

DESIGNING A HYPERMEDIA ENGINE

77303

by
Bahadır ONAY

TC.YÜKSEKÖĞRETİM KURULU
DÖĞÜŞÜM VE İNNOVASYON MERKEZİ

September, 1998

İZMİR

DESIGNING A HYPERMEDIA ENGINE

**A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of
Dokuz Eylül University
In Partial Fulfillment of the Requirements for
the Degree of Master of Science in Computer Engineering**

**by
Bahadır ONAY**

77303

**September, 1998
İZMİR**

M.Sc THESIS EXAMINATION RESULT FORM

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.



Assoc. Prof. Dr. Alp Kut
(Advisor)

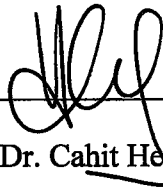


.....
(Committee Member)



.....
(Committee Member)

Approved by the
Graduate School of Nature and Applied Sciences



Prof. Dr. Cahit Helvacı
Director

ACKNOWLEDGMENTS

I want to thank Mr. Alp Kut for his great support in my project. I also thank Mr. Esen Özkarahan for giving me fundemantal concepts of computer engineering. And at last Fevzi Karaman... I am glad to know such a perfect man. He always supported me in this project with his friendship and knowledge.

Bahadır ONAY



ABSTRACT

Hypermedia is a very popular topic in 90's. We first heard the term of hypertext (especially HTML). As known hypertext means nonlinear text. When you click on a linked word in hypertext you reach different parts of this document or you may reach another document. It is a very user friendly and easy-to-use method for users.

This project concerns the new face of hypertext: Hypermedia. It is more capable than hypertext because it has 4 media. These are: Text, picture, sound and movie. This sounds good but it is a big problem to create hypermedia document and shape it as a easy-to-use document. The main objective in this project is to design a hypermedia engine. I understand that from the term hypermedia engine: Hypermedia engine is a complete tool for create and browse hypermedia documents. As can be seen from definition it has two basic tasks. Editing and browsing. So hypermedia engine consists of two tools. These are: Hypermedia editor and hypermedia browser.

Hypermedia editor gives user an environment to create hypermedia documents. It allows adding links to any word in the document. This links can be in different types. After creating hypermedia document, hypermedia browser allows users to browse this document.

CONTENTS

	Page
Contents.....	V
List of Figures.....	VIII

Chapter One

INTRODUCTION

1.1. Hypertext Technology	1
1.2. Description of Hypertext.....	1
1.2.1. Definition of Hypertext.....	1
1.2.2. Non-linearity of Hypertext.....	2
1.2.3. Advantage over Linear Texts.....	3
1.3. Use of Hypertext	3
1.3.1. The Author and the Reader.....	3
1.3.2. Purpose of Hypertext.....	4
1.3.3. Classification of Information.....	4
1.4. Limitation of Hypertext.....	5
1.4.1. Experience.....	5
1.4.2. Lost in Hyperspace.....	5

Chapter Two

HYPERMEDIA TECHNOLOGY

2.1. History of Hypertext and Hypermedia.....	7
2.2. Hypermedia Technology.....	8

Chapter Three

USING HYPERMEDIA TECHNOLOGY

3.1. Uses for Hypermedia and Hypermedia Engine.....	15
3.2. Hypermedia Glossary.....	16
3.3. Multimedia, Hypertext and Hypermedia in Education.....	21
3.3.1. Educational Impact.....	22
3.3.2. Potential For Staff Development.....	22
3.3.3. Survey Results On The Demand For This Training Package.	22

Chapter Four

RELATED WORKS

4.1. Related Works.....	23
-------------------------	----

Chapter Five

OVERVIEW for HYPERMEDIA ENGINE

5.1. General Description of Hypermedia Engine.....	27
5.2. Editor of Hypermedia Engine	28
5.3. Browser of Hypermedia Engine	30

Chapter Six

METHODS USED in HYPERMEDIA ENGINE

6.1. Creating Hypermedia.....	33
6.2. Displaying Hypermedia.....	34
6.3. Creating and Displaying Link to Text.....	36
6.4. Creating and Displaying Link to Picture.....	37
6.5. Creating and Displaying Link to Sound.....	37
6.6. Creating and Displaying Link to Movie.....	38

Chapter Seven

CODES for HYPERMEDIA ENGINE

7.1. Code for Hypermedia Editor.....	39
7.2. Code for Hypermedia Browser.....	45

Chapter Eight

USER GUIDE for HYPERMEDIA ENGINE

8.1. Using Hypermedia Editor.....	62
8.2. Using Hypermedia Browser.....	66
REFERENCES.....	70

LIST OF FIGURES

	Page
Figure 5.1 Hypermedia Editor.....	29
Figure 5.2 Hypermedia Browser.....	31
Figure 8.1 Hypermedia Editor.....	62
Figure 8.2 Adding Link to Text.....	63
Figure 8.3 Adding Link to Picture	64
Figure 8.4 Adding Link to Sound.	65
Figure 8.5 Adding Link to Movie.	66
Figure 8.6 Hypermedia Browser.....	67
Figure 8.7. Hypermedia Browser with Link to Picture.....	68
Figure 8.8. Hypermedia Browser with Link to Movie.....	69

CHAPTER ONE

INTRODUCTION

1.1 Hypertext Technology

Hypertext (non-linear language text) is a concept of a writing surface which is not constrained to be linear. It is composed of blocks of words or images electronically linked by multiple paths. In hypertext terminology, a node is anything which can sensibly be presented as a unit of information, while a link is a means of accessing nodes. In linear, printed documents, authors provide information hierarchically and readers follow the structure. Hypertext, however, possesses no single hierarchical-linear structure. A program called browser allows readers to jump from one place to the next, or make changes, or add new connections in the text. Thus, the reader becomes a co- author, determining the structure of the text for himself or for the next reader.

Hypertext is used for course materials and composition media. The World Wide Web (WWW) is one of the projects which use the concept of hypertext. Since hypertext creates new document forms, it is difficult to classify information types.

For the future, readers need to develop reading strategies and manipulation techniques in order to avoid misinterpretation and being lost in hyperspace. Authors need to learn appropriate skills to hypertext.

1.2. Description of Hypertext

1.2.1 Definition of Hypertext

Nowadays, we find ourselves using computers that provide a new writing surface that requires conventions different from those of the printed and typed pages. One of

the examples is hypertext -- non-linear language text. Hypertext is not a product but a concept. Roland Barthes calls hypertext "text composed of blocks of words (or images) linked electronically by multiple paths, chains, or trails in an open-ended, perpetually unfinished textuality described by the terms link, node, network, web, and path" (Landow 185). Simply stated, hypertext is a form of electronic text that consists of nodes, or chunks, of information and links between them. A node can be a fragment of music, a piece of text, a map, or a complete film -- anything which can sensibly be presented as a unit of information. A link is a means of accessing any two nodes in some way. A group of related nodes or a series of links is called a path. The idea of nodes and links is very general, and there are no 'rules' about how big a node should be or what gets linked to what.

1.2.2. Non-linearity of Hypertext

Hypertext has the 'non-linearity' of the node and link structures. Hypertexts have been favorably compared to the fixed, linear formats of both paper-based and standard electronic texts, such as a wordprocessor. Linear structures are typically seen as being constrained for both authors and readers, and an inferior way of presenting textual information. The author provides information hierarchically and the reader explores the information in a fixed path -- from page to page and from chapter to chapter.

The connections of hypertext, on the other hand, are organized into paths that make operational sense to authors and readers. The reader is able to jump around easily in the text, "choosing to view an image, or to go to different content (e.g. to the start of the next chapter) or to other related materials (e.g. cross-reference)" (Wright and Lickorish 90). When the author provides these jumping opportunities, they are usually accompanied by signals that indicate the availability of a jump and may also suggest where the jumper -- reader -- will land. These signals are called browsers. Since there are a number of ways of accessing of data, "browsers provide maps or overviews of the structure and content of a network of nodes and links so that the reader can read hypertext in more efficient and less confusing ways" (Barrett

94). Hypertext also offers multiple 'layers' of information, from general to specific, depending on the levels of the users' needs. As Duncan suggests, "a hypertext system can be both dynamic and interactive in a way that linear text cannot -- the user can explore a knowledge base in ways not previously determined by the system" (157).

1.2.3. Advantage over Linear Texts

Bolter foresees the future of computer technology, saying "many, perhaps most, of these texts will someday cease to be printed and will instead be distributed in electronic form" (2). Lory Hawkes, Senior Professor of General Education Department at DeVry Institute of Technology, says that the advantage of hypertext over linear texts is "cost" (interview). "More companies," she says, "will opt for a paperless system. When companies adopt a paperless environment, they hire college students who know how to use software which in turn causes a change in curriculum to accommodate the need." The shift from print to the computer will change the method of producing documents and contribute to reducing the cost of the production.

1.3. Use of Hypertext

1.3.1. The Author and the Reader

Generally, those who create, gather, and arrange hypertexts are called authors, while those who view, combine, or manipulate hypertexts are referred to as readers. However, a hypertext system offers reader and author the same environment. Hypertext readers, confronting structural choices established by the author, are usually allowed to make changes or add their own materials or links to hypertexts. The readers thus "determine its conduct for themselves, and often for successive readers and in a very real sense write (or rewrite) hypertexts" (Joyce 20). This increase in reader determinacy suggests a breakdown of the traditional reader/author distinction.

1.3.2. Purpose of Hypertext

Since hypertext has been used in a variety of ways -- as a program, a writing style and a synonym for multimedia presentation, hypertext will mean any software that has a dual nature. One is a computer program of electronically organized text that may include pictures and symbols. The other is a computer-generated writing medium that enables people to use the text for their own purposes, whether that involves amending, deleting, or regrouping the existing text.

