Organisation for Economic Co-operation and Development, Paris, France, 1967

into smaller tasks called orders and assigned to various designers depending on their area of expertise. The manager was responsible for assigning these tasks and the executives were responsible for defining the model that defines a project. Many graphic design studios and multimedia houses use this old method, since it is simple and appears to have significant control over the creative team.

Some of the problems faced with this method are that the managers become too involved with the total number of hours compiled by each designer and often the designer fearing this trend will pad their hours to total a full work week. Also, the paper work involved can often distract from the real work at hand - designing. This was a major concern found in the study and often felt throughout the industry. This reporting methodology could also build a rift between management and designers.

Unfortunately, multimedia often adds another problem to this method, since no two projects seem to be similar. This makes it very hard to define orders that make up a model for projects. Additionally, as the weekly reports come into the manager it becomes an almost impossible task to compile a meaningful progress report on the project.

Film Industry

Recently, many film studios have entered into the multimedia field as the two industries seem to be on a collision course. With these studios comes a different methodology to manage this process. Film production is a mature industry that has a longer history than the software industry.

Film has always had to deal with pulling together the resources of many different types of creative resources to complete a project. Film has also had to adapt to changing technology and redefine how the process was managed. Many multimedia production houses have borrowed terminology from the film studio.

Filming is concerned with three areas in the development of a film: Pre-production, production and post-production. Pre-production involves gathering the resources necessary for the film and developing a storyboard based on the script of a film. Once everything has been planned the production phase is entered and actual filming is done. At this stage the resources and storyboard can be changed as needed. After filming has been completed the film enters into post-production where film editors, sound engineers and others compile the film into its final form.

Film is a very linear format in its production and final form. Multimedia is more dynamic in that filming, interface designing, and programming can occur at any and all stages of development. Most important, multimedia is not delivered to the user in a linear format. The user is not a passive viewer watching from start to finish as with film, but rather the user actively exchanges with the multimedia product to create separate strands of media experiences depending on the content and user selection. Often film studios introduce branching story lines for the user to interact, but this is a poor use of the multimedia technology that only slightly deviates from a viewer changing channels on a TV.

Literature on Multimedia

There are many books, magazines, and journal articles about multimedia, but there is a lack of titles devoted to managing the process of multimedia. Several titles that try to touch upon the subject of managing multimedia approach the topic by explaining how to create multimedia. This would be similar to teaching automotive executives managing principles of the industry by having them learn how to build a car. This is an important aspect of the industry, but it does not provide the tools to management to run an automotive manufacturing firm successfully.

Apple's book, "Demystifying Multimedia"¹¹ is a good attempt at bringing some tools to manage the multimedia process. However, the book's target audience would seem to be multimedia developers in general and not those managing the process. Apple's primary target has always been the individual consumer, so it would make sense that any publication from Apple would cater to individuals creating multimedia on their easy-to-use platform.

Various magazine articles provide bits and pieces to a very large puzzle, but finding a publication dedicated to this new and promising industry seems to be lacking. The remainder of this thesis strives to piece together some techniques, tips, and tools to arm the potential manager of

¹¹ Apple Computer, Inc., *Demystifying Multimedia: A Guide for Multimedia Developers*, Apple Computer, Inc., Cupertino, California 1993

multimedia. This work builds on current articles, a survey of practice, and the author's experience developing multimedia.

Chapter IV

Multimedia Environment

At the moment, multimedia technologies are a hot topic in the media and the heat is turning into hype. The attention is gratifying for the industry, but it's also creating some unrealistic expectations that need to be defused. Many people have bought into the notion that multimedia is the spark that will transform their business and revitalize their careers. Sometimes this "technotopian" dream could come true, but often there will be many that are greatly disappointed.

Many factors influence the development of multimedia and shape the industry. This chapter focuses on the multimedia environment with the first section outlining the problems faced by external components. The internal components section describes the production team. Finally, the various approaches used by management in this industry is examined.

External Components

Many factors influence the development of multimedia and external components play a large role. These components include vendors, the changing landscape, and industry perceptions of the development process. The industry's perceptions of multimedia development influence stereotypes of the production team, ease of multimedia production that effect project estimating, design changes, and documentation. These problems are the topic of this section.

Vendors

Many new users are entering the authoring marketplace without formal training or a background in programming, so tool vendors are hoping to attract these users with products that offer ease of use. The war rages between software companies claiming that their product is the easiest to use. At Apple's Worldwide Developer Conference in May 1996, a child demonstrated Cocoa, a new Web-playable interactive authoring tool for kids. After building an interactive program for the Internet, the boy turned to the audience and said, "Now, I'm your competition."

What the crush of new tools seems to be telling us is that we are about to experience an explosion of new authors who perhaps will never consider themselves multimedia producers at all. Rather, they are professional and amateur communicators-trainers, teachers, marketers, designers, and advertisers-who develop multimedia as part of their work. That is not to say that professional, full-time, dedicated multimedia developers and development companies will fade away. It simply means we are seeing a broader penetration and acceptance of multimedia as a mode of communication in all areas of endeavor.

This brings up the pressing question that if multimedia is simple to create and everyone can do multimedia with these new tools, should they? Multimedia is on the verge of what happened in the early days of desktop publishing. The rash of new tools means we will be seeing a lot more ghastly multimedia and all those terrible newsletters of the 1980s will now have sound, video, and a user interface.

The other potential problem with niche and ease-of-use products is that they typically trade design flexibility for a gentler learning curve. By lowering the wall, vendors are also lowering the functionality of their products. So rather than asking, "What user experience would best support our interactive goals?" Developers can only ask, "What information do we have that we can retrofit into this design structure?"

Changing Landscape

Multimedia is a series of rapid and unpredictable changes in a technology that seems to have no barriers. That's good news for companies of multimedia content, since new technologies means new opportunities. However, Managers must deal with two obstacles to unlimited and rapid expansion. First, the consumers of multimedia products have a limited ability to absorb new technologies and accept new ways of doing things. Second, managers face the limitations of their own ability to climb the steep and slippery learning curve while continually reinvesting and retooling.

Under the new rules of change, multimedia development adapts continuously by borrowing and sharing advances across disciplines. In multimedia, for example, the industry assimilates the best of graphical design from the world of print and paper. It adopts such things as blue screens, camera dollies and animation techniques from Hollywood. Multimedia production embraces recording methods from the music world and shares technologies, such as video-on-demand and the Internet, with all other forms of entertainment and communication.

Advances in instructional design from the world of corporate training are absorbed. Multimedia even incorporates evaluation and field-test strategies from educators, market researchers, and social behaviorists.

Clearly, the key to successful transition is the leveraging of talents, investments, experience, tools, and content in such a way that each area can be quickly adapted to the changing communications landscape.

Stereotypes

A typical multimedia developer has often been classified as under 30 and thriving on insane work schedules and debilitating all-nighters. In multimedia, you are expected to have miserable hours, impossible deadlines and a losing race to stay on top of the technology.

Unfortunately, this is not an uncommon stereotype of multimedia production and production professionals. This is evidenced from a help-wanted brochure distributed by DreamWorks Interactive, the multimedia division of the new Spielberg-Katzenberg-Geffen studio venture in Los Angeles.

"The ideal candidate would have the following qualities:
Likes: Midnight Brainstorming, Any Type of Music (especially LOUD), Avant-Grunge, Killer Tomatoes. Dislikes: Big Fat Code, Engineer-Rendered Artwork, Me-Too Products, SUITS!"

Estimating

"Six weeks? No problem." Somebody somewhere is making that promise on a standard 10-week project. The important question is not whether

developers are over-promising, but why. The simplest answer is the developer's lack of experience. Many multimedia developers will promise products based on technology that has not been released or is in a stable state. They will promise functionality based on product reviews covered in the latest computer trade magazine or paper.

That does not explain the larger problem of why seasoned developers who seem destined to over-promise and under-estimate on virtually every job. Developers seem to be afflicted with some sort of selective amnesia and just forget how long it takes to create some types of multimedia projects.

Over-promising usually has a more unsavory cause by the developer saying anything to secure a contract. In an effort to win the client, the producer or account executive is expected to employ risky optimism or to even lie when making a bid, since the competition will likely make similar promises.

Design Revisions

Another cause of chaos is the industry's tendency to demand ongoing design revisions, in some cases occurring right up until the date of delivery. One of the benefits of this medium is the inherent creative flexibility, giving the multimedia producer the leeway to explore, experiment and implement changes quickly. Creativity, however, is always a trade-off. At some rational point, the design must be set, the decisions made, and development wrapped up.

During the making of the Rolling Stones *Voodoo Lounge* CD-ROM, published by Virgin Interactive, several developers overheard project executives debating whether to go with two discs instead of one, as if such a change would not affect the ship date ("Sparks Will Fly," Multimedia Producer, October 1995).

Design Documentation

The *Voodoo Lounge* project also proves that there is a direct relationship between coherent, detailed design documents and the ultimate stability or instability of the project.

According to team members, *Voodoo Lounge* had a production binder worth bench-pressing, but revealed virtually nothing except visuals, storyboards and screen designs. There were no real logic flow charts or diagrams, and no specific description of functionality. Team members had to work it out for themselves, but they also put in 18 hours a day for several months that would likely have been avoided.

Then again, sometimes documents start out complete, but fall out of date. If the documentation is not detailed or is not kept current, then it's just a waste of trees. Megan Wheeler, co-founder and creative director at Ad hoc Interactive, developers of "A Passage to Vietnam", admitted to an audience at the Macromedia Developer's conference that the project was completed several months beyond the original expectation. And when someone in the audience asked whether the developers had a design

document that accurately reflected the program as it was finally delivered, the answer was a sheepish "no."

Internal Components

Many people believe that multimedia is simply the process of putting various pieces of media together into a coherent form. This is only partially true, since originality is one the most important concerns of any multimedia product. The copyright of content is a serious issue for multimedia in any environment. Even in-house multimedia projects can be the cause of legal actions by the originator of media piece in a copyright infringement suit.

This means that every piece of the multimedia puzzle needs to be originals created by the company or have licensing agreements worked out with the original creator. It is impossible for any multimedia project to avoid the involvement of creative people though the development cycle.

Content Experts

Somewhere there exists in every multimedia production a body of knowledge as the source for the project. A content expert is someone who knows a subject area intimately and can help a project team find and select materials. Content experts come from all walks of life: teachers, historians, amateur researchers, or anyone who commands the authority on a given subject.

If an expert is unable, then a content researcher is needed to collect information from different sources such as literature, historical media, interviews, and other materials. The researcher must not only be able to select appropriate content, but also evaluate the validity of the source.

Great strides must be made to translate this existing knowledge or work into a digital format. Meaning can be lost if care is not taken to transform the content into a viable format.

Programmers

Programming skills are almost always essential to a multimedia project. Usually the programmer of multimedia has a much shorter development cycle than traditional software. This is mainly due to the existence of authoring systems for multimedia and the emphasis placed on content rather than delivery mechanics of the software. Sometimes development needs the creation of an original multimedia engine to deliver the content, since more speed or functionality is required. Often programmers provide the model to structure the content of the multimedia piece and will work closely with the graphic designer to create a viable interface and delivery tool for the content.

Graphic Artists

Essential to all multimedia projects are the graphic artists filling various roles in the production cycle. These artists are needed to create the graphics, the layout and visual design of the electronic product, and its packaging. Graphic professionals are usually involved in the whole

development cycle from original storyboard concepts to production of the media pieces to packaging of the final product.

Graphic artists can be separated into several categories: designers, photographers, illustrators, and animators. Designers lay out screens, design icons and symbols, type specs and colour schemes, and decide the overall visual balance of elements. Photographers are often required to create photos that may be the main focus of a multimedia project. Illustrators create drawings, diagrams, cartoons, and three dimensional models often to convey more meaning than a photograph can. Finally, animators work in either the familiar traditional cell based format or in the three dimensional worlds created by such technologies like VRML (Virtual Reality Modeling Language) to allow the exploration of an object or event more so than video.

Video Professionals

This is usually the most under-estimated task facing multimedia productions as several misconceptions face this medium. Most people are under the impression that armed with a Hi-8 camera and a video editing program the video production is plausible. The assumed final quality is targeted as low, since the video playback on multimedia currently is not broadcast quality like on TV. However, the quality of the original material is very important to the quality of the final digital form of video.

Video is a demanding task upon the computer and poorly created video is exaggerated on the computer screen. Lighting, colour balance, and

framing are just a few considerations in filming and usually only possible by filming professionals. Most film produced on commercial multimedia titles are full scale productions complete with a director, actors, camera, and lighting crews. Once the filming has been completed the film is then edited.