According to "Using Hypertext in the Classroom," computer programs for hypertext projects in the Computer Writing & Research Lab (CWRL) classroom are "Storyspace (Macintosh), Toolbook (IBM), Hypercard (Macintosh), and so forth" (5). In a composition classroom, hypertext has been used to produce proposal presentations and multimedia research projects. In a literature course, students collect a number of related poems and contextual information. In addition to reading a text, they listen to excerpts from actors or view scenes from performances. The World Wide Web (WWW) is also a project that uses a concept of hypertext. The TCU Netscape and the HELP menu of the Windows are other examples available on the TCU campus.

1.3.3. Classification of Information

As we categorize printed documents such as books, newspapers, magazines, and reports, there are various ways of classifying them. Obvious distinctions can be drawn between fiction and non-fiction, technical and non-technical, and serious and humorous, and so forth. Such discriminations, however, are not based on how the text is used or the readers' view of the contents. McKnight et al. developed a technique known as repertory grid analysis to describe readers' view of hypertext. They conclude that "readers conceive of text in terms of three characteristics: how they are read, why they are read and what sort of information they contain. For example, a novel is likely to be read serially (how), for leisure (why) and contain general or non-technical information (what)" (53). It is not necessary for readers to

perceive any particular text in the same way. For example, two readers may see a particular text differently and a reader may view a text differently depending on their information needs at any one time. Thus, users need to analyze the how, why and what questions in detail when they design and build a hypertext version.

1.4. Limitation of Hypertext

1.4.1. Experience

According to "Using Hypertext in a Classroom," a completed hypertext requires the user very little knowledge, while the degree of knowledge the user need to produce successful hypertext ranges "from very little (e.g. Storyspace) to quite a lot (e.g. Macromind Director)" (5). At the most basic level, both authors and readers need to gain experience of hypertext. "There is a learning curve which requires one to spend at least a year to become expert" (Hawkes interview). Most people's experience of electronic documents via word processors may not be an appropriate model for hypertext and may cause negative transfer. Similarly, authors might have developed skills not totally appropriate to hypertext and will have to learn new skills in direct interaction with the computer. When users interact with hypertext, "the most useful technique -- really a concept -- is to understand that there is an interplay of writer's expression and software implementation" (Hawkes interview).

1.4.2. Lost in Hyperspace

It is clear that insufficient experience of readers and authors leads to disorientation or becoming lost in a display network. Elm and Woods describe this situation as "the user not having a clear conception of the relationships within the system, or knowing his present location in the system relative to the display structure, and finding it difficult to decide where to look next within the system" (Edwards & Hardman 105). Elm and Woods regard this disorientation as degradation of user's performance rather than a subjective feeling of being 'lost.' In order to prevent the user from getting lost, they define navigation skills that the user needs: "the ability to generate specific routes as the task demands; the ability to generate new routes as skillfully as familiar

ones; the ability to develop a concept of 'here' in relation to other places" (Edwards & Hardman 106).



CHAPTER TWO

HYPERMEDIA TECHNOLOGY

2.1. Hypertext Technology

1945 Vannevar Bush proposes Memex

1965 Ted Nelson coins the word "hypertext" (later elaborated in his pioneering book Literary Machines)

1967 The Hypertext Editing System and FRESS, Brown University, Andy van Dam

1968 Doug Engelbart demo of NLS system at FJCC

1975 ZOG (now KMS): CMU

1978 Aspen Movie Map, first hypermedia videodisk, Andy Lippman, MIT Architecture Machine Group (now Media Lab)

1984 Filevision from Telos; limited hypermedia database widely available for the Macintosh

1985 Symbolics Document Examiner, Janet Walker

1985 Intermedia, Brown University, Norman Meyrowitz

1986 OWL introduces Guide, first widely available hypertext

1987 Apple introduces HyperCard, Bill Atkinson

1987 Hypertext'87 first major conference on hypertext

1991 World Wide Web at CERN becomes first global hypertext, Tim Berners-Lee

1992 New York Times Book Review cover story on hypertext fiction

1993 Mosaic anointed Internet killer app, National Center for Supercomputing Applications

1993 A Hard Day's Night becomes the first full-length feature film in hypermedia (originally for Macintosh; now also available for Windows)

1993 Hypermedia encyclopedias sell more copies than print encyclopedias

1995 Netscape Corp. gains market value of almost \$3G on first day of stock market trading

We can summarize this history as follows:

1. Vannevar Bush is credited with one of the earliest descriptions of hypermedia [3]. In 1945, he envisioned a machine that would support browsing and making notes in a large on-line text and graphics system.
2. Inspired in part by Bush, Douglas Engelbart developed one of the first computer-based hypermedia systems (NLS) in the 1960's. [6]
3. The terms "hypertext" and "hypermedia" [10] were coined by Ted Nelson [11], whose Xanadu system aimed to provide easy access to large amounts of literature.
4. In the 1970's and 80's, researchers at Brown University [7, 13] and Carnegie Mellon University [2, 9] (as well as other hypermedia research projects) engaged in long-term programs to study the hypermedia approach to human-computer interaction. KMS is based on the work done on ZOG, its predecessor, developed at Carnegie Mellon beginning in 1972.
5. In 1990, the Association of Computing Machinery formed SILINK, an ACM special interest group on hypertext and hypermedia. SILINK is the organizer of one of the principal conference series on this technology ("Hypertext XX") which alternates years between Europe and the United States.
6. In the early 1990's, the World Wide Web was developed and, along with Mosaic and other web browsers, greatly popularized the concept on hypermedia and its use on the Internet.

2.2. Hypermedia Technology

Hypermedia is a superset of hypertext. Hypermedia documents contain links not only to other pieces of text, but also to other forms of media - sounds, images, and movies. Images themselves can be selected to link to sounds or documents. This means that browsers might not display a text file, but might display images or sound or animations. Hypermedia simply combines hypertext and multimedia.

Some examples of Hypermedia might be:

- You are reading a text on the Japanese language. You select a Japanese phrase, then hear the phrase as spoken in the native tongue.
- You are viewing a company's floor plan, you select an office by clicking on a room. The employee's name and picture appears with a list of their current projects.
- You are a law student studying the California Revised Statutes. By selecting a passage, you find precedents from a 1920 Supreme Court ruling stored at Cornell. Cross-referenced hyperlinks allow you to view any one of 500 related cases with audio annotations.

Hypermedia is an environment that combines several media: text, graphics, sound and video, and supports communal nonlinear 'writing' and reading. This environment is characterised by networks of nodes that contain information in the form of text, pictures, soundfiles, audio or video files that are connected by links. Hypertext is a text only version.

Hypertext may be defined as "interlinked electronic text", hypermedia as "an interlinked collection of materials that include a mixture of electronic media". The course materials you are now using illustrate both, since some images are involved. Inter-referentiality is, of course, coeval with text itself, and the interlinking of texts is an ancient technique found, for example, in glosses, marginalia, and footnotes. Nevertheless, the kind of linking made possible by software, and its extension to media other than text, is new and has consequences worthy of our attention.

The term "hypertext" (predecessor and basis of the later, broader notion "hypermedia") seems to have been coined by Theodore Nelson some time in the 1960s. Nelson, who still works on the development of a very ambitious hypermedia system (see Xanadu), was inspired by Vannevar Bush's essay "As We May Think", which appeared in *The Atlantic Monthly* in 1945. In his article, U.S. President Roosevelt's advisor envisaged a then non-existent but not totally utopian machine that could store, index, retrieve and interconnect large amounts of information. This

hypothetical machine is, in fact, very similar in its functionality (not in its appearance) to the World-Wide Web.

During the second half of the 1980s, several stand-alone hypermedia products for personal computers appeared on the market. They have been followed by dozens of similar programs for all operating system platforms. But the event that had the greatest impact on the development and also the popularization of hypermedia happened in March 1989, when the World-Wide Web project started at the European Laboratory for Particle Physics (CERN) in Geneva.

Detailed information on the history of and the concepts behind hypermedia can be found on the WWW, including material on the Web itself, texts by Theodore Nelson, and Bush's journal article.

Hypermedia on the WWW represents a specific approach to the storage and retrieval of computer documents, the so-called client-server solution. Browser programs like Mosaic or Netscape act as clients which request and receive material from servers, other programs that reside on remote computers. This separation of service provider and service requester lends itself very naturally to such a globally distributed data bank as the Web.

But there is also a different type of hypermedia software which combines server and client in one and the same program on a local computer. One could say that the WWW is an extension of this older (but still not obsolete) way of creating and using hypermedia applications. The Macintosh's HyperCard, mentioned earlier, was the first such program not restricted to text and available to a large number of computer users. Nowadays, there exist various comparable programs for nearly every operating system. On the Mac, Hypercard and its extensions have been joined by SuperCard, Storyspace and AuthorWare among others. For DOS/Windows, there is ToolBook, Guide, Microcosm and AuthorWare, to name a few.

These programs enable their users to create hypermedia documents and to peruse them, sometimes also the reading of documents produced by other programs is possible, either directly or after a conversion. Some programs come in two versions: one supporting both preparation and browsing of documents, the other (cheaper) one working only as a viewer for the file format in question.

Multimedia in: Medium:	Information Contents	Human-Computer Interaction	Human-Human Communication
Text	Detailed and informative (technical) descriptions, narration	Short or detailed technical message, indications for (meta-)information	Long, exact descriptions (letters, email), 'secret' information
Graphics	Overviews and diagrams, visualization of spatial and multidimensional data, comparative information	Orientation cues, visual effects, sense of spatial relationships, artistic pictures, thematic cues	Schemes plans, maps, spatial instructions
Images	Static objects and structures of the real world	Thematic cues and visual effects	Recognition and verification of the communication partner
Video	Descriptive visualizations of dynamic objects and processes of the real world, entertaining effects	— (currently not used)	Mimics, gestures, feelings
Animation	Dynamic objects and processes of imaginary worlds, visualizations of complex structures	Visualization of long or continuous computer processes, 'microns' (moving icons)	— (currently not used)
Audio	Additional, exact descriptions of other media, persuasive information, entertainment	Warnings, signals for process status info, "carcons" (auditory icons), auditory textures and cues	Exact descriptions, advice, emphasis, feelings

Uses of different media with respect to different components of application systems.

Since the emergence of the concept of hypermedia engines several approaches have been suggested. Some (HAM, HyperBase (GMD-IPSI), and HyperBase (Aarlborg University)) have been labeled as "hyperbases of the HAM generation" [31] since these approaches have main concepts in common with the HAM. Several other approaches have emerged, too, introducing different concepts. In this section, we will take a look at some of the most important ones with respect to the task of application development.

Hypertext is text that is not constrained to be linear. Books, for example, are a linear sequence of pages. In contrast, hypertext organizes information as an interconnected web of linked text. Different paths can be followed through the work by different readers; readers can choose, among all the links the authors provided, those associations most relevant to their immediate needs.

Hypermedia refers to hypertext applications that contain things other than text objects. Hypermedia applications encompass images, video, sound, and more. The Hypertext Markup Language contains markup tags for specifying links to multimedia objects. How these objects are displayed is left up to the browser, but generally, images are expanded as illustrations or figures within the text, while sound and animation are presented in their own windows with stop and play controls.

Links from an HTML hypertext page appear as highlighted text, usually blue and underlined. The text itself is called the anchor of the link and can be embedded in other HTML elements, like lists and tables. Images can be anchors as well as text. Small images can be used as clickable icons. This is useful for creating navigational controls that appear on a series of Web pages. Images can also have defined sub-areas; each an anchor linking to somewhere else. Such images are called *imagemaps* and are quite handy for organizing spatially related data.

The wonderful thing about hypertext is that it adds an extra dimension of structure to the content of your work. With hypertext, you can highlight alternative relationships in the text besides the linear ordering of sections, chapters, and

subchapters found in the table of contents. The size of a hypertext work is limited only by the physical storage space available. Since the World Wide Web is on and of the Internet, this means that terabytes of data are organized and given structure by the Web. Because it's growing faster than any one person could possibly keep up with, your experience is that of "surfing" through an unbounded information space.

The concepts of hypertext and hypermedia have been around for a while; Ted Nelson is given credit for coining the terms in 1965. One of the first practical hypermedia applications was the Aspen Movie Map done at MIT in 1978. It used videodisc and touch-screen technology. Filevision from Telos, released in 1984, gave hypermedia databases to early Macintosh users. In 1987, Apple introduced Hypercard, written by Bill Atkinson, which incorporated many hypermedia concepts. Windows users take advantage of a hypertext based help system. The development of CD-ROM drives for personal computers made the commercial development and marketing of multimedia applications a reality. And in 1989, Tim Berners-Lee and Robert Cailliau submitted a proposal to their colleagues at CERN for a client-server-based hypermedia system, and the World Wide Web was born.

HTML hypermedia applications are similar in many ways to Macintosh Hypercard applications; enough so that it's possible to mimic simple Hypercard applications in HTML and vice versa. Both systems take the form of a web of linked nodes with one node designated as home-the home stack for Hypercard and the home page for a Web server. The differences, however, are significant; not the least important of which is that Hypercard applications only run on Apple Macintosh computers, whereas HTML applications run on networks connecting a variety of computers.

Unlike a Hypercard card, an HTML page can be many physical pages long, corresponding to a chapter in a book or a section of a manual; however, the width is variable-the browser presenting the page word-wraps the text and positions the images to fit the width of the display window. HTML hypertext links are activated by clicking with a mouse button on highlighted text or image. The link may be to a

point in the text on the same page, to a new Web page, or to some other object or resource on the network.

Another key difference between other hypermedia and HTML/Web applications is that the former are designed for running on individual personal computers-every user has their own copy. HTML documents, on the other hand, exist in a client/server environment where one copy serves all. The clients are the readers'browsers. The servers are programs running on remote computers that provide the Web pages requested by the browsers. Because media (paper and plastic) doesn't have to be physically reproduced and distributed each time information is added or changed, Web pages can be updated quickly at low cost.

For an HTML application to be "on the Web" means that the HTML files and other documents that make up the application must reside in a directory that is accessible to a server. Usually a Web server, however, non-interactive HTML files can reside on ftp or gopher servers, too. This does not mean there must be a link from some existing Web page to your document in order for your document to be part of the Web. There is a unique address, called a Uniform Resource Locator (URL), for practically every file and resource on the Internet. There are URL formats for resources such as Gophers, Wide Area information Servers (WAIS), ftp archives, and USENET newsgroups. Since most browsers have the ability to load any URL entered by the user, just about anything on the Internet is also on the World Wide Web.

CHAPTER THREE

USING HYPERMEDIA TECHNOLOGY

3.1. Uses for Hypermedia and Hypermedia Engine

Business:

- Product catalogs and advertisements
- Organisational charts and policy manuals
- Annual reports and orientation guides
- Resumes and biographies
- Treaties, contracts, and wills
- Newsletters and news magazines
- Software documentation and code

Information resources:

- Encyclopedias, glossaries, and dictionaries
- Newspapers, Newsletters, Specialized information sources
- Medical and legal reference books
- Religious and literary annotations
- College catalogs and departmental guides
- Travel and restaurant guides
- Scientific journals, abstracts, and indexes

Personal learning:

- Instruction and exploration

- Repair and maintenance manuals
- Time lines and geographical maps
- Online help and technical documentation
- Cookbooks and home-repair manuals
- Mysteries, fantasies, and jokebooks
- Hypernovels and hyperpoems

3.2. Hypermedia Glossary

This is a glossary of terms used within the hypermedia technology. In most cases, their use corresponds to conventional use in hypertext circles.

Anchor

An area within a the content of a node which is the source or destination of a link. The anchor may be the whole of the node content. Typically, clicking a mouse on an anchor area causes the link to be followed, leaving the anchor at the opposite end of the link displayed. Anchors tend to be highlighted in a special way (always, or when the mouse is over them), or represented by a special symbol. An anchor may, and often does, correspond to the whole node. (also sometimes known as "span", "region", "button", or "extent").

Annotation

The linking of a new commentary node to an existing node. If readers can annotate nodes, then they can immediately provide feedback if the information is misleading, out of date or plain wrong. Thus the quality of the information in the web can be improved. (More...)

Authoring

A term for the process of writing a document. "Authoring" seems to have come into use in order to emphasise that document production involved more than just writing.

Back link

A link in one direction implied from the existence of an explicit link in the other direction. See: Building back-links

Browser

A program which allows a person to read hypertext . The browser gives some means of viewing the contents of nodes, and of navigating from one node to another.

Button

An anchor which is the source of a link . Often, but not always, represented on screen to look like a push-button.

Card

An alternative term for a node in a system (e.g. HyperCard, Notecards) in which the node size is limited to a single page of a limited size.

Client

A program which requests services of another program. Normally, the browser is a client of a data server.

Cyberspace

This is the "electronic" world as perceived on a computer screen, the term is often used in opposition to the "real" world. With Web-extensions like VRML and the Cyberspace Protocol, Virtual Reality will one day come to your home computer.

Database

We have used this vaguely as a term for a collection of nodes. We imagine management information for one of these being kept in one place and all being accessible by the same server. Links outside this are "external", and those inside are "internal". We do not imply anything about how the information should be stored.

Daemon

A program which runs independently of, for example the browser . Daemons may perform various management tasks such as building indexes, overviews, and back-links. Under unix, "daemon" is used for " server ", because servers normally run independently.

Document

A term for a node on some systems (eg Intermedia). Sometimes used by others as a term for a collection of nodes on related topics, possibly stored or distributed as one. The preferred term in W3 documentation.

Domain

We have used this specifically for a unit of protection. It could possibly correspond to a database , and in that case would be a better (less vague) term for it.

External

A link to a node in a different database. See Database

Host

A computer on a network. We use this term rather than the term " node " which is often used for a document in a hypertext web .

Hypermedia

MultiMedia Hypertext . HyperMedia and HyperText tend to be used loosely in place of each other. Media other than text typically include graphics, sound, and video. (More...)

Hypertext

Text which is not constrained to be linear. (More...)

Index

Something which points at other data; a server facility which provides pointers to particular data as a function of a query; a table of contents of a book in hypertext form. (More).

Internal

A link to a node in the same database . See database .

Link

A relationship between two anchors , stored in the same or different database . See "Internal" and "External" .

Navigation

The process of moving from one node to another through the hypertext web . This is normally done by following links . Various features of a particular browser may make this easier. These include keeping a history of where the user has been, and drawing diagrams of links between nearby nodes. (More...)

Node

A unit of information. Also known as a frame (KMS), card (Hypercard, Notecards). Used with this special meaning in hypertext circles: do not confuse with "node" meaning "network host". For user's benefits, we use the term " document " as this is the nearest term outside the hypertext world.

Protection

The prevention of unauthorized users from reading, or writing, a particular piece of data. Also known as "authentication", "access control", etc. (More...)

Path

An ordered set of nodes or anchors which represent a sequence in which a web can be read. A path may represent the sequence a reader actually used, or may be a sequence recommended to the reader by the author.

Reader

We have used this term for the person who browses, to distinguish him/her from

Reader

We have used this term for the person who browses, to distinguish him/her from the program (browser) which (s)he uses.

Server

A program which provides a service to another, known as the client . In a hypertext system, a server will provide hypertext information to a browser . See also: daemon .

Tracing

The automatic finding of nodes by automatic navigation . Examples might be finding all nodes dependent on another node, all people interested in a given node, all modules which use a given module. Another example is a trace starting with more than one node, such as to find a node in common between two groups, or path linking two nodes.

Topology

The allowable connectivity between nodes, anchors and links: for example, 1-1 or many-1 mappings. (More...)

Versioning

The storage and management of previous versions of a piece of information, for security, diagnostics, and interest. This is important when many users are allowed to edit the same material. (More...)

VRML

Virtual Reality Modeling Language. The term "VRML" had been coined by Dave Ragget at the 1st WWW Conference in Geneva, May 1994. VRML is proposed as a logical markup format for non-proprietary platform independent VR.