Again, editing is best left to the professional capable of keeping continuity between takes and laying the audio and music tracks. Finally, the piece is digitized to be played back on the computer and again many variables enter into this process like frame rate, data rate, and compression techniques. Bad choices in compression techniques can make the best film clips unbearable on the computer screen.

The digital process of video is changing as digital cameras becoming available, so video may soon be in immediate digital form upon initial filming. This will then involve the industry to redefine how video production is done.

Audio

Multimedia can also involve the creation of various forms of audio for the final product. Audio can take the form of ambient sounds (like clicking a button), voice-overs or music. Audio professionals can be sound designers, audio engineers, musicians, and voice talents. Sound designers put the overall sound experience in a multimedia project to create a certain mood or tension. Audio engineers record voices, sound effects, and background ambient sounds. Musicians are often needed to

create tracks for background music and scores. Finally, voice talent is needed to supply dialog, narration, voices for characters, and translating products for foreign markets.

Management Approaches

It's easy to find examples of why multimedia production is logging in overtime, so it's tempting to assume that the solution is to not make those mistakes. Making informed, realistic promises, minimizing late design changes, and building and maintaining good design documents would be a great place to start. But practices have a way of veering from the track of common sense. The common approaches to managing multimedia is examined in this section.

Just Make It

A common approach in multimedia is the "just make it" model. Popular with some creatively driven producers, JMI assumes everyone on the team knows what the program is about and what it is supposed to do, but does not provide a road map of how to get there. A small team of pros who communicate effectively can sometimes do surprisingly well, building ad hoc documents and working out the unresolved issues as they appear. This model also can produce some truly creative programs, crafted more like works of art than software. But JMI is not for the timid or financially challenged. It's full of risk and stress, and the probability of delivering on time is extremely remote and increases the chance of creating an undeliverable product.

JMI seems to be the norm for the multimedia industry, since managing a diverse group of professionals can appear to be an impossible task. How is a manager able to oversee a diverse talent pool such as programmers, graphic artists, musicians, photographers, and video people? To be an expert in all these areas would be rare and likely questionable at best. A manager is often in charge of such events ranging from a video shoot to finding a programming bug to graphic designs for the interface.

Cecil B. DeMilles

Many managers believe a hands-on approach to every phase of the development cycle is crucial to the success of the product. These are the jack-of-all-trades and master-of-none, who must be involved in every development process. TV producers have long had a term for these types: Cecil B. DeMilles, after the pioneering, autocratic movie director known for carrying a riding crop around the set.

This enthusiasm of a manager may seem to be warrant, since multimedia is such a multifaceted process and lack of direction or cohesion may feel threatening. This enthusiasm can easily begin to overwhelm the project, since professionals must step down several levels to bring the manager up to speed. The manager should know when it's appropriate to let the professionals do what-they-do-best and try to learn as much about the process from a passive reference.

This is not to suggest that the manager can not be an expert in any of the fields of development, but it is unrealistic to expect the manager to excel in all the phases of multimedia. The focus of the manager is exactly what the title implies - managing. A manager that is constantly designing, filming and/or programming is not managing the process which is key to the success of the final product. Multimedia is not enjoyed by the individual pieces, but best as a complete final product!

Bureaucrat

Multimedia production often falls prey to the bureaucracy of large companies that have many channels to report. The developers involved in a multimedia project are likely filing some type of weekly report of their progress as some unknown force is compiling this data. This approach bears the resemblance to the design firms methodology mentioned in the previous chapter.

This style of management can also lead to the overwhelming of a project even though this approach is opposite to the Cecil B. DeMilles types. Reports have a tendency to be padded by the people filing them and misunderstood by those compiling the reports (if they are indeed being compiled). Additionally, project focus is lost by the development team as they produce their modules in a vast vacuum controlled environment.

It is not critical that the manager is involved in the day-to-day operations of the development of multimedia, but care must be taken to not alienate management from the development team. The manager must provide the

direction, since the big picture is often lost by the development team as they become centric to their development efforts.

Best Approach

It is very important to note that there is not one single style or approach to multimedia that is correct. Business in general does not have definitive rules as to what management style works best in each industry, since a company's culture defines how an organization will ultimately operate. Multimedia is no exception and likely exaggerates this more than a typical business as the culture of the employee is a creative-type from many different disciplines.

Multimedia managers should be accustomed to a rapidly changing environment and struggle to keep abreast of the latest technologies. That is also true of the client as well, although more typically they do not have the advantage of the comfort and proximity to change that most of the multimedia professionals have come to expect as routine. Instead, many exist in an environment in which bureaucracy is an integral function of productive mass-market output, where stop-gap solutions prevent rather than encourage change, and where hierarchies of approval and fragmented work tasks provide no perspective for identifying areas of improvement. Clients from such a background tend to be much more comfortable with incremental, periodic change, or no change at all. They are sometimes ill-adapted to the monumental, constant change demanded in multimedia.

Leadership Skills

Multimedia managers must see the big picture while attending to the fine details as needed. They must be aware of the forces of change, and the corresponding "corrections" and impacts, as they design solutions to reflect the realities of change. They are often the bridge between the development team and the client.

It would be unrealistic to believe a manager could know all the details and processes involved in a multimedia project. This would require a very diverse knowledge set for the manager to possess. This knowledge would span from illustration skills like graphic design to analytical skills of programming to composition skills in music. Also, handling all the details in a large project is impossible as some multimedia projects can incorporate hundreds of resources to complete. Finally, all these processes must communicate throughout the team, since multimedia is a collaborative effort.

To be an effective leader, a manager must steer the development by promoting communication among the various resources and only become involved in the fine details when they affect the big picture.

Communication and interpersonal skills are the most important skills to the manager to effectively lead this type of development. The manager must in effect be the conduit between the various resources involved the development cycle.

Chapter V

Framework

How a company produces multimedia can vary from project to project. It is important to have a model to define where and how a company will be involved in the development cycle. Often, content experts develop many of the media elements and develop the storyboard and prototype ideas. Companies may then be only required to create the final product and thus enter into the development cycle at later phase of development. Other times, the company may need to help the content expert develop the product concept at an earlier phase of the development cycle.

Models also help build a gauge to estimate cost of resources and timelines for multimedia production. This helps monitor the process which at times can seem very chaotic as various creative types from many different fields work in tandem to produce a multimedia product.

This chapter will examine the company profile of the multimedia environment. Then, the areas of pre-production, production, and post-production that define the development phases will be introduced. Finally, a model for this development process called the Turbulent Pond is presented.

Company Profile

Various stakeholders that are the profile of a company will have different goals for the firm. Freeman's (1984) definition of stakeholder as "any

group or individual who can affect or is affected by the achievement of the organization's objectives"¹². Each stakeholder group - owners, managers, employees, clients, suppliers, distributors, and customers - views the firm from a different perspective. Stakeholders establish goals from this perspective based on their own interests. A manager must understand this profile to be able to work within this process.

Culture

Anyone that has been involved in a multimedia development environment knows that the process is far from simple. A company's ability to overcome problems in the development process needs to deal largely with attitudinal factors that are often difficult to control. These attitudes are often the results of the need for creative types required for this process and the interaction among the project team of various backgrounds.

These attitudes are the result of a much larger set of influences as seen in many well-known studies. These can include the behavior and values of various stakeholders described by Freeman or can be strongly shaped by environmental influences as seen in the Hawthorne lighting experiments¹³ as described by Mayo. Finally, a company's administrative

¹² Freeman, R.E., *Strategic management: A stakeholder approach*, Pitman Boston, Massachusetts, 1984

¹³ Mayo E., *The Human Problems of an Industrial Civilization*, Viking Press, New York 1960

system (e.g. measurements, accounting, appraisal, reward, and training) can have an adverse effect on attitude.

Harold Guetzkow¹⁴ defines organizational processes into two types, those which hinder the development of innovative behavior and those which enhance the creativity of the members of the organization. Traditionally, companies have hindered the creative process to make gains in automation and hence productivity. However, innovation is seen in today's market as a competitive advantage as illustrated in Michael Porter's¹⁵ generic value chain in the support activities outlined in the model as technology development (innovation).

Ultimately, organizations exhibit simultaneous demands for automation and for innovation. The balance of these countervailing pressures determines the organization's climate for the creative member.

The Creative

The creative individual must charter into unexplored areas, which has a high potential for error. New ideas, by their very definition, have not been tried before, so the chances that they may be unpractical is great. It is difficult to gauge whether innovations are useful or impractical until

¹⁴ Steiner G.A., *The Creative Organization*, The Graduate School of Business University of Chicago, 1965

¹⁵ Porter, M., *Competitive Advantage: Creating and Sustaining Superior Performance*, The Free Press, New York, 1985

further research and development is done. To avoid possible premature judgment requires time and resources, which the organization may not always be able to absorb.

The creative individual also must have contact with the organizational environment through the various communication systems. Often the trivial conversations of various team members can spawn solutions for creative blocks faced by an individual. Also, studies have shown that individual creativity is often a “lonely” process, with the innovator at times needing isolation and seclusion. How to balance these two extremes is best left to the creative individual, since this is best monitored by the creative themselves.

Role of the Manager

A systematic and disciplined approach to managing the multimedia process is important. As seen thus far, the development task at hand is a complex process and the management of a successful multimedia company can not participate in the daily events. These events are too numerous, extremely interrelated, and require substantial details to be controlled by management.

The role of the manager is a subtle approach. The manager must gauge the performance of the team and manage the company’s culture to balance production gains and enhance innovation. The manager must not concentrate on intrinsic technical skills of the design process, but

rather focus upon the larger picture of the development cycle and to arbitrate among the various stakeholders.

Development Phases

Developing multimedia covers a very wide and diverse process from the inception of an idea to design and into a final deliverable product. There exists a need to break this process into definable segments for management. Popularized by the film industry is the notion of production broken into three phases known as pre-production, production, and post- production.

Pre-production

Before any product is developed there initially existed an idea for this venture. A company must be cautious before committing any significant resources into creating a product to determine its potential for success. Failure to examine such factors as purpose, audience, and content early on can quickly destroy a company.

The pre-production phase concentrates upon the steps a multimedia company must address before the actual product is developed. A systematic approach to charting a potential product's success by managing key deliverable events is covered in more detail in chapter five.

Production

The product phase is begun when a multimedia company commits to the development of a product. The work created in the previous phase (pre-production) provides the guide for development in production. The production phase requires the tandem effort of many processes and coordination between these is critical for the product's success.

The main focus of the production phase is to build the engine for the project, develop the interface on the engine, and create the elements that fill the interface. Chapter six examines the product phase of the development process.

Post-production

Once the product has been completed by the development team it enters into the post-production phase. The best product is worthless until it reaches and is perceived of value by the targeted audience. To reach this goal the product must be tested, packaged, and delivered to complete the development cycle.

Often it is assumed that once the development cycle is in this phase the product is nearly complete, but if the product falls short of expectations then the process may need to be reiterated in previous steps or even phases. It is even possible that the original project plan developed in the pre-production cycle may need to be revised and the development cycle restarted. Post-production is covered in more detail in chapter seven.

Models

The three phases of pre-production, production, and post-production that comprise the turbulent pond model have similarities to other models. However, the Turbulent Pond model represents the iterative process between the phases and the intercommunication between the stages within each of the phases.

Standard Models

Standard models from SDLC approaches show a linear relationship to the development process. A project plan (system plan) is created and then passed on to a requirements analysis step and on to design and finally to implementation and maintenance. The model does not support any iterations of a previous step, since the assumption is made that the product being developed has been accurately defined at the beginning of the development cycle.

The Waterfall approach improves upon the model by allowing some iterations of stages within the cycle, but does not provide a mechanism for intercommunication between the processes. Also, the Waterfall model lacks a feedback loop for remote steps. Any imperfections found in the final product (operations) will not effect changes to the original concept (system plan).

Turbulent Pond

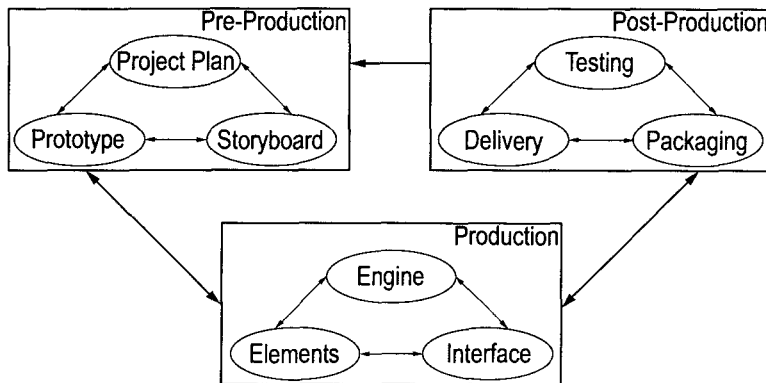
The early software models have a linear flow of processes that encourage each step to begin only when the previous step was completed. In multimedia there are too many processes to allow for this step-by-step approach. The steps within the phases must be developed in tandem and the phases closely integrated. In effect, the Waterfall model needs to be compressed down, thus turning the easily manageable flow of a waterfall into a turbulent pond.

Some linearity exists in multimedia development, but generally the process is far from a step-by-step approach. This can be very disturbing for a manager that requires a well defined control structure. Any type of control structure in multimedia is subject to constant revision and only useful for estimating time schedules and costs for production.

Process Flow

The diagram in figure 4 models this Turbulent Pond methodology for the multimedia development cycle. The steps within the phases can be run in tandem; as in the production cycle that may require the engine, interface, and multimedia elements developed at the same time and put together as the designs begin to finalize. This helps cut down the development time to build a multifaceted project like multimedia.

Figure 4 - Turbulent Pond Model



It is important to note that all three phases are inter-related and development can move between any phase or step within the phases as needed. For example, if testing reveals that the targeted audience has trouble using the product interface then the development process will move back to the pre-production phase and the storyboard and prototype will be redesigned.

There is a notable exception to this inter-relational flow between post-production and pre-production phases. Development can move from the post-production to the pre-production phase as in the previous example, but it is not possible to move from the pre-production to the post-production phase. This would likely suggest that the original project plan developed in the pre-production phase be revised to fit the testing results without any changes in design. The implications are that the product is either being shipped prematurely based upon false assumptions by sample testing or that development in production has deviated from the pre-production design work.

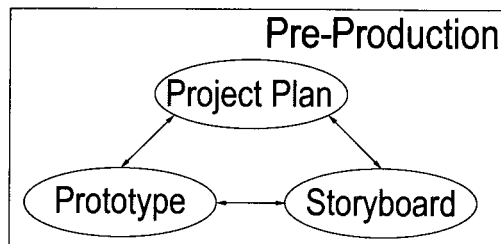
Chapter VI

Pre-production

As discussed in chapter four, pre-production (figure 5) is the first phase of the development cycle of a multimedia project. Within pre-production there are three stages:

- 1) project plan
- 2) storyboard
- 3) prototype

Figure 5 - Pre-Production Phase of Turbulent Pond Model



A well planned out project plan, the first step of the pre-production phase, ensures that everyone understands the task at hand and puts some initial metrics for success. Once the project plan is completed the storyboard can begin to develop ideas of how the project plan can be created. From the final storyboard can begin the prototype stage to give shape to the ideas presented in the storyboard.

Each of the above three stages need some formal agreement (called sign-off) after completion to signal to all parties involved that development moves to the next stage. Additionally, to reach this sign-off should

involve some form of testing to guarantee success in later stages of development. It may become necessary to revisit this stages if later development finds flaws in previous assumptions made in these early stages.

Project Plan

At the beginning of all multimedia production exist the concept for the project to be developed called the project plan. Unfortunately, this is often an informal effort and the assumption is made that everyone understands what the product is all about. It should be clear to all parties involved in this production of what is expected from the client, audience and production team. Many companies will commit significant resources before any agreement is reached on what is to be developed.

Purpose

Multimedia is created for a purpose. It is essential that everyone understands the reason for the creation of this product, that is, the software's mission. This is the first step for the creation of a good multimedia product and is often the most ignored step. A simple one sentence of why this software exists can save a multimedia production from losing its identity as the tides of change can cause a project to be continually updated.

Critical success factors (CSFs) are another important part of defining the multimedia project. Setting early CSFs for a production can help evaluate

the finished product to determine if it has been successfully created. Poorly formulated CSFs can bring failure to a project, since balancing the right number is almost as important as the CSFs themselves.

Defining just a few CSFs may not provide enough of a gauge to see whether the message of the software was conveyed or the content easily used. Defining too many CSFs will cause a multimedia title to turn into an all-purpose tool, that tries to do everything, but ultimately accomplishes nothing. J. F. Rockart¹⁶ who has written much on the topic of CSFs claims that defining between 4 to 6 CSFs will contribute most to the overall success of a function.

Audience

Every multimedia project will appeal to a different audience and the challenge is to convince this audience of the added value. Audiences for multimedia projects fall into two distinctive categories, vertical or consumer markets.

Customers in the vertical market have very specific characteristics and interests and products need to reflect these. Vertical markets are mainly found in the business and educational industries. The challenge in this market is to meet the specific needs within the targeted audience to dictate the success of the product.

¹⁶ Rockart, J.F., "Chief executives define their own data needs." Harvard Business Review, 57 (4), pp. 81-93

Consumer markets are more difficult to define, since they cut across many specific characteristics and interests. Also, the expectation of this audience is generally higher than vertical markets. Marketing experts may be needed to best identify the target audience for consumer market development. Listed below are some factors to consider on determining the audience in this market.

- Age
- Education
- Culture
- Occupation
- Language
- Computer experience

Identifying the correct audience will allow the recruitment of this targeted audience to test the product before release in the later stage of development. This will give a gauge on the potential of success for a product and the need for any revisions before release of the product.

Environment

An important consideration in the early phase of pre-production will be concerned on how the product is to be delivered to the targeted audience. The intended environment can effect how a multimedia piece is constructed, since an environment can consist of a single user, lab, kiosk or presentation. Each of these environments need to convey the

message(s) of the multimedia piece differently. What will work as a shrink-wrapped consumer product will not likely be effective in a multimedia presentation.

Single user environments will have the lengthiest interaction between the user and software. The user is willing to invest a certain amount of time to get familiar with the product and likely will invest the time to get the most out of the multimedia pieces. Continuity and consistency are very important and if the software lacks these, then the user will perceive this as a flaw in the design. The goal of this design is in the overall packaging of the multimedia pieces in the product.

The lab environment requires short quick segments as the user will be working on items along with the software. Software in a lab environment needs continuity and consistency, but it is not critical as in a single user environment. Additionally, the software will need clear segments so the user will know when to focus their attention elsewhere and the ability to quickly move to the next stage. Software is often mistakenly categorized as a lab environment when it actually replaces a process. This type does not replace the entire process (just portions) or it would be categorized as a single user environment. Lab environments should focus on delivering the content in well defined steps or stages.

Kiosk environments need quick unrelated segments for fast interaction, since the user has a specific information need and will not spend time learning an interface. The diversity of information will likely need little to no continuity, but a consistent interface is still important as to

streamline a user's request for information. Information should be fairly flat and not buried in an overly designed interface. This environment's goal is to present information in a quick and easy format.

In a presentation environment the software requires only one segment or flow, but requires breaks within the segment for the speaker to elaborate and entertain. The focus is not primarily on the software as in single and kiosk environments, but is shared with the speaker controlling this type of multimedia product. The software needs to have frequent interrupts within the presentation flow and allow for a clear message to be delivered to the audience. The goal here for software is to enhance a presentation by delivering clear and simple messages.

Delivery

A question to ask early should be how is this multimedia product going to be distributed and how will it be viewed. Simply determining the format of distribution, (i.e. CD-ROM, Internet or floppy diskette) solves the problem of duplication. What is more important will be how the targeted audience will view this format on their system. Developing content on CD-ROM can range from 1x spin drives that can only play very small video clips to 8x spin drives capable of full screen and motion video.

Deciding how to create multimedia for a specific delivery format would seem an obvious consideration. Unfortunately, a quick glance on many web sites will show designs that overwhelm the average modem connection of a user. This clearly tells us that many sites (including large

corporations) are not considering how the average user will be viewing their software.

The format does effect how the multimedia pieces are going to be put together, since the delivery of media over the Internet will differ than a CD-ROM. Ultimately, it is very important that once a format is chosen, that an effort is put forth to determine how the targeted audience will be viewing the final product. Testing should begin early in the process on a typical system intended for the product.

Content

Another important consideration in the pre-production phase will be to determine what types of media (graphics, text, video, etc.) will be involved. Once the purpose, audience, environment, and delivery has been determined then focus on the type of content to use should take place. Often this process is flipped and content is used to determine the purpose, audience, environment, and delivery. This can likely flaw the production of the final product, since this leads the development team to believe the content is the most important consideration and will influence the success of the final product. Content that has been created from other media can help define the previous factors, but care must be taken to realize that the final multimedia product will be different from the original content pieces.

A rough estimate will be needed of each type of media to estimate the total size of a project and what resources will be needed. Care must be

taken not to exceed the limitations of the format intended as video can quickly fill up space on a CD-ROM. Also, large or multiple graphics can make a web page's download time unbearable. It is important to develop an estimation on the intended media pieces for the total size of the product and the average screen.

Resources

Not all projects are created equal, so tailoring a crew together to complete a multimedia project is crucial to its success. A multimedia project can be a simple in-house presentation to a full scale production. Some full scale multimedia productions can have costs reaching into the millions of dollars. The development of *Wing Commander IV* from Origin Systems, Inc. reportedly cost \$12 million to produce¹⁷.

Finding the good resources to work on the project will reflect in the quality of the final product even for small productions. Also, having the right people work on the project from the start will save time and money.

I enjoy directing, but I know my limitations. There's no way that I'm going to try and write a screenplay when I can use a professional.

Chris Roberts, Executive Producer (Wing Commander series)

¹⁷ Coleman, T.L., "Is THE PRICE Of Freedom Worth \$12 Million?", Computer Gaming World, Ziff Davis, 1995
<http://www.zdnet.com/gaming/content/951116/feat1/main.html>

This is the most important part of any multimedia development, since how the pieces are presented and put together shape how well the final product will be received. This is a creative craft and not a simple process of automation, so picking the right resources will vary from project to project. Additionally important, will be how the various resources will work together, since multimedia is a collaborative effort.

Storyboard

Storyboarding is an informal process that attempts to tie all the multimedia pieces into a coherent product that meets the criteria laid out in the project plan described in the previous section. Before beginning this phase make sure that everyone is in agreement with the project plan developed and the client (if involved) has given some type of formal sign-off, so as to limit any potential disputes later in the development process. This ensures that the next step proceeds under an equal assumption by all parties.

Non-digital

A storyboard should be as a series of sketches, white board drawings or even paper scrap doodles. The moment it is put into digital form it becomes a prototype and the conceptual design has been lost to specific items like font type, image sizes and other details that distract from the creative brainstorming process.

Storyboarding is often the most difficult process within the multimedia development cycle, since the concept for a project is just an idea at this point. Converting an idea into a digital form is a large transition, since the digital form has limitations on delivery of multimedia due to space, screen size, and delivery dependence. What the storyboard should bring out is the relationship of the media types and the model of how everything will fit together. Specifics that should be worked out in the next stage of development can hinder how the various elements are created. Thus effecting the final quality of the product and reflecting the mold of the software it was created within rather than the intended concept of the project.

Collaboration

Brainstorming and preliminary storyboarding activities should generate several designs of which any could be the basis for a prototype. There are several forms of storyboards that can be developed in a multimedia project. Storyboard ideas can be developed by graphic artists, video professionals, programmers, and project leaders. However, all these forms need to be collaborated into a single storyboard idea.

The main storyboard idea focuses around the interface design usually developed by graphic designers. A consistent look-and-feel to the project is developed based on style, format, and layout. Style will focus on such items as colour, logos, etc. to give an overall theme. Format will concentrate upon what the various media pieces will contribute to the product and finally layout decides where these pieces are to be placed.

Often the storyboard process is considered just a design process, but this generalization can lead to poor concept implementations along the development cycle. Design does play a major role in creating storyboard ideas, but equally important is creating a model of the processes the user will interact. This storyboard type maps out the functionality of a project and requires the input from the programmers involved. Certain potential limitations in functionality of the software, hardware or delivery can effect the design and should be an early consideration.

Video production requires a separate type of storyboarding to consider scenes, lighting, camera, props, and acting talent. This process needs to be done early, since it will have to sync with the development of the multimedia project. What is required of the video production is to create a simple storyline to each video intended to be created for the storyboard. A specific storyboard for video production will be developed in more detail once an agreed software storyboard has been reached. However, resource estimates will be needed initially for the software storyboard to avoid budgeting problems.

The project team should evaluate each possible direction and choose one that best embodies the project plan developed earlier. A good method is to choose three different directions and have the project team approve the one to develop or to present to a client. This promotes buy-in from all the team members and gives a sense of ownership to those involved. Throughout this process, the project leader should be involved to make sure there is collaboration among the various departments. Also, the

project leader is responsible to maintain the objectives of the original project plan.