Web

A set of nodes interconnected by links . Often, the set of all the nodes which are interconnected. See also Topology.

3.3. Multimedia, Hypertext and Hypermedia in Education

The concept of combining text, graphics, sound, animation and video around a computer is commonly called 'Multimedia' . Multimedia has the potential to extend the information technology methods previously established by interactive learning modules in teaching. 'Hypertext' is a software method permitting the storing and linking of text in logical ways such that the user can freely access information when and where required. Hypertext programs are ideal for controlling multimedia packages. We use the term 'Multimedia' to include the use of Hypertext and other interactivity with Multimedia (although this combination is sometimes called 'Hypermedia').

Multimedia, used innovatively, has strong attractions for both teachers and students of all levels across a wide variety of subjects. Teachers can design learning materials for interactive use by single individuals or groups of students. Interactive modifications of the teaching system during use, permits matching of the student's learning progress so as to maximise learning experiences. Multimedia based interactive learning technology allows students to experience a subject from a diverse series of angles such that their understanding of the subject is multi-dimensional. Apart from providing various perspectives in the teaching of a subject, such systems allow the user to explore the subject as deeply as needed while the incorporation and utilisation of various resources keeps the interest level high.

Exciting though these possibilities are, the techniques for authoring multimedia systems are not yet widely known. This project addresses that problem by building upon an extensive body of practical experience gained in the component parts of these systems over a range of disciplines. Details of our experience are included as an appendix.

3.3.1. Educational Impact

The learning advantages claimed for Multimedia discussed above are expected to be achieved by Higher Education teachers who use this training package.

3.3.2. Potential For Staff Development

One of the main objectives of this project is to produce a package which will be used for developing staff skills in multimedia projects. The package can be used by teachers alone, or as groups who will eventually form learning software production teams.

3.3.3. Survey Results On The Demand For This Training Package

Emashe carried out a survey in 1993 of all teaching staff at Glasgow University. It showed that a high proportion of the teaching staff were already using computers for teaching, and wanted to learn how to use multimedia packages or produce their own .

- only 7% were "complete novices with computers"
- 77% had computers in their own workspace
- most used word processing, some used spreadsheet,databases, graphics & statistics
- 50% used computers for handouts, presentations & course administration
- 25% used computers for communication with students,
- 25% used computer based learning
- 65% interested in adapting their own material to the computer environment (or do already)

CHAPTER FOUR

RELATED WORKS

4.1. Related Works

HAM.

The HAM (Hypermedia Abstract Machine, [4]) has been used to support hypertext-based CAD and CASE applications. However, it was designed to provide sufficient generality for use with other applications. This led to a very low-level storage engine leaving the definition and implementation of all application-specific functionality and the integration of application-level design decisions into the database server to the application developer. The HAM's data model provides five different objects: A graph as the highest level object can contain the other HAM objects (contexts, nodes, links, and attributes). Contexts are used to divide the data within graphs. Nodes may contain arbitrary contents. Links define relationships between nodes. Contexts, nodes, and links may have attributes with arbitrary attribute values which can be used to describe application specific semantics to HAM objects.

HyperBase (GMD-IPSI).

GMD-IPSI's HyperBase approach [23] aimed at supporting hypertext-based authoring systems. It provides the hypertext application developer with an application interface to the hypermedia engine. While the storage of persistent objects is handled by the engine, the application developer has to define the application-dependent semantics of those objects in the application program. The data model offers nodes, links, composite objects, and attributes. Nodes, links and composites may carry application-defined attributes. Nodes may have content. Links connect two HyperBase objects, where nodes as well as links and composites are

HyperBase objects. Composites are collections of HyperBase objects. Hyperbase and its successor CHS [22] provided multi-user access and transactions.

HyperBase (Aalborg University).

Aalborg University's HyperBase [32] is a layered system providing three layers: basic entities (a simple hypertext model with nodes and links), basic services (basic operations to be performed on the basic entities, like e.g. creating nodes and links and connecting nodes with links), and multi-user services (e.g., simple user-controlled locking mechanisms to support asynchronous collaborative work). The basic entities form the data model: Links are separate objects which can only refer (unidirectionally) to nodes. Each link stores its destination node and each node keeps a list of its outgoing links. While nodes are versioned, links are not.

Hyperform.

The successor to Aalborg University's HyperBase, Hyperform [31], provides a set of built-in classes which can be used to enrich a self-defined hypermedia data model with DBMS functionality. There are three basic classes (Meta Class, System Object, and Object) and five subclasses of Object. These five classes provide basic functionality for hypermedia applications: concurrency control, notification control, access control, version control, and query and search. Using subclassing, the application developer creates the classes he needs in order to model the application's data model and can then - by means of multiple inheritance - enrich the data model with the functionality provided by the five subclasses of Object, thus adding concurrency control, etc. to the model.

HB3.

Within the System Prototype 3 (SP3) at Texas A&M University, HB3 [15] is designed to meet the storage needs of the SP3 which supports process-oriented hypermedia models like e.g. the Dexter model [12]. HB3 has two main components: The association set manager (ASM) and the versioned object manager (VOM). The system's hypermedia data model itself is defined outside of HB3 within the link

The DeVise hypermedia system prototype [9] follows the Dexter model [12] of a layered architecture. An object-oriented database server provides persistent storage for hypermedia objects while a so-called run-time process (RP) in the run-time layer provides generic and specific storage classes whose instances are stored by the underlying server. The object-oriented database server provides notification and locking mechanisms that can be used by the RP to monitor the creation, deletion, and updating of hypertext elements (nodes, links, composites).

TecPad.

In the context of the ESPRIT project TecPad the OODBMS O2 [6] was used to provide a storage module for hypertext documents [5]. This approach does not provide a hypermedia engine with a well-defined interface to hypermedia applications. It merely offers a mapping of SGML [8] onto classes in O2 and an extended O2SQL query language as a means to access or manipulate the hypertext documents stored in the database.

These hypermedia engines either provide a rather fixed hypermedia data model that can only be extended by adding attributes whose semantics are not maintained by the engine and therefore have to be handled by the hypermedia application ("HAM generation") or they only provide storage functionality which can be plugged into a hypermedia data model that has to be specified by the application developer (Hyperform, HB3, and DeVise).

In the first case, this leads to a large semantic gap between the application's data model and the hypermedia engine's data model that has to be overcome by the application developer and in the second case the application developer is bothered unnecessarily with decisions about the persistent storage of the hypermedia application he is to develop. Therefore, we now focus on the requirements of an extensible object-oriented hypermedia engine that eases the tasks of hypermedia application developers.



CHAPTER FIVE

OVERVIEW for HYPERMEDIA ENGINE

5.1. General Description of Hypermedia Engine

Hypermedia Engine was designed for preparing and also presenting hypermedia documents. It is possible with this system to prepare hypermedia documents. Your document may contain text, image, animation, sound and any combination of these.

Why do we need a Hypermedia Engine? If we want to create attractive documents which includes text, image, animation, sound we can use this solution. Some examples for this application are as follows:

- This Hypermedia Engine may be used for preparing company catalogues. For example if you want to introduce a company to customers, you can use this Hypermedia Engine. A very interesting Digital Catalogue can be easily designed. If a customer or user of this document click a hypertext word a picture occurs, an animation plays, a sound plays. You can add pictures of the company's products, movies of chairman's and different parts of company, graphics about financial situation of company or production reports of company. This document will be more attractive than a group of paper.

- Another application for Hypermedia Engine is Digital Education. An Education Program about geography may be prepared by using this system. For example a hypertext is prepared about countries. Animations about these countries, flag and map images for each country, sounds about these countries national languages can be added to hypermedia document. This kind of document is more enjoyable than traditional geography books for children to learn geography.

The Hypermedia Engine consists of two different parts:

- 1) Hypermedia Editor
- 2) Hypermedia Browser

Hypermedia Editor is used for preparing hypermedia document. You add all your links to words in this part. Hypermedia Browser is used for browsing hypermedia document. All links to a word can be reached in this part. Detailed information will be given in next section.

Hypermedia Browser is the browser part of this application. You can browse hypermedia document by using this tool. At first it shows main hypermedia document. If user clicks on a linked word then this tool shows a picture, plays a movie or sound.

5.2. Editor of Hypermedia Engine

Editor of Hypermedia Engine has some differences from a normal editor. This editor has some special buttons to add hyperlinks to a document. Editor with blank page looks like as in Figure 5.1. Form of Editor consists of 4 main elements:

- ToolBar
- CommonDialog
- ImageList (for buttons in Tool Bar)
- RichTextBox

ToolBar was used to locate buttons on tool bar. These buttons are necessary for a user friendly interface. There are 9 buttons on ToolBar:

- New Document
- Open Document
- Save Document

- Link to Text
- Link to Picture
- Link to Sound
- Link to Movie
- Hypermedia Browser
- Help