Prototype

The prototype is the first digital form created from the storyboard and it is important that sign-off on the storyboard has been reached before resources are put into motion in this stage. A prototype creates the form for the idea in the project plan and storyboard. This stage also pioneers the techniques for the production cycle described in the next chapter.

Depth-First

The depth-first or slice approach develops a particular section of the multimedia project completely before moving to another section. An example of the depth-first approach was used in the creation of “Macmillan’s Multimedia Dictionary for Children”.

The project team decided to begin with the letter *A* to examine every possible issue and concern for designing an interactive dictionary for children. Everything from interface design to animation to navigational path was explored in depth, but only for the letter *A*. By constraining the prototype to one letter, the project team could test different elements such as sound synchronization, animation sequences, and accuracy of text.¹⁸

¹⁸ Apple Computer, Inc., *Demystifying Multimedia: A Guide for Multimedia Developers*, Apple Computer, Inc., Cupertino, California 1993

Depth-first approach can help define clear stages for the prototype development. Also, difficult milestones for the project team can be worked out early in the development cycle. A drawback of this approach can lead to tunnel-vision as the development loses sight of the overall picture and concentrates too much effort on the mechanics of the project.

Breath-First

Breath-first takes the opposite approach than depth-first by building the framework for the entire project and working out the details later. This approach helps develop a solid user interface and a consistent look-and-feel to the project. This has even been used in the film industry as in the film, “Bram Stoker’s Dracula”.

The filmmaker Francis Ford Coppola created a rough videotaped version of “Bram Stoker’s Dracula” before committing final scenes to film. Doing this helped him visualize continuity and make decisions more quickly than with viewing separate daily rushes.¹⁹

Breath-first has the advantage of creating a good prototype of the storyboard, since representation of the full storyboard is the goal of this approach. Also, this approach will create an interface early that can be tested with a sample of the audience. However, the details of how the project is to be developed can be overlooked, since this is not a concern

¹⁹ Apple Computer, Inc., *Demystifying Multimedia: A Guide for Multimedia Developers*, Apple Computer, Inc., Cupertino, California 1993

of this approach. Some details may be technically impossible to develop and this could stop the development process.

Not the Final

Both depth-first and breath-first approaches have their benefits and drawbacks. Choosing between these approaches will depend on the project to develop. Many times a combination of both approaches may be required to work. Developing a breath-first prototype will communicate to the project team how everything will fit together and the depth-first approach will give confidence in completing the project to the team.

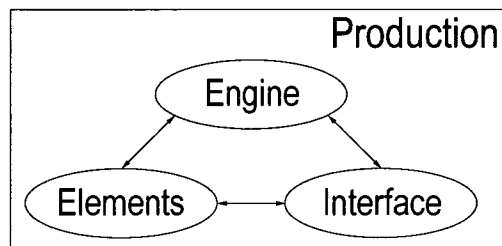
The prototype is an experimentation in which the project team explores the technology, tools, and methods to determine what will work best for the final product. It is important to note that the prototype is only a slice of the whole product, so care should be taken not make the prototype a final product. A proof of concept is when a good prototype is clear to the team that the digital form is an accurate implementation of the storyboard. Once the proof of concept has been reached, then sign-off is again required and the project will move into the product phase of the multimedia development cycle. Then work will begin on the final form of the software to be developed.

Chapter VII

Production

In the production phase of the Turbulent Pond model (Figure 6), work must be run simultaneously for the various departments. Often in the development of multimedia, it is difficult to determine where to begin. Many of the tasks at hand require the completion of other task that depend on other task and so on. The development of the engine for the multimedia project requires interface design and elements, without these it would be like building a car without a blue print or parts. The interface is difficult to create without the elements and the engine, since fit of these elements and how they perform together impacts the interface design. Finally, the elements need to be created to fit within the design of the interface and the engine.

Figure 6 - Production Phase of Turbulent Pond Model



The interrelationship of these tasks can stumble many multimedia productions at the start of this phase, so it's important to get started quickly and effectively. This would seem like an almost impossible proposal, but at the beginning of this phase a kick-off meeting can

provide the direction, motivation, and commitment from the project team.

The kick-off meeting helps pave the direction of the development and foster innovation and intercommunication among the staff. Additionally, any useful items created from the previous pre-production phase can be identified in this meeting. These provide good temporary pieces to put this interrelated product together and be replaced as these pieces are finalized in this phase.

Engine

The interface and all the elements that constitute a multimedia product must be organized into a software engine. The engine can be developed from off-the-shelf packages like Director, Toolbook or HyperCard. Also, a custom engine can be developed for a multimedia piece, but this requires additional resources to create such a product (primarily programming resources). As stated in the previous chapter, the intended environment can effect how a multimedia piece is constructed.

Off-the-Shelf vs. Custom

The various environments of single user, lab, kiosk or presentation affect how a multimedia product will be used. Each environment will require a different balance between performance, flexibility, and interaction.

Performance is a measure of speed between the user's actions and the software's response. How structured an engine is determines its

flexibility, since an engine allowing for limited programming options will restrict the creation of the product and the type of elements allowed. How the intended user will interact with the software (interface) will dictate what type of controls will be required for the engine.

Table 1 shows the relative weight of these factors in each environment. Low performance expectations in a presentation environment does not mean that the user is willing to except slow software performance. However, the table does imply that a user in a presentation environment is more likely to except performance decreases as compare to a user in the other environments.

Table 1 - Factors influencing user environments

| | Performance | Flexibility | Interaction |
|--------------|---------------|---------------|---------------|
| Single User | High | High | High |
| Lab | Medium | Medium | Medium |
| Kiosk | High | Low | Low |
| Presentation | Low | Low | Low |

The most demanding environment is the single user, since a user is weighing the purchase against other similar products. The development of this type of software is likely to be based on a custom engine. A custom engine will allow for greater performance, flexibility, and interaction as these items are only limited to the capabilities of the programming staff. Emphasis on the selection of a team in this environment would best benefit from solid programming resources.

Lab environments need to be more carefully considered as they could benefit from either a custom or off-the-shelf engine. A major distinction of this environment is the software will share the user's focus with external events. This can mask some of these factors, but poor software engine design can also distract the user from the external events. A well designed product can utilize off-the-shelf technology effectively, but a custom engine can bring capabilities that would not be possible with off-the-shelf. The pre-production phase of the model is important to help determine the best solution in this environment.

Kiosk development is similar to single user in performance requirements, since the user will not tolerate slow information retrieval on a kiosk. However, the interface and how the information is designed needs to be fairly uniform. A user in this environment is not willing to learn or become familiar with the software, so a generic design is best suited. Off-the-shelf packages are usually a good choice for an engine, since these packages have been optimized for this environment.

The presentation environment has the luxury of having the speaker as the main focus. The software plays only a supporting role and any limitation in engine design could be addressed through the speakers presentation style. The main consideration of software development in this environment will be on content organization and the presentation's ability to communicate a message(s).

Testing

Once the product has been assembled it must be immediately tested to detect system flaws or defects known as bugs. Testing is a crucial step in the production phase and must be integrated in this process from the start. A development team must agree upon a classification system for the identification of bugs with priority levels like in Table 2.

Table 2 - Sample bug types

| Code | Level | Description |
|--------------------|--------------|---|
| Show Stoppers | 1 | Critical error causes program to crash. |
| Design Flaw | 2 | Unclear interface or user unable to utilize a function. |
| Unexpected Results | 2 | Incorrect output from a user's input. |
| Inconsistency | 3 | Engine, interface, or element anomaly. |
| Program Gap | 4 | Release does not function as the project plan intended. |
| Needed Item | 5 | Item forgotten in project plan and is critical to the success of the product. |
| Wish List | 6 | Item forgotten in project plan and is NOT critical to the success of the product. |
| Future Upgrade | 7 | Item to be deferred to the next product release. |

Depending on the time frame to deliver the product will determine to what level the bugs can be addressed. Care must be taken to address the bugs levels sequentially and not fix any bugs of lower priority until each

level is complete. It is important to avoid simple fixes that take priority over more severe bug types and delay finishing the product. However, any severe bugs due to engine limitations or impossible design implementations may require a new approach. This can bring the development back to the pre-production phase to reexamine the project plan, storyboard and/or prototype.

The test process is best performed from individuals not involved in the development process to avoid any testing biases. Also, the bugs found must be well documented and confirmed by reproducing the bug again. This will avoid later confusion when trying to fix the bug. Items to track for testing include tester's name, date, bug type, tester's input, and resulting bug.

The development of the engine will iterate into several versions and classified into several categories called alpha, beta, and final release. An alpha release is when the first workable version with all the main elements, interface, and functionality are intact. This release will concentrate upon fixing major design flaws and critical programming errors. The minor bug fixes will be ignored at this stage, but will be documented for testing in the next version. Once the major issues have been fixed, the product will enter into the beta release which will cleanup the product for final release. On large productions each release category (alpha, beta, and final) can have multiple versions to address the bug fixes.

Interface

Interactivity in software means that the user, not the designer, controls the sequence, the pace, and most importantly, what to look at and what to ignore. Design of the interface must rely upon understanding human interaction, but yet be able to create an interface that can work with the technical components of the software engine and elements. A software's interface is the glue between the engine's functionality and the element's information requested by the user. Successful interface design happens when the user tells the computer what to do and not the other way around.

Consistency

Consistency is an important factor when developing software interfaces. This helps the user to become quickly familiar with a product and to successfully utilize it. Some of the factors that influence consistency will be style and layout. Creating a style to an interface will invoke a theme that helps shape the interface by using common elements. How the interface is organized on the screen is accomplished by the layout design.

Choosing a style for a product begins by establishing what style users will find consistent with the ideas represented by the content. Designing a multimedia product to teach research skills to a young audience could use the style of an archeological excavation, complete with maps, excavation "tools", "finds", and perhaps a local guide. Organizing the interaction around a style can only be useful if the metaphor is familiar, stable, and consistent.

Successful layout requires many considerations like icons, WYSIWYG, feedback, multiplicity, and the 90/10 rule. Icons should represent interactions to the user rather than typed commands. A multimedia product that requires a complex command set to operate will not last long. WYSIWYG (What You See Is What You Get) implies that the obvious input to an interface should have an obvious result and can additionally benefit from well implemented feedback controls when not obvious. Multiplicity allows a user to get at information in several ways, so as to ensure the successful interaction for the user. Finally, the 90/10 rule specifies that the most often (90%) used functions should be very simple to operate. The infrequent (10%) and sometimes critical system options should be difficult to do, thus preventing inadvertent selection.

Proof of Concept

Once the interface design has been developed it needs to be tested. The primary concern in interface testing is how closely the interface meets the original assumptions made in the pre-production phase. This type of testing is called proof of concept and is primarily done in the interface design within the production phase. Often the storyboard designs reflect much of the interface and were likely created by the same staff involved in the interface production.

Proof of concept helps the interface evolve into a finalized product through iterations of design efforts to meet the original assumptions. It is important that the production interface has not significantly deviated

from the original assumptions due to engine or element changes in production. If the interfaces created in the production phase can not meet these assumptions, then it is necessary to go back to the pre-production phase.

Elements

Text, photos, graphics, videos, and other types of data elements form the content of a multimedia piece. Creation of these elements will come from various sources and a multimedia piece can easily have thousands of these elements. Upon creation of a single element can occur multiple versions, as the element moves from original to several edited forms and finally into the final form. Managing the naming and storing of these elements is easily overlooked as this is often left to each of the element's designers to maintain.

Without a system to manage the creation and storage of these elements can quickly overwhelm any development project. The first concern of managing this process will be to have a naming convention to help identify the elements and a central source that defines these elements known as an element dictionary. Next, the team must agree upon what forms of the elements will be saved and how the versions move through each stage similar to the anthropological term known as the "Rites of Passage". Finally, where these official versions are to be stored should be managed efficiently, since location of these various elements must not be a constant drain on resources.

Element Dictionary

Often when working in teams it becomes difficult for the members to identify the various elements being developed without the help of the original element designer. In a cross-platform environment, how elements are named can adversely effect other team members' ability to use these elements. Also, the program code will require the name of these elements. To avoid any incompatibilities, an effort to coordinate the naming of these elements by the various team members must be managed.

As these elements are required by the interface designers, programmers, and other members there needs to exist a source to locate these items. Unfortunately, in a production environment that is poorly managed this will be accomplished by locating the element's designer. This is a waste of valuable resources and causes friction between team members in a high production environment. Avoiding this waste is accomplished by the creation of an element dictionary. As a designer has completed an element it is logged into a central database or a log book. Items to enter should include designer's name, date, file name, location, and a very brief description.