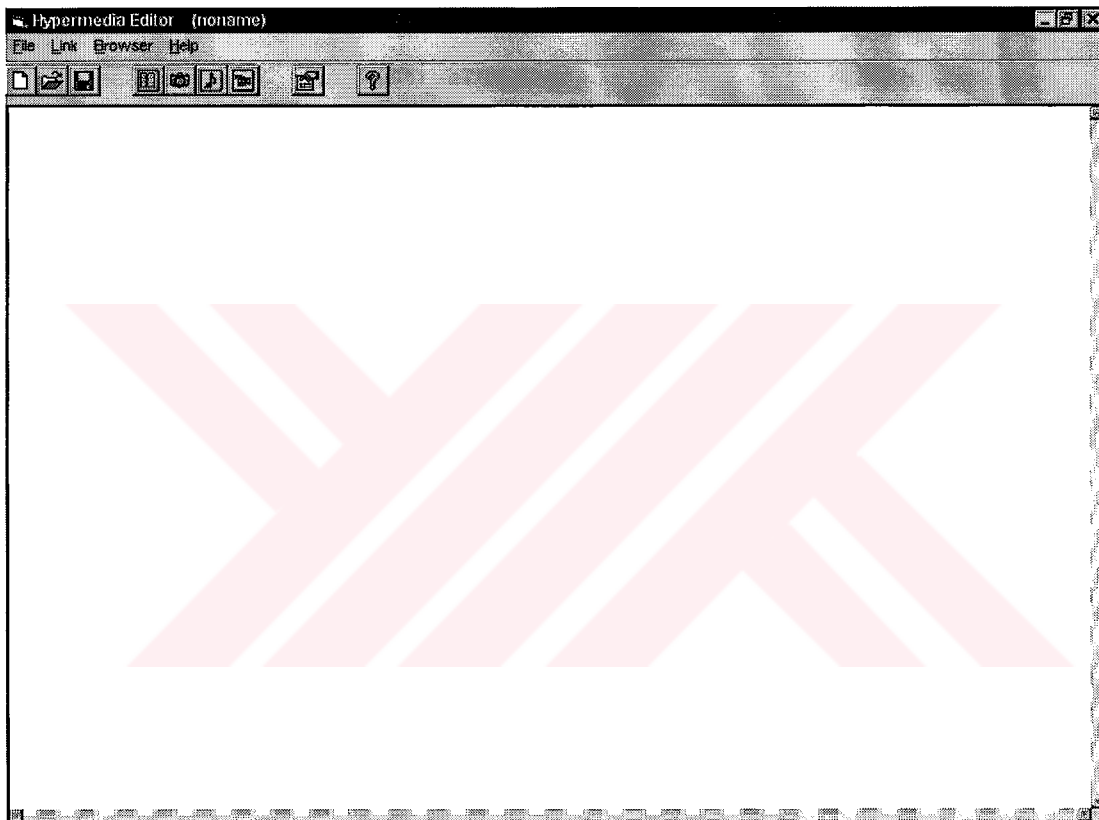


Figure 5.1 Hypermedia Editor

CommonDialog was used for Open, Save, Link operations. As known CommonDialog simlifies file choosing and similiar steps. By using this tool it is very easy to access all drives on your computer and files in this drives. So it is very useful for Open, Save and Link operations.

ImageList was used for pictures needed for buttons. In fact it is a list which consists of selected images. Any image from this list can be assigned to any button on the tool bar.

For editing environment RichTextBox was used. Since TextBox's capabilities are not enough for this application RichTextBox was preferred. In TextBox you can not change text attributes. For example it is not possible for a TextBox to use different font colors or different font sizes. You have to use same text attributes in all TextBox. As can be predicted it is not applicable for this application. Because we need different font colors and different font attributes in the same window. As an example a word which has a link to a picture has green color. But a word which has a link to a movie has purple color. And all words which contain a link must have bold fonts.

5.3. Browser of Hypermedia Engine

Browser of Hypermedia Engine is a special browser which detects all links in a hypermedia document. Browser with blank page looks like as in Figure 5.2. Form of Browser consists of 6 main elements:

- ToolBar
- CommonDialog
- ImageList
- PictureBox (One for text and one for images)
- VscrollBar
- Animation

ToolBar was used to locate buttons on tool bar. These buttons are necessary for a user friendly interface. There are 2 buttons on ToolBar:

- Open Document
- Hypermedia Editor

CommonDialog was used for Open operation. CommonDialog tool simlifies file choosing and similiar steps. As said before by using this tool it is very easy to access all drives on your computer and files in this drives. So it is very useful for Open operation.

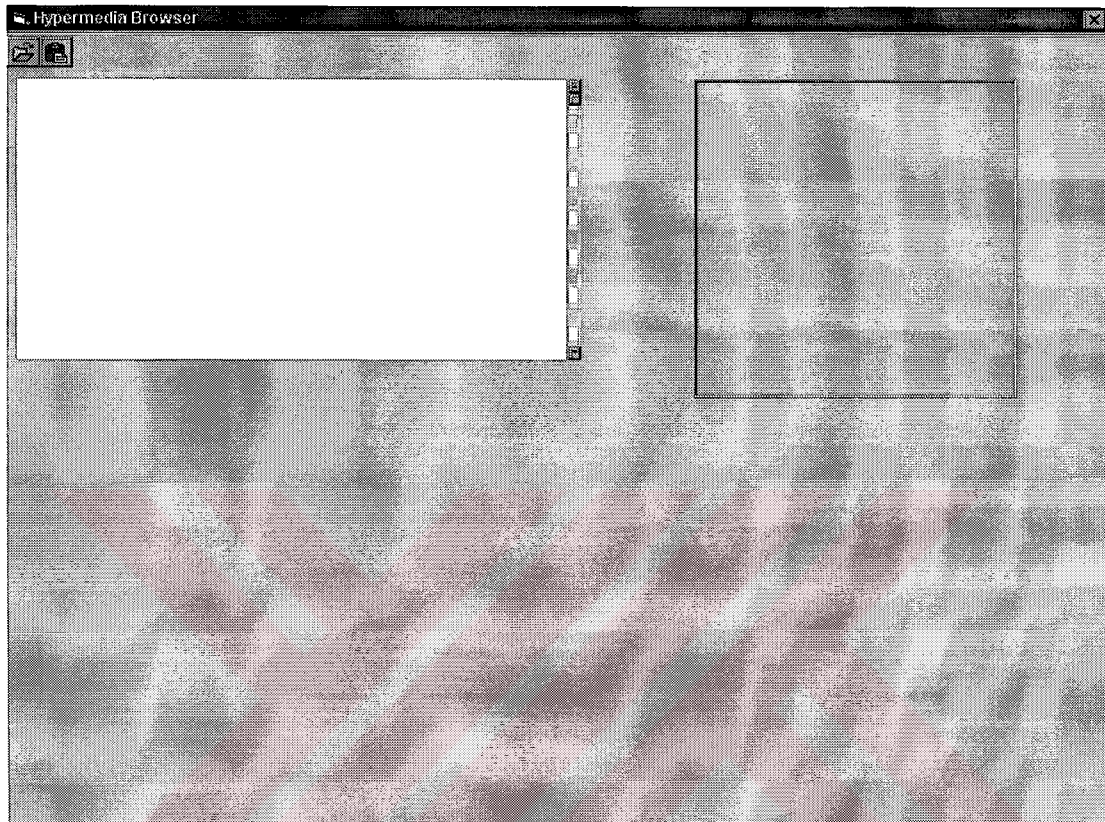


Figure 5.2 Hypermedia Browser

ImageList was used for pictures needed for buttons. In fact it is a list which consists of selected images. Any image from this list can be assigned to any button on the tool bar.

For displaying hypermedia a PictureBox was used. It was the most difficult side of the project. Because it is not possible to detect mouse position on a TextBox in Visual Basic. This property is same in RichTextBox. So TextBox and RichTextBox were not suitable for displaying hypermedia. Coordinates are very important for hypermedia. Since coordinates for every word are different we have to check

coordinates at every mouse click. A PictureBox has the coordinate property. So you can detect the coordinates of a mouse click.

Other PictureBox was used for this tool's main object: Displaying pictures. If a clicked word has a link to a picture, this picture is shown by this PictureBox.

VscrollBar was used for scrolling hypermedia document in PictureBox. If your document is longer than the size of PictureBox, VscrollBar is necessary to scroll the document.

Animation tool was used to play movies. If a clicked word has a link to a movie, this movie is shown by this Animation tool.



CHAPTER SIX

METHODS USED in HYPERMEDIA ENGINE

6.1. Creating Hypermedia

The most difficult part of this project was the displaying hypermedia. But creating hypermedia was also a difficult part.

The Hypermedia Editor saves the all hypermedia document (with the links of linked words) as a text file. The extension for hypermedia documents is “hmd”. A document with the “hmd” extension contains text side of hypermedia document and also link information for words in hypermedia document. For example our text is:

“This is a project about Hypermedia Engine”

and we want to add a link to “project”. Assume that we want to link to a picture located at “C:\Pictures\pic1.jpg” when the user clicks on the word “project”. To add this picture link to the specified word we have to click the specified word at first. After that we click the Link to Picture button in toolbar. This click brings an OpenFileDialog window for choosing the picture file we want to link. We can choose any picture file. It can be a jpg, bmp or gif file. After choosing the file, path of this file is attached to specified word “project” and the sentence seems as:

“This is a <##LP=C:\Pictures\pic1.jpg>project about Hypermedia Engine”

The specified word “project” and its link’s path color are changed. This is for a user friendly interface. Different link types has different colors. Colors for links are as follows:

- Path for Link to Text: Blue
- Path for Link to Picture: Green
- Path for Link to Sound: Red
- Path for Link to Movie: Purple

Different methods can be used for creating hypermedia document. For example link’s path may not shown . A “Link Table” may made. It has two columns. First column contains the specified words. Second columns contains the link’s path. But this method is less user friendly than my method.

6.2. Displaying Hypermedia

As i said before displaying hypermedia is the most difficult side of this project. Since i used a PictureBox tool for displaying hypermedia it was not possible to use all hypermedia text as a string. As i said before i could not use a TextBox to display hypermedia document. Because TextBox tool has no coordinate propert for mouse clicks.

The Hypermedia Browser has to detect all mouse clicks on hypermedia document. If the user clicks on a linked word, Hypermedia Browser should detect this and do the required thing.

I used a two dimension array, in fact a matrix system to store all attributes of a word in hypermedia document. I limited the size of matrix with (150,150). It means that a hypermedia document can have 150 words in a line and it can have maximum 150 lines. Each element in this matrix has 7 attributes. These attributes are:

- Leftpoint: It shows the left X coordinate of the word.

- Toppoint: It shows the top Y coordinate of the word.
- Rightpoint: It shows the right X coordinate of the word.
- Bottompoint: It shows the bottom Y coordinate of the word.
- Linktype: It shows the type of link the word has. If the word has no link this attribute is empty.
- Thisword: It shows the word.
- Destinationsubject: It shows the path of link the word has. If the word has no link this attribute is empty.