Adopting a generic naming convention to control name length, element type, and version can easily eliminate the bottleneck of element incompatibility. Simple agreed upon rules as common prefix and suffix codes can help this process. Common naming conventions use DOS suffixes to denote file type and 2 letter codes for element function illustrated in Table 3. Placing these agreed rules in the element

dictionary helps reinforce the validity of these rules and decrease any chances of ambiguity.

Table 3 - Sample naming conventions

| Suffix | | Prefix | |
|---------------|------------|---------------|--------------------|
| .txt | ASCII text | ur | Url or title item |
| .jpg | JPEG image | bg | Background item |
| .wav | Sound file | bt | Button item |
| .mov | Movie file | ss | Splash screen item |

Rites of Passage

Another important managerial issue to address is at the beginning of the production and defining the official forms of the elements. It should be decided early what forms of the element versions will be saved as they are created and edited. A balance will be needed to conserve disk storage within a company and having enough versions for element updates for any interface or programming changes that can occur. Items to address can be the following:

- What is an original (element's first digital form or first edited form)
- How many edited versions are maintained
- What is an acceptable draft version for interface or engine designers

Once the versions have been agreed upon how the elements transition into the next version needs to be well managed. Communication of newly

transitioned elements needs to be a streamline process, since this is a frequent event. Other team members need to be made aware of the updates for the elements to be useful. Also, care must be made not to prematurely write over critical files or deliver substandard elements. Having some type of procedure to announce and coordinate this event is the development of rites of passage a common culture process to manage. Usually, this is accomplished by having a single person in the element design department responsible for version control. This individual is then responsible for quality control and version announcements.

Storage

Storing of the data elements in a central location seems like an obvious condition, but is rarely done in today's multimedia production environment. Until recently, storage costs were considered prohibitive to a centralize system. Also, a designer feels a sense of ownership for their work and can be hesitant to share unfinished designs with the rest of the production team. Culture will play a large role in determining whether a centralize storage system will work or not. Having elements stored in a decentralized network requires a good tracking system.

Once elements are in the agreed version form they need to be readily available to all the team members. This requires the creation of a release area for these elements to be stored and retrieved. To be effective this release area needs to be located centrally and backed up on a periodic basis. Once items are moved into the release area they become locked. A

locked element can not be updated unless everyone on the team agrees upon the update and signifies another rite of passage for this event.

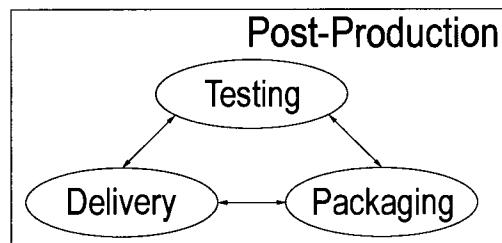
Finally, if the elements are not as originally intended in the pre-production phase due many potential factors the focus should be to reevaluate the original plan. As in the engine and interface design, the final product needs to be as expected in the pre-production phase. Going back to an earlier phase does not imply that the entire process will need to be redone and the entire production phase will need to be repeated. However, releasing an inferior product based upon flawed assumptions made early should not effect the final product.

Chapter VIII

Post-Production

The last phase of the turbulent pond model is the post-production phase (Figure 7). Post-production's primary focus is to create the final form of the product through testing, packaging, and delivery. Testing tries to determine usability of the product for the user. Packaging brings the product together to ensure receipt and acceptance by the user. Finally, how the final product gets to the targeted audience is managed in the delivery stage in this phase.

Figure 7 - Post-production Phase of Turbulent Pond Model



Testing

Testing at this phase of development is concerned with user usability rather than design functionality as in the testing steps in the production phase of the Turbulent Pond model. The focus of the post-production phase is on the final complete product and how the user will utilize the product. Production testing is primarily concerned with product performance with respect to engine, interface, and element design. The

System Usability Scale provides a testing format to evaluate a product at this stage of development.

System Usability Scale

Usability does not exist in any absolute sense; it can only be defined with reference to particular contexts. This means that there are no absolute measures of usability, since, if the usability of a product is defined by the context in which that product is used, measures of usability must of necessity be defined by that context too. There exists a need for broad general measures that can be used to compare usability across a range of contexts. In addition, there is a need for “quick and reliable” methods to allow low cost assessments of usability in multimedia product evaluation. The System Usability Scale (SUS) a reliable, low-cost usability scale developed by Digital Equipment Corporation can be used for assessments of product usability. SUS tries to score usability through three factors:

- effectiveness,
- efficiency,
- satisfaction.

Effectiveness is defined as the ability of users to complete tasks using the system, and the quality of the output of those tasks. The level of resource consumed in performing tasks is the measure of efficiency in this test.

Satisfaction is based on users’ subjective reactions to using the system.

SUS is a ten-question survey based on a five-point Likert scale. The following questions are ranked by the user on the scale ranging from “strongly agree” to “strongly disagree”.

1. I think that I would like to use this product frequently
2. I found the product unnecessarily complex
3. I thought the product was easy to use
4. I think that I would need the support of a technical person to be able to use this product
5. I found the various functions in this product were well integrated
6. I thought there was too much inconsistency in this product
7. I would imagine that most people would learn to use this product very quickly
8. I found the product very cumbersome to use
9. I felt very confident using the product
10. I needed to learn a lot of things before I could get going with this product

To calculate the SUS score, first sum the score contributions from each item. Each item's score contribution will range from 0 to 4. For the odd numbered items, scoring is 0 for “Strongly Disagree” to 4 for “Strongly Agree”. The even items are scored opposite as 4 is for “Strongly Disagree” to 0 for “Strongly Agree”. Each of the respondent’s questions are added and multiplied by 2.5 to obtain the overall value of SU. The scores have a low rank of 0 to a high rank of 100.

Packaging

Packaging within the post-production phase of the turbulent pond model is often the most overlooked and least planned for effort in multimedia. Much of the effort is focused upon identifying a product through the pre-production phase and in developing the concept in the production phase. Finally, testing and delivering a product in the last phase of post-production consume the remaining resources to finish the project.

Product design, documentation, and duplication that comprise the packaging stage within post-production are critical to the success of the target audience receipt and acceptance of a good product. Product design focuses upon designing the container packaging of a product to attract the attention of any potential users. Documentation helps the user utilize the product after purchase while evaluating its usefulness. Finally, duplication takes the product design and documentation to manufacture the final product and fill the intended channel to reach the targeted audience.

Package Design

In single-user environments package design is essential to advertise the existence of a product and attract users. Even in environments that do require competitive advantages, good package design gives the targeted audience the impression of professional product. This is the first impression a user will have of the product and will form the basis of how the user perceives of the quality.

The package design should give the user a glimpse of the product by using actual screen shots, listing product functionality and the software's purpose. This will help the user quickly accept the perceived utility of the product. Additionally, it should be clear on the package what the system requirements are for the software, so the user will not be disappointed when trying to install the product for the first time.

Package design needs some preliminary work done in the pre-production phase when trying to identify the targeted audience, but the majority of the design will be completed in this post-production phase. Actual screen shots and final system requirements will not be completed until this phase, but the software's purpose and functionality along with some design theme ideas are done in earlier phases.

Documentation

The best developed product can fail if the targeted audience does not understand how to use it from the instructions provided through the documentation. It would be ideal for any multimedia product to be so easy to operate that no additional source of information is necessary. Unfortunately, this is rarely the case and the product will require a source of instructions. Documentation is the effort to produce some form of instructions either in electronic help screens or printed manuals.

Most types of documentation appear as forms of text and diagrams, but a product can benefit from multimedia forms of instructions. Movies or voice annotation in installation or trouble-shooting procedures can be

highly informative. This can instruct an audience as effectively as providing in-person support.

The primary goal of any type of good documentation is to successfully reach the targeted audience identified in the pre-production phase. Documentation will fail when the instruction or training material begins to exceed the common skill sets of the targeted audience and requires large investments of time for the user to learn the product. A balance to provide enough useful instructions, but not overwhelm the user is required to have the product accepted by the targeted audience.

Documentation can provide more value than just how to use the product, it can describe how to use the product with examples, ideas, and background concepts. The documentation can also address common problems and misconceptions of the product. The amount of resources to produce effective and in-depth documentation should be weighed against their cost-effectiveness. However, everyone will benefit when the targeted audience use the products to their fullest capacity.

Duplication

Multimedia project proposals submitted for a Provincial grant revealed that all the CD-ROM projects had failed to correctly estimate the cost of duplication. Only 40% actually allocated the project's budget for duplication, but had severely underestimated the actual cost for duplication. This is a mistake made by many first-time multimedia projects, since this effort is often overlooked.

It is common to expect that the duplication process is a simple task that only requires the master disk to be created. However, depending on the format (usually CD-ROM) the product needs to have been thoroughly tested in the production phase in the intended format for performance issues. Then the master has to be properly setup to ensure accurate duplication through a duplication house or in-house. This process requires a fair amount of expertise and repetition, since this is rarely successful on the first try.

Duplication houses can provide useful expertise in a first time production environment and for high volume requirements. Costs for a duplication house can be high for low volume and multiple delivery environments. All duplication plants have a minimum CD-ROM order, which is typically as high as 250 units and require higher setup fees for low unit orders. In volume (usually 5,000 units or more), the cost per unit drops well under \$1 per CD-ROM. However, low unit orders can cost above \$10 per unit for the duplication alone.

CD-ROM recorders have recently dropped in price and any multimedia production environment can benefit from having a recorder in-house. A recorder's main utility allows for testing in the production cycle for performance issues. Also, it is an excellent source for very low duplication needs, but is often not a satisfactory source for duplication in most environments. One caveat with the CD-ROM recorders are that some older CD-ROM drives have trouble with the CD-Rs (CD-ROM created with a recorder).

Often miscalculation is the true cost of in-house duplication. The cost of a blank CD-R in bulk is usually around the same cost of a low unit order in a duplication plant. The expense of labor to produce the duplicated CD-Rs can easily be underestimated, since the process is fairly straight forward of just copying files. However, the majority of quad-speed recorders (units that spin at four times of music CDs to increase performance) can only write at double-speed rates. A single CD-R can take approximately 1.5 hours to duplicate as there requires setup time (10-20 minutes), duplication time (up to 45 minutes), and packaging (15-30 minutes). A simple calculation shows that just 75 CD-Rs can take an individual working full time 3 weeks to complete if there are no problems encountered.

Delivery

Delivering the final product into the hands of the customer is not an easy task. It is also the last import stage to successfully completing a multimedia production. Creating and duplicating a solid product is a phenomenal task as illustrated in the previous chapters and sections. This last stage requires a final push from the resources within a company to get that product into the market and finally receive the benefits for the effort.

How well this last stage succeeds will determine success of the product and is accomplished by managing the channel distributions, support, and follow-up. How the product is physically delivered to the targeted

audience is through channel marketing. Support will fill any gaps from product flaws to poor user implementations. Finally, follow-up will concentrate upon how well the final product is received by the targeted audience.

Channel Marketing

Getting the product to the correct targeted audience is the single most important goal for the product. Channels change rapidly and how to proceed into these markets is not always obvious. Every environment and each product will need different approaches to these channels. Marketing expertise and channel relationships are the keys to managing this area. Just creating a good product will not guarantee product success. This needs to be a well-planned effort from the pre-production phase of development and nurtured through the production phase until final product delivery in this post-production phase.

The channel market is also a good indicator for what the market demands. If this process is well managed, then any feedback from vendors, distributors, or users within the channel can help improve on the product. These sources have a vested interest into well-informed feedback, since they will benefit from a better product.

Support

Support is commonly defined as providing assistance to customers and clients in response to specific problems and inquiries. Most multimedia products will not require a significant amount of resources to support. Much of the support should be found in the documentation outlined in the previous section. If the documentation is not meeting this demand, then it is likely the documentation will need to be revised to meet the user's response to a problem or question.

A product's design may need revision if the user requires training to utilize the product. Multimedia products by design need to be relatively easy to use, but some cases may involve difficult installations like in kiosk or lab environments. Training or support will in these cases need additional resources from the development team, that the documentation may not be able to provide. However, extensive drain on development resources is a good indicator that the development process may need to rework the product in the pre-production phase of the model.

Follow-up

The development team can gain lasting insights by looking back on the project and build new opportunities by looking on to new projects. This is the confirmation for the team on early assumptions made in the beginning of the pre-production phase. Using this information allows the team to improve for the next project or iterations of this project if required. This is the last stage to wrap up the project into a final product.

Follow-up can include new ways to capitalize on past efforts through repurposing material, planning upgrades, or launching special programs to invite continued customer participation. Repurposing can entail using the engine developed for future products or separating content elements for other uses. Upgrades can extend the life of a product as technology advances and additional user needs develop. Finally, getting the customer to participate in special programs can allow the multimedia house the opportunity to improve an existing product with less effort and to better understand the market for future products.