Hypermedia Browser works as follows:

When the user open a hypermedia document, Hypermedia Browser shapes this document. Each word in hypertext document is read with the following command: `Left$(anystring, InStr(anystring, " "))`. If the word starts with the string "<##", it means that this is a link path. The first two letters after this string show the type of link. If these two letters are "LT" it means this word has link to a text. LP shows this word has a link to picture. LS shows this word has a link to a sound. LM shows this word has a link to a movie. Program writes this type to Linktype section of array for this word.

If the word has a link, the path of this link is obtained by using the following command : `Mid$(getword, 7, InStr(getword, ">") - 7)`. This path is written to Destinationsubject part of array.

After these steps, program finds the coordinates of this word. The Leftpoint of first word is 0 and the Toppoint of first word is also 0. Rightpoint of a word is found by using following commands:

```
Picture1.Print getword
```

```
Picture1.CurrentX = X1 + Picture1.TextWidth(getword)
```

```
hyperlinkarray(a, b).rightpoint = Picture1.CurrentX
```

“Picture1.Print getword” command prints the word to PictureBox. After a word printed to PictureBox, CurrentX property of the PictureBox changes. Old CurrentX value shows the Leftpoint of the word and new CurrentX value shows the Rightpoint of the word.

Bottom values are easily calculated with CurrentY value of PictureBox. The Newline command sends the cursor to a new line and the CurrentY value changes. The new CurrentY value defines the Toppoint value for all words in the new line. This new value also defines the Bottompoint value for all words in the previous line.

After all these steps hypermedia document is displayed completed in the PictureBox. But it doesn't mean that there is no problem anymore. There is another big problem. Assume that your hypermedia document is longer than your PictureBox. So we need a ScrollBar to scroll hypermedia document in the PictureBox. But this is not as easy as to add only a ScrollBar. Because all array system that we constructed should be change when we press the ScrollBar's up or down button. Remember that Leftpoint, Toppoint, Rightpoint, Bottompoint values determine the coordinates of a word in PictureBox. If we scroll this document by using ScrollBar we have to change Leftpoint, Toppoint, Rightpoint, Bottompoint values dynamically. This means that our array structure is a dynamic array structure instead of a static one.

I used a method to create a dynamic array structure. In this method i count the number of clickings on ScrollBar. If the user press once the down button of ScroolBar the first line goes out. And the second line becomes the first line. Since the height of a line is 16 unit, new Bottompoint and Toppoint values of the all lines can be easily calculated. These new values are written the array structure as valid values.

6.3. Creating and Displaying Link to Text

This part was not completed in this project. But the method is clear to create and display Link to Text. To create Link to Text, related button in the Hypermedia Editor

should be used. This button does required tasks and at the end specified word has a link. Hypermedia Editor accepts files only with “txt” extension.

To display Link to Text a TextBox should be used. Assume that we want to create a geography CD for children. The PictureBox of Hypermedia shows the country names. When the user clicks on the name of a country detailed information about this country may shown in TextBox of Link to Text

6.4. Creating and Displaying Link to Picture

To create Link to Picture, related button used in the Hypermedia Editor. Hypermedia Editor accepts all files with “gif, jpg, bmp, ico” extensions.

To display Link to Picture a PictureBox was used as said before. This PictureBox was set to dynamic size. It means that if the picture's size is different than the PictureBox's size the PictureBox's size is set to picture's size. If we think the same example, when the user clicks on the name of a country the map of this country may shown in PictureBox.

6.5. Creating and Displaying Link to Sound

This part was not completed in this project. But the method is clear to create and display Link to Sound. To create Link to Sound, related button in the Hypermedia Editor should be used. This button does required tasks and at the end specified word has a link. Hypermedia Editor accepts files only with “wav” extension.

To play Link to Sound a sound tool should be used in Visual Basic. For the same example when the user clicks on the name of a country the name of country may pronounced or some simple sentences may said in the language of this country.

6.6. Creating and Displaying Link to Movie

Adding Link to Movie is similiar to other adds. Related button is Link to Movie button in Hypermedia Editor. Hypermedia Editor accepts the files only with “avi” extension.

To play Link to Movie an Animation tool was used. This tool has a lot of propert. So it is possible to play the same movie many times or do something else. This ype of link also may used in the same application. Some historical places can shown in this picture about the selected country.



CHAPTER SEVEN

CODE for HYPERMEDIA ENGINE

7.1 Code for Hypermedia Editor

```
Dim hyperlinkarray(150, 150) As hyperlinkelement
```

```
Private Sub Form_Load()
```

```
Form1.Caption = Form1.Caption + " (" & "noname" & ")"
```

```
End Sub
```

```
Private Sub Form_Resize()
```

```
RichTextBox1.Width = Form1.ScaleWidth
```

```
RichTextBox1.Height = Form1.ScaleHeight - RichTextBox1.Top
```

```
End Sub
```

```
Private Sub menuClose_Click()
```

```
'MsgBox "Document will be closed"
```

```
End Sub
```

```
Private Sub menuExit_Click()
```

```
End
```

```
End Sub
```

```
Private Sub menuHelp2_Click()
```

```
MsgBox "HELP"
```

```
End Sub
```

```
Private Sub menuMovie_Click()
```



```

RichTextBox1.SelBold = True
RichTextBox1.SelColor = RGB(256, 0, 256)
Dim sFile As String
With dlgCommonDialog
    .Filter = "All Movie Files (*.avi;*.mpg)|*.avi;*.mpg"
    .ShowOpen
    If Len(.filename) = 0 Then
        Exit Sub
    End If
    MsgBox "Seçilen Dosya" & .filename
    RichTextBox1.SelText = "<##LM=" & .filename & ">" &
RichTextBox1.SelText
    RichTextBox1.SelLength = 0
    RichTextBox1.SelBold = False
    RichTextBox1.SelColor = RGB(0, 0, 0)
    RichTextBox1.SetFocus
End With
End Sub

Private Sub menuNew_Click()
    MsgBox "New document will be opened"
End Sub

Private Sub MenuOpen_Click()
    MsgBox "Available document will be opened"
    Dim sFile As String
    With dlgCommonDialog
        .Flags = 4
        .Filter = "All Hypermedia Documents (*.hmd)|*.hmd"
        .ShowOpen
        If Len(.filename) = 0 Then
            Exit Sub
        End If
    End With

```

```

    End If
End With
filename = dlgCommonDialog.filename
RichTextBox1.LoadFile dlgCommonDialog.filename
Form1.Caption = "Hypermedia Editor" & " (" & filename & ")"
End Sub

Private Sub menuPicture_Click()
    RichTextBox1.SelBold = True
    RichTextBox1.SelColor = RGB(0, 255, 0)
    Dim sFile As String
    With dlgCommonDialog
        .Filter = "All Picture Files (*.gif;*.jpg;*.bmp;*.ico)|*.gif;*.jpg;*.bmp;*.ico"
        .ShowOpen
        If Len(.filename) = 0 Then
            Exit Sub
        End If
        MsgBox "Seçilen Dosya" & .filename
        RichTextBox1.SelText = "<##LP=" & .filename & ">" &
RichTextBox1.SelText
        RichTextBox1.SelLength = 0
        RichTextBox1.SelBold = False
        RichTextBox1.SelColor = RGB(0, 0, 0)
        RichTextBox1.SetFocus
    End With
End Sub

Private Sub menuSave_Click()
    MsgBox "Döküman Kaydedilecek"
    captiontext = Form1.Caption
    If InStr(captiontext, ".") > 0 Then
        savenumber = 1
    End If
End Sub

```

```

Filename2 = Left$(Form1.Caption, Len(Form1.Caption) - 1)
filename = Mid$(Filename2, 22)
RichTextBox1.SaveFile filename, 1
End If
If savenumber <> 1 Then
    Dim sFile As String
    With dlgCommonDialog
        .Flags = &H4 Or &H2
        .Filter = "Hypermedia Documents (*.hmd)|*.hmd"
        .ShowSave
        If Len(.filename) = 0 Then
            Exit Sub
        End If
    End With
    Form1.Caption = "Hypermedia Editor" + " (" & dlgCommonDialog.filename &
    ")
    RichTextBox1.SaveFile dlgCommonDialog.filename, 1
    filename = dlgCommonDialog.filename
    savenumber = 1
    RichTextBox1.SaveFile filename, rtfText
End If
End Sub

```

```

Private Sub menuSound_Click()
    RichTextBox1.SelBold = True
    RichTextBox1.SelColor = RGB(255, 0, 0)
    Dim sFile As String
    With dlgCommonDialog
        .Filter = "All Sound Files (*.wav)|*.wav"
        .ShowOpen
        If Len(.filename) = 0 Then
            Exit Sub
        End If
    End With

```

```

End If
MsgBox "Seçilen Dosya" & .filename
RichTextBox1.SelText = "<##LS=" & .filename & ">" &
RichTextBox1.SelText
RichTextBox1.SelLength = 0
RichTextBox1.SelBold = False
RichTextBox1.SelColor = RGB(0, 0, 0)
RichTextBox1.SetFocus
End With
End Sub

```

```

Private Sub menuText_Click()
RichTextBox1.SelBold = True
RichTextBox1.SelColor = RGB(0, 0, 255)
MsgBox "Text'e Link Olacak"
Dim sFile As String
With dlgCommonDialog
.Filter = "All Text Files (*.txt)|*.txt"
.ShowOpen
If Len(.filename) = 0 Then
Exit Sub
End If
MsgBox "Seçilen Dosya" & .filename
RichTextBox1.SelText = "<##LT=" & .filename & ">" &
RichTextBox1.SelText
RichTextBox1.SelLength = 0
RichTextBox1.SelBold = False
X = Form1.CurrentX
Y = Form1.CurrentY
MsgBox X, Y
RichTextBox1.SelColor = RGB(0, 0, 0)
RichTextBox1.SetFocus