Follow-up will always lead into the development of another project or new iteration of the current project. This brings us back to the pre-production phase of the turbulent model and the importance that the model is a continual cycle upon itself. As multimedia production teams move through this cycle, the process will streamline in some areas and change in other due to technology advancements. However, the importance of this model to multimedia is that this process is a never ending cycle that improves the quality of design for a multimedia team.

Chapter IX

Software

The task management system software provided with this thesis gives the manager of multimedia a means to gauge the progress of production. The software focuses upon tasks that comprise the stages within the Turbulent Pond model. This software is not designed to measure the performance of individual team members. Rather, the goal of the software is to give management and team members a way to gauge the progress of a project.

This chapter provides an overview of the task management system, how to use the software, and where it fits within the Turbulent Pond model. The first section gives an overview by covering the software components and system requirements. The screens are detailed in this section to show software utilization. The final section outlines the fit of the software into the model and gives a strategy on implementation.

Overview

The key to successful implementation of this task management system is to stress the importance of the tasks. The primary function of the software is to communicate task status and the amount of resources that were required to complete a task. Inter-communication between the various resources involved in a project is critical to its success, since these tasks are interrelated in the multimedia environment.

A task is a definable process deliverable within a project. A task must have a clear starting point and a tangible means to measure its completion. Definitions of tasks will vary in scope between different companies as to how detailed the tasks are to be broken-down. Tasks will vary slightly between projects at a single company, but previous metrics will help build a basis to forecast future projects.

The software is task-centric and not employee-centric. The measurement of task metrics becomes inaccurate if the software is used to gauge individual performance. If the focus is switched to employee-centric, then individual team members begin to pad hours and prematurely submit tasks as "completed". This results from employees that will not want to appear as not working to management and others. It is important that the staff believes the software will communicate task information throughout the company and not the information about themselves.

Individual employee performance in multimedia is based on quality of design, intercommunication skills, and ability to complete tasks in a timely manner. Creativity is a difficult value to measure accurately, since design is highly subjective to personal tastes. How the individual completes tasks and works within the team is often the focus of evaluation of an individual's performance. Unfortunately, it is easily mistaken to measure the amount of time an individual spends working as a measure of hours per week.

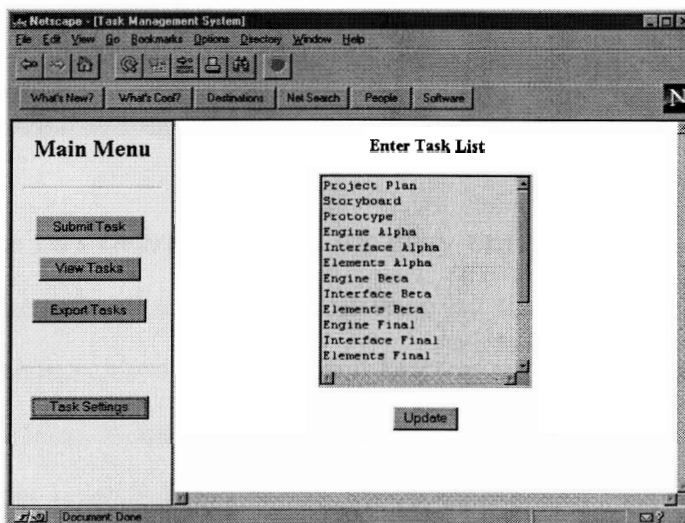
Software Components

There are three components to the software: project definition, task screen, and task reporting. The project definition component manages the various projects and tasks that comprise each project. The task screen allows team members to signify the amount of time spent on tasks. The last component, task reporting provides several formats of reports submitted through the task screen.

Project Definition

The project definition component defines the tasks within the project. This information is used to conform the task screens submitted by individual team members, so that the tasks can be compiled and managed. The tasks that define a project are maintained through a simple listing as shown in figure 8.

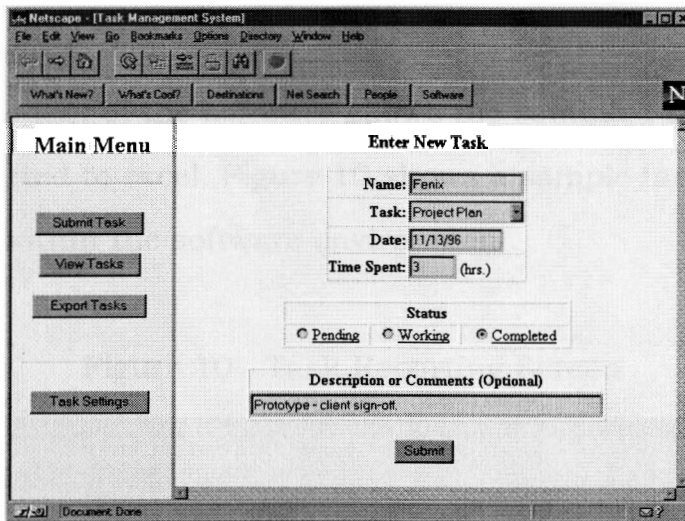
Figure 8 - Project Definition Screen



Task Screen

The task screen as shown in figure 9 is used by a team member to report work done on a task. This screen is completed as a team member moves between various tasks within a project. Submissions of tasks do not occur only when a task item has been completed, but rather as a team members efforts move to another task. The reporting functionality of the software will compile the total number of hours on a task, so it becomes more important to accurately report blocks of effort completed by individual team members.

Figure 9 - Task Screen



The screenshot shows a web browser window titled "Netcape - [Task Management System]". The browser's address bar and menu bar are visible. The main content area is divided into two sections. On the left is a "Main Menu" with buttons for "Submit Task", "View Tasks", "Export Tasks", and "Task Settings". On the right is the "Enter New Task" form. The form contains the following fields and options:

- Name:** Text input field containing "Fenix".
- Task:** Dropdown menu showing "Project Plan".
- Date:** Text input field containing "11/13/96".
- Time Spent:** Text input field containing "3", followed by "(hrs.)".
- Status:** Radio button options: Pending, Working, Completed.
- Description or Comments (Optional):** Text area containing "Prototype - client sign-off".
- Submit:** A button at the bottom of the form.

The task screen has been designed for easy entry of task information. This ensures the adoption of this software by the staff. Table 4 details the input fields of this screen. The *Submit* button will send the task data to the server. The *View* button displays a list of recently submitted tasks by the current team member.

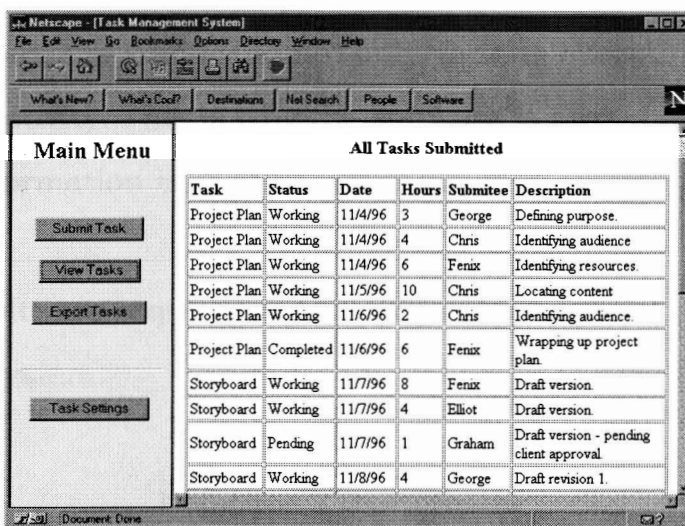
Table 4 - Task Screen Field Definitions

| Field Name | Description |
|-------------------------|---|
| Your Name | This field will default to the last entered team member's name. |
| Date | This defaults to the current date. |
| Time Spent | Number of hours on task (not cumulative). |
| Task Name | The name of the task being entered. |
| Status | Pending, working, and completed define the current state of the task. |
| Description or Comments | Additional descriptions or comments to help define the task. |

Task Reporting

This last component of the software allows the compiled tasks to be viewed on-line or exported to excel. Figure 10 shows a sample task reporting screen viewed within the software environment.

Figure 10 - Task Reporting Screen



For specific reporting requirements (i.e. by date range, by task, by employee, etc.) the data can be easily exported into excel as seen in figure 11, since the data is in a tab-delimited format.

Figure 11 - Export Task Items

| | A | B | C | D | E | F | G | H | I |
|----|--------------|-----------|----------|-------|-----------|---|---|---|---|
| 1 | Task | Status | Date | Hours | Submittee | Description | | | |
| 2 | Project Plan | Working | 11/4/96 | 3 | George | Defining purpose. | | | |
| 3 | Project Plan | Working | 11/4/96 | 4 | Chris | Identifying audience. | | | |
| 4 | Project Plan | Working | 11/4/96 | 6 | Fenix | Identifying resources. | | | |
| 5 | Project Plan | Working | 11/5/96 | 10 | Chris | Locating content. | | | |
| 6 | Project Plan | Working | 11/6/96 | 2 | Chris | Identifying audience. | | | |
| 7 | Project Plan | Completed | 11/6/96 | 6 | Fenix | Wrapping up project plan. | | | |
| 8 | Storyboard | Working | 11/7/96 | 8 | Fenix | Draft version. | | | |
| 9 | Storyboard | Working | 11/7/96 | 4 | Elliot | Draft version. | | | |
| 10 | Storyboard | Pending | 11/7/96 | 1 | Graham | Draft version - pending client approval. | | | |
| 11 | Storyboard | Working | 11/8/96 | 4 | George | Draft revision 1. | | | |
| 12 | Storyboard | Pending | 11/8/96 | 4 | Mary | Draft revision 1 - pending client approval. | | | |
| 13 | Storyboard | Completed | 11/11/96 | 0.5 | Fenix | Draft final - client sign-off. | | | |
| 14 | Prototype | Working | 11/11/96 | 8 | Chris | Interface prototype. | | | |
| 15 | Prototype | Working | 11/11/96 | 8 | Fenix | Engine prototype. | | | |
| 16 | Prototype | Working | 11/11/96 | 8 | Mary | Elements prototype. | | | |
| 17 | Prototype | Working | 11/11/96 | 8 | Fenix | Engine prototype. | | | |
| 18 | Prototype | Working | 11/12/96 | 8 | Chris | Interface prototype. | | | |
| 19 | Prototype | Working | 11/12/96 | 5 | Mary | Elements prototype. | | | |
| 20 | Prototype | Working | 11/12/96 | 3 | Graham | Elements prototype. | | | |
| 21 | Prototype | Working | 11/12/96 | 5 | Graham | Engine prototype. | | | |
| 22 | Prototype | Working | 11/12/96 | 3 | Mary | Interface prototype. | | | |
| 23 | Prototype | Pending | 11/12/96 | 4 | Chris | Prototype demo - pending client approval. | | | |
| 24 | Prototype | Pending | 11/12/96 | 4 | Fenix | Prototype demo - pending client approval. | | | |
| 25 | Prototype | Completed | 11/13/96 | 1 | Fenix | Prototype - client sign-off. | | | |

System Requirements

The task management system is a cross-platform tool based on the Java language developed by Sun Microsystems. The success of communicating between various resources within a multimedia development environment needs to transcend not only various operating systems, but forms of connectivity. This is why the software is rooted as a web-based product, so information is readily communicated.

Below is the software required to run the components of the task management system.

- Browser: Netscape 3.0 or greater
- Server Software: Support for CGI scripts written in Perl

The hardware requirements need only meet the software capabilities outlined above.

Software and Turbulent Pond

In the Turbulent Pond model, there are the three phases of development, pre-production, production, and post-production. Each of these phases is comprised of three stages. These stages can then be further broken down into deliverable tasks. The number of tasks within each stage will vary from project to project. The task management system software tracks the deliverable task items, so the progress of a multimedia project can be easily gauged at any point of the development.

The most difficult phase to manage is the production phase for most companies, since there are many processes and resources involved. This phase gains the largest benefit from the software. Development of the engine, interface, and elements involve many coordinated tasks that are interdependent. Successful communication of the status of these various tasks will influence the smooth transition of these tasks between the various resources involved. The software helps provide a quick mechanism to communicate this information throughout the staff.

Defining the tasks is best accomplished as a collaborated effort by all the team members. Having an initial meeting with the staff will develop solid task definitions, foster communication, and gain accountability from the team. The task definition screen of the software enables the definitions to

be entered into a list, so that tasks submitted will have a common data set for ease of tracking and forecasting.