```

```
End With
End Sub

Private Sub Toolbar1_ButtonClick(ByVal Button As ComctlLib.Button)
Select Case Button.Key
    Case "New"
        menuNew_Click
    Case "Open"
        MenuOpen_Click
    Case "Save"
        menuSave_Click
    Case "Link to Text"
        menuText_Click
    Case "Link to Picture"
        menuPicture_Click
    Case "Link to Sound"
        menuSound_Click
    Case "Link to Movie"
        menuMovie_Click
    Case "Help"
        menuHelp2_Click
        Case "Browser"
            Form3.Show
    End Select
End Sub
```

7.2 Code for Hypermedia Browser

```
Dim hyperlinkarray(150, 150) As hyperlinkelement
Dim changescroll, numberofchange As Integer
Dim tekrar As Integer
Dim a, b, c, d As Integer
Dim TextLine As String
```

```
Sub newline()
    Picture1.Print
    currenttop = Picture1.CurrentY
End Sub
```

```
Sub loadsubject(anystring As String)
    Dim getword As String
    Dim destination As String
    Dim creatingalink As Integer
    Dim thisword As String
    Picture2.Cls
    a = 1
    b = 1
    oldline = ""
    Do While anystring <> ""
        getword = Left$(anystring, InStr(anystring, " "))
        destination = ""
        If InStr(anystring, " ") = 0 Then
            getword = anystring
            stringend = True
        End If
        If stringend = True Then
            anystring = " "
        Else
```

```

    anystring = Mid$(anystring, InStr(anystring, " ") + 1)
End If
If Left$(getword, 3) = "<##" Then
    creatingalink = True
    'Linked word follows the sign <##. So this kind of words should be
seperated
    If Mid$(getword, 5, 1) = "T" Then
        Picture1.ForeColor = RGB(0, 0, 255)
        Picture1.FontBold = True
        destination = Mid$(getword, 7, InStr(getword, ">") - 7)
        getword = Mid$(getword, InStr(getword, ">") + 1)
        If ((Picture1.CurrentX + Picture1.TextWidth(getword)) >
Picture1.ScaleWidth) Then
            For k = 1 To 25
                hyperlinkarray(k, b).toppoint = hyperlinkarray(1, b).toppoint
            Next k
            newline
            a = 1
            b = b + 1
        Else
            a = a + 1
        End If
        hyperlinkarray(a, b).linktype = "T"
        hyperlinkarray(a, b).destinationsubject = destination
    End If
    If Mid$(getword, 5, 1) = "P" Then
        Picture1.ForeColor = RGB(0, 255, 0)
        Picture1.FontBold = True
        destination = Mid$(getword, 7, InStr(getword, ">") - 7)
        getword = Mid$(getword, InStr(getword, ">") + 1)
        If ((Picture1.CurrentX + Picture1.TextWidth(getword)) >
Picture1.ScaleWidth) Then

```

```

    For k = a To 25
        hyperlinkarray(k, b).toppoint = hyperlinkarray(1, b).toppoint
    Next k
    newline
    a = 1
    b = b + 1
Else
    a = a + 1
End If
hyperlinkarray(a, b).linktype = "P"
hyperlinkarray(a, b).destinationsubject = destination
End If
If Mid$(getword, 5, 1) = "S" Then
    Picture1.ForeColor = RGB(255, 0, 0)
    Picture1.FontBold = True
    destination = Mid$(getword, 7, InStr(getword, ">") - 7)
    getword = Mid$(getword, InStr(getword, ">") + 1)
    If ((Picture1.CurrentX + Picture1.TextWidth(getword)) >
Picture1.ScaleWidth) Then
        For k = a To 25
            hyperlinkarray(k, b).toppoint = hyperlinkarray(1, b).toppoint
        Next k
        newline
        a = 1
        b = b + 1
    Else
        a = a + 1
    End If
    hyperlinkarray(a, b).linktype = "S"
    hyperlinkarray(a, b).destinationsubject = destination
End If
If Mid$(getword, 5, 1) = "M" Then

```



```

Picture1.ForeColor = RGB(256, 0, 256)
Picture1.FontBold = True
destination = Mid$(getword, 7, InStr(getword, ">") - 7)
getword = Mid$(getword, InStr(getword, ">") + 1)
If ((Picture1.CurrentX + Picture1.TextWidth(getword)) >
Picture1.ScaleWidth) Then
    For k = a To 25
        hyperlinkarray(k, b).toppoint = hyperlinkarray(1, b).toppoint
    Next k
    newline
    a = 1
    b = b + 1
Else
    a = a + 1
End If
hyperlinkarray(a, b).linktype = "M"
hyperlinkarray(a, b).destinationsubject = destination
End If
X1 = Picture1.CurrentX
Y1 = Picture1.CurrentY
hyperlinkarray(a, b).leftpoint = X1
hyperlinkarray(a, b).toppoint = Y1
Picture1.Print getword
Picture1.CurrentX = X1 + Picture1.TextWidth(getword)
Picture1.CurrentY = Y1
hyperlinkarray(a, b).rightpoint = Picture1.CurrentX
hyperlinkarray(a, b).thisword = getword
Picture1.ForeColor = RGB(0, 0, 0)
Picture1.FontBold = False
Else
    creatingalink = False

```

```

    If ((Picture1.CurrentX + Picture1.TextWidth(getword)) >
Picture1.ScaleWidth) Then
        For k = a To 25
            hyperlinkarray(k, b).toppoint = hyperlinkarray(1, b).toppoint
        Next k
        newline
        a = 1
        b = b + 1
    Else
        a = a + 1
    End If
    X1 = Picture1.CurrentX
    Y1 = Picture1.CurrentY
    hyperlinkarray(a, b).leftpoint = X1
    hyperlinkarray(a, b).toppoint = Y1
    Picture1.Print getword
    Picture1.CurrentX = X1 + Picture1.TextWidth(getword)
    Picture1.CurrentY = Y1
    hyperlinkarray(a, b).rightpoint = Picture1.CurrentX
    hyperlinkarray(a, b).thisword = getword
    Picture1.ForeColor = RGB(0, 0, 0)
    Picture1.FontBold = False
End If
Picture2.Cls
If stringend = True Then
    getword = anystring
    anystring = ""
    If Left$(getword, 3) = "<##" Then
        creatingalink = True
        'Linked word follows the sign <##. So this kind of words should be
seperated
        If Mid$(getword, 5, 1) = "T" Then

```

```

Picture1.ForeColor = RGB(0, 0, 255)
Picture1.FontBold = True
destination = Mid$(getword, 7, InStr(getword, ">") - 7)
getword = Mid$(getword, InStr(getword, ">") + 1)
If ((Picture1.CurrentX + Picture1.TextWidth(getword)) >
Picture1.ScaleWidth) Then
    For k = a To 25
        hyperlinkarray(k, b).toppoint = hyperlinkarray(1, b).toppoint
    Next k
    newline
    a = 1
    b = b + 1
Else
    a = a + 1
End If
hyperlinkarray(a, b).linktype = "T"
hyperlinkarray(a, b).destinationsubject = destination
End If
If Mid$(getword, 5, 1) = "P" Then
    Picture1.ForeColor = RGB(0, 255, 0)
    Picture1.FontBold = True
    destination = Mid$(getword, 7, InStr(getword, ">") - 7)
    getword = Mid$(getword, InStr(getword, ">") + 1)
    If ((Picture1.CurrentX + Picture1.TextWidth(getword)) >
Picture1.ScaleWidth) Then
        For k = a To 25
            hyperlinkarray(k, b).toppoint = hyperlinkarray(1, b).toppoint
        Next k
        newline
        a = 1
        b = b + 1
    Else

```

```

        a = a + 1
    End If
    hyperlinkarray(a, b).linktype = "P"
    hyperlinkarray(a, b).destinationsubject = destination
End If
If Mid$(getword, 5, 1) = "S" Then
    Picture1.ForeColor = RGB(255, 0, 0)
    Picture1.FontBold = True
    destination = Mid$(getword, 7, InStr(getword, ">") - 7)
    getword = Mid$(getword, InStr(getword, ">") + 1)
    If ((Picture1.CurrentX + Picture1.TextWidth(getword)) >
Picture1.ScaleWidth) Then
        For k = a To 25
            hyperlinkarray(k, b).toppoint = hyperlinkarray(1, b).toppoint
        Next k
        newline
        a = 1
        b = b + 1
    Else
        a = a + 1
    End If
    hyperlinkarray(a, b).linktype = "S"
    hyperlinkarray(a, b).destinationsubject = destination
End If
If Mid$(getword, 5, 1) = "M" Then
    Picture1.ForeColor = RGB(256, 0, 256)
    Picture1.FontBold = True
    destination = Mid$(getword, 7, InStr(getword, ">") - 7)
    getword = Mid$(getword, InStr(getword, ">") + 1)
    If ((Picture1.CurrentX + Picture1.TextWidth(getword)) >
Picture1.ScaleWidth) Then
        newline

```

```

    For k = a To 25
        hyperlinkarray(k, b).toppoint = hyperlinkarray(1, b).toppoint
    Next k
    a = 1
    b = b + 1
Else
    a = a + 1
End If
hyperlinkarray(a, b).linktype = "M"
hyperlinkarray(a, b).destinationsubject = destination
End If
X1 = Picture1.CurrentX
Y1 = Picture1.CurrentY
hyperlinkarray(a, b).leftpoint = X1
hyperlinkarray(a, b).toppoint = Y1
Picture1.Print getword
Picture1.CurrentX = X1 + Picture1.TextWidth(getword)
Picture1.CurrentY = Y1
hyperlinkarray(a, b).rightpoint = Picture1.CurrentX
hyperlinkarray(a, b).thisword = getword
Picture1.ForeColor = RGB(0, 0, 0)
Picture1.FontBold = False
Else
    If ((Picture1.CurrentX + Picture1.TextWidth(getword)) >
Picture1.ScaleWidth) Then
        For k = a To 25
            hyperlinkarray(k, b).toppoint = hyperlinkarray(1, b).toppoint
        Next k
        newline
        a = 1
        b = b + 1
    Else