Managers gain better insight for forecasting future projects, since all tasks and hours are collected. Additionally, the software allows the manager to access the various parts of the Turbulent Pond model, since multimedia projects vary in scope. For example, a future project may only require developing the engine for existing content. The software can focus upon only those tasks involved in developing past engines, thus providing a better means to forecast future projects and tasks.

Appendix

Software Source Code

Index Page (index.html)

```
<HTML>
<HEAD>
<TITLE>Task Management System</TITLE>
<FRAMESET BORDER=1 MARGINWIDTH=0 MARGINHEIGHT=0 COLS=175,*>
    <FRAME SRC="tasknav.html" NAME="navigator" SCROLLING=no>
    <FRAME SRC="task.html" NAME="screen" SCROLLING=yes>
</FRAMESET>
</HEAD>
<BODY>
</BODY>
</HTML>
```

Navigation Frame (tasknav.html)

```
<HTML>
<HEAD>
<TITLE>Task Management: Home Page</TITLE>
<SCRIPT>
<!-- This will hide the script from being displayed

// *** GLOBAL VARIABLES DEFINED HERE ***
// *** Edit the system configurations here ***

theScriptLocation = "http://www.cet.ubc.ca/cgi-bin2/";
theHomeLocation = "http://www.cet.ubc.ca/misc/";
theSettingsPassword = "Easy2hack";

// *** End of system configurations. Do not edit below this line.

function invokeCGI(theAction) {
    theScriptAction = theScriptLocation + "task.cgi?" + theAction;
    thePromptString = "To change the task settings, enter the password...";
    if(theAction == "task"){
        if(prompt(thePromptString,"") == theSettingsPassword){
            top.screen.location.href = theScriptAction;
        }
        else {
            alert("That is the incorrect password!");
        }
    }
    else if(theAction == "export"){
        theScriptAction = theHomeLocation + "task.xls";
        top.screen.location.href = theScriptAction;
    }
}
```

```

    }
    else{
        top.screen.location.href = theScriptAction;
    }
}
// -->
</SCRIPT>
</HEAD>
<BODY BGCOLOR="antiquewhite">
<CENTER>
<H2>Main Menu</H2>
<HR>
<FORM METHOD="post" TARGET="screen">
<INPUT TYPE="button" VALUE="Submit Task" onClick="invokeCGI('update')">
</FORM>
<P>
<FORM METHOD="post" NAME="viewForm" TARGET="screen">
<INPUT TYPE="button" VALUE="View Tasks" onClick="invokeCGI('view')">
</FORM>
<P>
<FORM METHOD="post" NAME="exportForm" TARGET="screen">
<INPUT TYPE="button" VALUE="Export Tasks" onClick="invokeCGI('export')">
</FORM>
<BR CLEAR="all">
<HR>
<BR CLEAR="all">
<FORM METHOD="post" NAME="taskListForm" TARGET="screen">
<INPUT TYPE="button" VALUE="Task Settings" onClick="invokeCGI('task')">
</FORM>
<P>
</CENTER>
</BODY>
</HTML>

```

New Task Page (task.html)

```

<HTML>
<HEAD>
  <TITLE>Task Submission Form</TITLE>
<SCRIPT>
<!-- This will hide the script from being displayed

// *** GLOBAL VARIABLES DEFINED HERE ***
// *** Edit the system configurations here ***

mainUrlLocation = "http://www.cet.ubc.ca/misc/index.html"
saveTaskCGI = "http://www.cet.ubc.ca/cgi-bin2/task.cgi?save";

// *** End of system configurations. Do not edit below this line.

function startUpItems() {
  theUserName = "";
  userPrefs = getCookie("userPrefs");
  if(userPrefs != null) {
    theSplitPrefs = userPrefs.split("\n");
    theUserName = theSplitPrefs[0];
    theUrlLocation = theSplitPrefs[1];
  }
}

```

```

    }

    // Set the user's name.
    document.taskForm.userName.value = theUserName;
    enterDate();
}

function buildPopup() {
    theSelectList = "<SELECT NAME='taskName'>";
    taskInfo = getCookie("taskInfo");
    if(taskInfo != null) {
        thetaskItems = taskInfo.split("\n");
        for(i = 0; i < thetaskItems.length; i++){
            (i == 0) ? (theSelected = " SELECTED") : (theSelected = "");
            theSelectList += "<OPTION VALUE='" + thetaskItems[i] + "'" +
                theSelected + ">" + thetaskItems[i] + "\n";
        }
    }
    else {
        theSelectList = "<OPTION VALUE='-1' SELECTED>Click on update!"
    }
    theSelectList += "</SELECT>";
    return theSelectList;
}

function changeUserPrefs() {
    setCookie("userPrefs",document.taskForm.userName.value);
}

function setCookie(theName, theString) {
    expirationDate = new Date();
    // Expiration of cookie set to five years!
    expirationDate.setTime(expirationDate.getTime() + (24*60*60*1000*31*60));
    document.cookie = escape(theName) + "=" + escape(theString) + "; expires=" +
    expirationDate.toGMTString() + "; path=/";
}

function getCookieVal (offset) {
    var endstr = document.cookie.indexOf (";", offset);
    if (endstr == -1)
        endstr = document.cookie.length;
    return unescape(document.cookie.substring(offset, endstr));
}

function getCookie (name) {
    var arg = escape(name) + "=";
    var alen = arg.length;
    var clen = document.cookie.length;
    var i = 0;
    while (i < clen) {
        var j = i + alen;
        if (document.cookie.substring(i, j) == arg)
            return getCookieVal (j);
        i = document.cookie.indexOf(" ", i) + 1;
        if (i == 0) break;
    }
    return null;
}

function deleteCookie (name) {

```

```

var expire = new Date();
expire.setTime (expire.getTime() - 2 * 86400001);
document.cookie = name + ".*; expires=" + expire.toGMTString() + "; path=/";
}

function enterDate() {
    if(document.taskForm.taskDate.value == ""){
        theDatetoday = new Date();
        theDate = eval(theDatetoday.getMonth() + 1) + "/" +
theDatetoday.getDate() + "/" + theDatetoday.getYear();
        document.taskForm.taskDate.value = theDate;
    }
}

function getPopupValue(objSelected){
    k = -1;
    for(i = 0; i < objSelected.length; i++){
        if(objSelected.options[i].selected){
            return objSelected.options[i].value;
        }
        else if(objSelected.options[i].defaultSelected){
            k = i;
        }
    }
    return (k != -1) ? (" " + objSelected.options[i][k].value) : null;
}

function getRadioValue(objSelected){
    k = -1;
    for(i = 0; i < objSelected.length; i++){
        if(objSelected[i].status){
            return objSelected[i].value;
        }
        else if(objSelected[i].defaultStatus){
            k = i;
        }
    }
    return (k >= 0) ? objSelected[k].value : null;
}

// This function set the radio buttons when the text is clicked.
// Netscape has a poor implementation of the user interface for
// radio buttons, so this helps fix this problem.
function setRadioBtn(whichBtn) {
    document.taskForm.taskStatus[whichBtn].checked = 1;
}

function confirmSave() {
    aConfirm =
window.open('', 'Confirm', 'toolbar=no,location=no,directories=no,status=no,scroll
bars=no,resizable=no,width=320,height=320');
    confirmWindow = aConfirm.document;
    confirmWindow.close();
    confirmWindow.open();
    theTaskData = "<HTML><HEAD><TITLE>Confirm Task
Submission</TITLE></HEAD><BODY BGCOLOUR='beige'>";
    theTaskData += "<CENTER><H3>Confirm Data:</H3><TABLE>";
    theTaskData += "<TR><TH ALIGN='left'>Name:</TH><TD>" +
document.taskForm.userName.value + "</TD></TR>";
}

```

```

        theTaskData += "<TR><TH ALIGN='left'>Task:</TH><TD>" +
getPopupValue(document.taskForm.taskName) + "</TD></TR>";
        theTaskData += "<TR><TH ALIGN='left'>Date:</TH><TD>" +
document.taskForm.taskDate.value + "</TD></TR>";
        theTaskData += "<TR><TH ALIGN='left'>Time Spent:</TH><TD>" +
document.taskForm.timeSpent.value + "</TD></TR>";
        theTaskData += "<TR><TH ALIGN='left'>Status:</TH><TD>" +
getRadioValue(document.taskForm.taskStatus) + "</TD></TR>";
        theTaskData += "<TR VALIGN='top'><TH ALIGN='left'>Comments:</TH><TD>" +
document.taskForm.jobDesc.value + "</TABLE>";
        theTaskData += "<FORM NAME='finalForm' METHOD='post' ACTION='" +
saveTaskCGI + "'>";
        theTaskData += "<INPUT NAME='userName' TYPE='hidden' VALUE='" +
document.taskForm.userName.value + "'>";
        theTaskData += "<INPUT NAME='taskName' TYPE='hidden' VALUE='" +
getPopupValue(document.taskForm.taskName) + "'>";
        theTaskData += "<INPUT NAME='taskDate' TYPE='hidden' VALUE='" +
document.taskForm.taskDate.value + "'>";
        theTaskData += "<INPUT NAME='timeSpent' TYPE='hidden' VALUE='" +
document.taskForm.timeSpent.value + "'>";
        theTaskData += "<INPUT NAME='taskStatus' TYPE='hidden' VALUE='" +
getRadioValue(document.taskForm.taskStatus) + "'>";
        theTaskData += "<INPUT NAME='jobDesc' TYPE='hidden' VALUE='" +
document.taskForm.jobDesc.value + "'>";
        theTaskData += "<INPUT TYPE='submit' VALUE='OK'> ";
        theTaskData += "<INPUT TYPE='button' VALUE='Cancel'
onClick='self.close()'>";
        theTaskData += "</CENTER></FORM></BODY></HTML>";
        confirmWindow.write(theTaskData);
        confirmWindow.close();
        self.theConfirmDialog = aConfirm;
    }
    // -->
</SCRIPT>

</HEAD>
<BODY bgcolor="#F1F1F1" text="#000000" link="#000000" vlink = "#000000" alink =
"#000000" onLoad="startUpItems()">
<CENTER><H3>Enter New Task</H3>

<FORM NAME="taskForm" METHOD="POST">

<TABLE BORDER="1" CELSPACING="0" CELLPADDING="0">
<TR>
<TD><P ALIGN=RIGHT><B>Name:</B></TD>
<TD><INPUT NAME="userName" TYPE="text" SIZE="15"
onChange="changeUserPrefs()"></TD>
</TR>
<TR>
<TD><P ALIGN=RIGHT><B>Task:</B></TD>
<TD><SCRIPT>document.write(buildPopup());</SCRIPT></TD>
</TR>
<TR>
<TD><P ALIGN=RIGHT><B>Date:</B></TD>
<TD><INPUT NAME="taskDate" TYPE="text" SIZE="12"></TD>
</TR>
<TR>
<TD><P ALIGN=RIGHT><B>Time Spent:</B></TD>
<TD><INPUT NAME="timeSpent" TYPE="text" SIZE="5"> (hrs.)</TD>

```



```

</TR>
</TABLE>
<P>

<TABLE WIDTH="300" BORDER="1" CELLSPACING="2" CELLPADDING="0">
<TR>
<TH COLSPAN="3">Status</TH></TR>
<TR>
<TD ALIGN="CENTER"><INPUT TYPE="radio" VALUE="Pending" NAME="taskStatus"><A
HREF="javascript:setRadioBtn(0)">Pending</A></TD>
<TD ALIGN="CENTER"><INPUT TYPE="radio" VALUE="Working" NAME="taskStatus"
CHECKED="true"><A HREF="javascript:setRadioBtn(1)">Working</A></TD>
<TD ALIGN="CENTER"><INPUT TYPE="radio" VALUE="Completed" NAME="taskStatus"><A
HREF="javascript:setRadioBtn(2)">Completed</A></TD></TR>
</TABLE>
<P>

<TABLE BORDER="1" CELLSPACING="2" CELLPADDING="0">
<TR>
<TH>Description or Comments (Optional)</TH></TR>
<TR>
<TD ALIGN="CENTER"><INPUT NAME="jobDesc" TYPE="text" SIZE="50"
MAXLENGTH="50"></TEXTAREA></TD></TR>
</TABLE>
<P>

<INPUT TYPE="button" VALUE="Submit" onClick="confirmSave()">
</FORM>

</CENTER>
</BODY>
</HTML>

```

Task Perl CGI (task.cgi)

```

#!/usr/bin/perl

#####
###--- GLOBAL VARIABLES HERE ---###
#####

$scriptPath = "/home/www2/cgi-bin2/";
$scriptUrl = "http://www.cet.ubc.ca/cgi-bin2/";
$homePath = "/home/www2/misc/";
$homeUrl = "http://www.cet.ubc.ca/cgi-bin2/";
$Cookie_Exp_Date = 'Wednesday, 09-Nov-1999 00:00:00 GMT';

#####
###--- DON'T EDIT BELOW HERE ---#
#####

$reqdFunction = $ARGV[0];
$dbfile = $scriptPath . "task.txt";
$xlsfile = $homePath . "task.xls";
$taskListDB = $scriptPath . "tasklist.txt";

```

```

$viewScript = $scriptUrl . "task.cgi\?view";
$taskListCGI = $scriptUrl . "task.cgi?tasklist";
$taskUpdateCGI = $scriptUrl . "task.cgi?task";
$taskForm = $homePath . "task.html";

#===== Function: Set Cookie for Projects/Tasks =====
if($reqdFunction eq "update") {
    $taskInfo = "";
    open(TASKLIST,"$taskListDB");
    while(<TASKLIST>) {
        chop;
        $taskInfo .= "$_\n";
    }
    # Get rid of the trailing newline character.
    chop($taskInfo);
    &SetCookies('taskInfo',$taskInfo);
    print "Content-type: text/html\n\n";
    open(TASKFORM, $taskForm);
    @taskFormHtml = <TASKFORM>;
    close(TASKFORM);
    print @taskFormHtml;
}
#=====

#===== Function: Save Task Data =====
if($reqdFunction eq "save") {
    # Get the form data
    read(STDIN, $bigcontent, $ENV{'CONTENT_LENGTH'});
    @content = split(/&/, $bigcontent);
    $count = 0;
    foreach $content (@content) {
        ($name, $value) = split(/=/, $content);
        $value =~ tr/+//;
        $value =~ s/%([a-zA-Z0-9][a-zA-Z0-9])/pack("C", hex($1))/eg;
        $input{$name} = $value;
    }

    $submitTask =
"\n$input{'taskName'}\t$input{'taskStatus'}\t$input{'taskDate'}\t";
    $submitTask .=
"$input{'timeSpent'}\t$input{'userName'}\t$input{'jobDesc'}";

    # Write task data to text database file
    open(DBFILE,">>$dbfile");
    print DBFILE "$submitTask";
    close(DBFILE);

    # Build HTML screen of submitted task data.
    $writeResults = "<TABLE CELLPADDING=1 CELLSPACING=1 BORDER=1>";
    $writeResults .= "<TR><TH ALIGN='left'>Task
Name</TH><TD><I>$input{'taskName'}</I></TD></TR>";
    $writeResults .= "<TR><TH ALIGN='left'>Task
Status</TH><TD><I>$input{'taskStatus'}</I></TD></TR>";
    $writeResults .= "<TR><TH ALIGN='left'>Task
Date</TH><TD><I>$input{'taskDate'}</I></TD></TR>";
    $writeResults .= "<TR><TH ALIGN='left'>Time
Spent</TH><TD><I>$input{'timeSpent'}</I></TD></TR>";
    $writeResults .= "<TR><TH ALIGN='left'>Team
Member</TH><TD><I>$input{'userName'}</I></TD></TR>";
}

```

```

    $writeResults .= "<TR VALIGN='top'><TH ALIGN='left'>Task
Description</TH><TD><I>$input{'jobDesc'}</I></TD></TR>";
    $writeResults .= "</TABLE><P>";

    $writeResults .= "<FORM METHOD='post' ACTION=$viewScript>";
    $writeResults .= "<CENTER><INPUT TYPE='button' VALUE=' Ok '
onClick='self.close()'></CENTER>";
    $writeResults .= "</FORM>";

    system("cp $dbfile $xlsfile");
    $theHeader = "Received Task Data";
    $theTitle = "Submitted Tasks";
    &resultsPage;
}
#####

##### Write to the text database #####
if($reqdFunction eq "view") {
    $writeResults = "<TABLE CELLPADDING=2 CELLSPACING=1 BORDER=1>\n";

    open(DBFILE,"$dbfile");
    while(<DBFILE>) {
        chop; # remove newline
        ($taskName,$taskStatus,$taskDate,$timeSpent,$userName,$jobDesc) =
split(/\t/);
        # First data set is the data labels for easy exporting to excel.
        if($timeSpent eq "Hours") {
            $writeResults .= "<TR><TH ALIGN='left'>$taskName</TH><TH
ALIGN='left'>$taskStatus</TH>";
            $writeResults .= "<TH ALIGN='left'>$taskDate</TH><TH
ALIGN='left'>$timeSpent</TH>";
            $writeResults .= "<TH ALIGN='left'>$userName</TH><TH
ALIGN='left'>$jobDesc</TH></TR>\n";
        }
        else {
            $writeResults .=
"<TR><TD>$taskName</TD><TD>$taskStatus</TD>";
            $writeResults .= "<TD>$taskDate</TD><TD>$timeSpent</TD>";
            $writeResults .=
"<TD>$userName</TD><TD>$jobDesc</TD></TR>\n";
        }
    }
    close(DBFILE);

    $writeResults .= "</TABLE><P>";

    $theHeader = "All Tasks Submitted";
    $theTitle = "View Tasks On-line";
    &resultsPage;
}
#####

##### Get task list #####
if($reqdFunction eq "task") {
    $theTaskList = "";

    $writeResults = "<CENTER><FORM NAME='taskForm' METHOD='post'
ACTION='$taskListCGI'>";

```

```

$writeResults .= "<TEXTAREA NAME='tasklist' ROWS='12' COLS='25'>";

open(TASKLIST,"$taskListDB");
while(<TASKLIST>) {
    chop;
    $theTaskList .= "$_\n";
}
close(TASKLIST);

$writeResults .= "$theTaskList</TEXTAREA><P>";
$writeResults .= "<INPUT TYPE='submit' VALUE='Update'>";
$writeResults .= "</FORM></CENTER>";
$theHeader = "Enter Task List";
$theTitle = "Task List";
&resultsPage;
}
#=====

#===== Update task list =====
if($reqdFunction eq "tasklist") {
    $newTaskList = "";

    # Get the new task list from the form.
    read(STDIN, $bigcontent, $ENV{'CONTENT_LENGTH'});
    @content = split(/&/, $bigcontent);
    $count = 0;
    foreach $content (@content) {
        ($name, $value) = split(/=/, $content);
        $value =~ tr/+// ;
        $value =~ s/%([a-zA-Z0-9][a-zA-Z0-9])/pack("C", hex($1))/eg;
        $input{$name} = $value;
    }
    $newTaskList = "$input{'tasklist'}\n";
    $newTaskList =~ s/\r\n/\n/g;

    # Save the new task list to the tasklist text file.
    open(TASKITEMS,">$taskListDB");
    print TASKITEMS "$newTaskList";
    close(TASKITEMS);

    # HTML Screen
    $newTaskList =~ s/\n/<BR>/g;
    $writeResults .= "<BLOCKQUOTE>$newTaskList</BLOCKQUOTE><P>";

    $writeResults .= "<FORM><INPUT TYPE='button' VALUE='Task Update Screen'
onClick=\"self.location.href='$taskUpdateCGI'\"></FORM>";

    $theHeader = "Task List Updated";
    $theTitle = "Updated Task List";
    &resultsPage;
}
#=====

#===== Create return screen =====
sub resultsPage {
    print "Content-type: text/html\n\n";
    print "<HTML>\n<TITLE>$theTitle</TITLE>\n";
    print "<BODY BGCOLOR=\"#FFFFFF\">";
}

```

```

print "<CENTER><H3>$theHeader</H3>\n";

print "$writeResults";

print "</CENTER></BODY></HTML>";
}
#=====

sub GetCookies {
    local(@ReturnCookies) = @_;
    if ($ENV{'HTTP_COOKIE'}) {
        if ($ReturnCookies[0] ne '') {
            local($cookie_flag) = "0";
            foreach (split(/; /,$ENV{'HTTP_COOKIE'})) {
                local($cookie,$value) = split(/=/);
                foreach $ReturnCookie (@ReturnCookies) {
                    if ($ReturnCookie eq $cookie) {
                        $Cookies{$cookie} = $value;
                        $cookie_flag = "1";
                    }
                }
            }
            if ($cookie_flag == 1) {
                return 1;
            }
        }
        else {
            return 0;
        }
    }
    else {
        foreach (split(/; /,$ENV{'HTTP_COOKIE'})) {
            local($cookie,$value) = split(/=/);
            # Decode the cookie data.
            $value =~ tr/+ / /;
            $value =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C",
hex($1))/eg;
            $Cookies{$cookie} = $value;
        }
        return 1;
    }
}
else {
    return 0;
}
}

sub SetCookieExpDate{
    if ($_[0] =~ /^~\w+,\,s\d\d\-\w\w\w\-\d\d\s\d\d\:\d\d\:\d\d\sGMT$/ ) {
        $Cookie_Exp_Date = $_[0];
        return 1;
    }
    else {
        return 0;
    }
}

sub SetCookies {
    local($cookie,$value) = @_;
    # Encode the data.

```

```
$value =~ s/([a-zA-Z0-9])/'%' .unpack("H*",$1)/eg;
$value =~ tr/ /+/;
# Set the cookie.
$Prepared_Cookie .= "$cookie\=$value\; ";
print "Set-Cookie: $Prepared_Cookie";
print "expires=$Cookie_Exp_Date\; path=/\; \n";

}
```

References

- America Online, Inc., "WebCrawler's Web Size", 1996
<http://webcrawler.com/WebCrawler/Facts/Size.html>
- Anderson Consulting, *Foundations of Business Systems*, The Dryan Press, Fort Worth, Texas 1991
- Apple Computer, Inc., *Demystifying Multimedia: A Guide for Multimedia Developers*, Apple Computer, Inc., Cupertino, California 1993
- Banta, G., "Internet Pipe Schemes", Internet World, 8 (10), pp. 62-70
- Beardsley, S., Warwick, B., and Rooijen, M., "The Great European Multimedia Gamble", The McKinsey Quarterly, 3 (1) pp. 178-195
- Boehm, B.W., "A Spiral Model of Software Development and Enhancement", IEEE Computer, 21 (5). pp. 61- 72
- Brown, E. and Amdur, D., "The NewMedia 500", *NewMedia*, 6 (11), pp. 34-52
- Bush, V., "As We May Think", The Atlantic Monthly, 182 (7), pp. 101-108
- Coleman, T.L., "Is THE PRICE Of Freedom Worth \$12 Million?", Computer Gaming World, Ziff Davis, 1995
<http://www.zdnet.com/gaming/content/951116/feat1/main.html>
- e-land, "The e-stats", 1996 <http://www.e-land.com/>
- Fairhead, J., "Design for Corporate Culture", A Report Prepared for the National Economic Development Council, London, England, 1988
- Freeman, R.E., *Strategic management: A stakeholder approach*, Pitman Boston, Massachusetts, 1984
- Hershey, W., "Guide", Byte, 12 (11), pp. 244-246
- Hoffman, D. and Novak, T., "Internet and Web Use in the United States: Baselines for Commercial Development", Vanderbilt Univ., 1996
<http://www2000.ogsm.vanderbilt.edu/>
- Kiamy, D., *The High-Tech Marketing Companion*, Addison-Wesley, Reading, Massachusetts, 1993
- Kidd, T., "Agents of Change", Multimedia Producer, 2 (2), pp. 54-63
- Kozel, K., "The Interactive Killing Fields", Multimedia Producer, 2 (5), pp. 42-55

- Kristoff, R. and Satran, A., *interactivity by design*, Adobe Press, Mountain View, California, 1995.
- OECD, "Design departments; a survey of the role, organisation and functioning of design departments and drawing offices in European engineering firms", Organisation for Economic Co-operation and Development, Paris, France, 1967
- Martin, M.P., *Analysis and Design of Business Information Systems*, Prentice Hall, Englewood Cliffs, New Jersey, 1995
- Mayo E., *The Human Problems of an Industrial Civilization*, Viking Press, New York 1960
- O'Flaherty, D., "Communications Breakdown", Internet World, 8 (10), pp. 47-52
- Porter, M., *Competitive Advantage: Creating and Sustaining Superior Performance*, The Free Press, New York, 1985
- Roberts, J., "Clientology, Part II", Multimedia Producer, 2 (8), pp. 27-28
- Rockart, J.F., "Chief executives define their own data needs." Harvard Business Review, 57 (4), pp. 81-93
- Royce, W.W., "Managing the Development of Large Software Systems: Concepts and Techniques", Proceedings of IEEE WESCON, 1970
- Sony Corporation, "Now Everyone Can Experience the Internet", 1996
<http://www.sel.sony.com/SEL/webtv/index.html>
- Steiner G.A., *The Creative Organization*, The Graduate School of Business University of Chicago, 1965