```

```

        a = a + 1
    End If
    X1 = Picture1.CurrentX
    Y1 = Picture1.CurrentY
    hyperlinkarray(a, b).leftpoint = X1
    hyperlinkarray(a, b).toppoint = Y1
    Picture1.Print getword
    Picture1.CurrentX = X1 + Picture1.TextWidth(getword)
    Picture1.CurrentY = Y1
    hyperlinkarray(a, b).rightpoint = Picture1.CurrentX
    hyperlinkarray(a, b).thisword = getword
    Picture1.ForeColor = RGB(0, 0, 0)
    Picture1.FontBold = False
    For k = 2 To 25
        hyperlinkarray(k, b).toppoint = hyperlinkarray(1, b).toppoint
    Next k
End If
newline
a = 1
b = b + 1
X1 = Picture1.CurrentX
Y1 = Picture1.CurrentY
hyperlinkarray(a, b).leftpoint = X1
hyperlinkarray(a, b).toppoint = Y1
For k = 2 To 25
    hyperlinkarray(k, b).toppoint = hyperlinkarray(1, b).toppoint
Next k
For Y = 1 To 25
    For X = 1 To 25
        hyperlinkarray(X, Y).bottompoint = hyperlinkarray(X, Y + 1).toppoint
    Next X
Next Y

```

End If

Loop

aaa = hyperlinkarray(1, 1).toppoint - hyperlinkarray(1, 1).bottompoint

bbb = hyperlinkarray(1, 1).toppoint - hyperlinkarray(1, 2).toppoint

changescroll = 0

numberofchange = 0

tekrar = 0

End Sub

Sub Command1_Click()

With dlgCommonDialog

.Filter = "avi (*.avi)|*.avi"

.ShowOpen

End With

With Animation1

.AutoPlay = True

.Open dlgCommonDialog.filename

End With

End Sub

Private Sub Command2_Click()

For k = 1 To 150

For j = 1 To 150

hyperlinkarray(k, j).leftpoint = 0

hyperlinkarray(k, j).rightpoint = 0

hyperlinkarray(k, j).toppoint = 0

hyperlinkarray(k, j).bottompoint = 0

hyperlinkarray(k, j).thisword = ""

hyperlinkarray(k, j).linktype = ""

hyperlinkarray(k, j).destinationsubject = ""

Next j

Next k

```

Picture1.Top = 0
Picture1.CurrentX = 0
Picture1.CurrentY = 0
Picture1.FontSize = 10
linkarraypos = 0
oldline = ""
loadsubject (anystring)
Text1.Text = hyperlinkarray(2, 1).leftpoint & " " & hyperlinkarray(2,
1).rightpoint & " " & hyperlinkarray(2, 1).thisword
End Sub

```

```

Private Sub HScroll1_Change()
P3.Left = -HScroll1.Value
End Sub

```

```

Private Sub Form_Load()
Picture2.Cls
End Sub

```

```

Private Sub Form_Resize()
Frame1.Left = Form3.Left + 150
Frame1.Top = Form3.Top + 550
Frame1.Width = Form3.ScaleWidth - 6000
Frame1.Height = Form3.Height - 6000
Picture1.Left = Form3.Left + 150
Picture1.Top = Form3.Top + 550
Picture1.Width = Form3.ScaleWidth - 6000
Picture1.Height = Form3.Height - 6000
Picture2.Left = 7500
Picture2.Top = Picture1.Top
Picture2.Width = 3500
Picture2.Height = 3500

```



```

VScroll1.Left = Frame1.Left + Frame1.Width + 30
VScroll1.Top = Picture1.Top
VScroll1.Height = Picture1.Height
End Sub

```

```

Private Sub Picture1_MouseUp(Button As Integer, Shift As Integer, X As Single, Y
As Single)
Picture2.Cls
linkcase = False
bulunan = 0
For b = 1 To 25
    For a = 1 To 25
        If hyperlinkarray(a, b).linktype = "" Then linkcase = False Else linkcase = True
        If linkcase = False Then GoTo nalink
        If ((hyperlinkarray(a, b).leftpoint < X) And (X < hyperlinkarray(a,
b).rightpoint)) And ((hyperlinkarray(a, b).toppoint < Y) And (Y < hyperlinkarray(a,
b).bottompoint)) Then
            bulunan = bulunan + 1
            linktype = hyperlinkarray(a, b).linktype
            destinationsubject = hyperlinkarray(a, b).destinationsubject
            Picture2.Print bulunan
        Else
            End If
        nalink: Next a
    Next b
    cer = 1
    If linktype = "P" Then
        Picture2.Picture = LoadPicture(destinationsubject)
    End If
    If linktype = "M" Then
        With Animation1
            .AutoPlay = True

```

```

        .Open destinationsubject
    End With
End If
End Sub

Private Sub MenuOpen_Click()
    For k = 1 To 150
        For j = 1 To 150
            hyperlinkarray(k, j).leftpoint = 0
            hyperlinkarray(k, j).rightpoint = 0
            hyperlinkarray(k, j).toppoint = 0
            hyperlinkarray(k, j).bottompoint = 0
            hyperlinkarray(k, j).thisword = ""
            hyperlinkarray(k, j).linktype = ""
            hyperlinkarray(k, j).destinationsubject = ""
        Next j
    Next k
    Picture1.Top = 0
    Picture1.CurrentX = 0
    Picture1.CurrentY = 0
    Picture1.FontSize = 10

    Dim sFile As String
    With dlgCommonDialog
        .Flags = 4
        .Filter = "All Hypermedia Documents (*.hmd)|*.hmd"
        .ShowOpen
        If Len(.filename) = 0 Then
            Exit Sub
        End If
    End With
    filename = dlgCommonDialog.filename

```

```

Open filename For Input As #1 ' Open file.
FileText = ""
Do While Not EOF(1) ' Loop until end of file.
    Line Input #1, TextLine ' Read line into variable.
    FileText = FileText + TextLine
Loop
Close #1 ' Close file.
loadsubject (FileText)
Form1.Caption = "Hypermedia Editor" & " (" & filename & ")"
End Sub

Private Sub Toolbar1_ButtonClick(ByVal Button As ComctlLib.Button)
Select Case Button.Key
    Case "Open"
        MenuOpen_Click
    Case "Editor"
        Form1.Show
End Select
End Sub

Private Sub VScroll1_Change()
Picture1.Cls
If VScroll1.Value > changescroll Then
    numberofchange = numberofchange + 1
    For b = 1 To 25
        For a = 1 To 25
            hyperlinkarray(a, b).toppoint = hyperlinkarray(a, b).toppoint - 16
            hyperlinkarray(a, b).bottompoint = hyperlinkarray(a, b).bottompoint - 16
        Next a
    Next b
    For b = (numberofchange + 1) To 25
        For a = 1 To 25

```

```

Picture1.CurrentX = hyperlinkarray(a, b).leftpoint
Picture1.CurrentY = hyperlinkarray(a, b).toppoint
If hyperlinkarray(a, b).linktype = "T" Then
    Picture1.ForeColor = RGB(0, 0, 255)
    Picture1.FontBold = True
    Picture1.Print hyperlinkarray(a, b).thisword
ElseIf hyperlinkarray(a, b).linktype = "P" Then
    Picture1.ForeColor = RGB(0, 255, 0)
    Picture1.FontBold = True
    Picture1.Print hyperlinkarray(a, b).thisword
ElseIf hyperlinkarray(a, b).linktype = "S" Then
    Picture1.ForeColor = RGB(255, 0, 0)
    Picture1.FontBold = True
    Picture1.Print hyperlinkarray(a, b).thisword
ElseIf hyperlinkarray(a, b).linktype = "M" Then
    Picture1.ForeColor = RGB(256, 0, 256)
    Picture1.FontBold = True
    Picture1.Print hyperlinkarray(a, b).thisword
Else
    Picture1.ForeColor = RGB(0, 0, 0)
    Picture1.FontBold = False
    Picture1.Print hyperlinkarray(a, b).thisword
End If
Picture1.ForeColor = RGB(0, 0, 0)
Picture1.FontBold = False
Next a
Next b
End If
If VScroll1.Value < changescroll Then
    numberofchange = numberofchange - 1
    For b = 1 To 25
        For a = 1 To 25

```

```

    hyperlinkarray(a, b).toppoint = hyperlinkarray(a, b).toppoint + 16
    hyperlinkarray(a, b).bottompoint = hyperlinkarray(a, b).bottompoint + 16
Next a
Next b
For b = numberofchange To 25
    For a = 1 To 25
        Picture1.CurrentX = hyperlinkarray(a, b).leftpoint
        Picture1.CurrentY = hyperlinkarray(a, b).toppoint
        If hyperlinkarray(a, b).linktype = "T" Then
            Picture1.ForeColor = RGB(0, 0, 255)
            Picture1.FontBold = True
            Picture1.Print hyperlinkarray(a, b).thisword
        ElseIf hyperlinkarray(a, b).linktype = "P" Then
            Picture1.ForeColor = RGB(0, 255, 0)
            Picture1.FontBold = True
            Picture1.Print hyperlinkarray(a, b).thisword
        ElseIf hyperlinkarray(a, b).linktype = "S" Then
            Picture1.ForeColor = RGB(255, 0, 0)
            Picture1.FontBold = True
            Picture1.Print hyperlinkarray(a, b).thisword
        ElseIf hyperlinkarray(a, b).linktype = "M" Then
            Picture1.ForeColor = RGB(256, 0, 256)
            Picture1.FontBold = True
            Picture1.Print hyperlinkarray(a, b).thisword
        Else
            Picture1.ForeColor = RGB(0, 0, 0)
            Picture1.FontBold = False
            Picture1.Print hyperlinkarray(a, b).thisword
        End If
        Picture1.ForeColor = RGB(0, 0, 0)
        Picture1.FontBold = False
    Next a
Next b

```

```
Next b  
End If  
changescroll = VScroll1.Value  
End Sub
```



CHAPTER EIGHT

USER GUIDE for HYPERMEDIA ENGINE

8.1 Using Hypermedia Editor

The Hypermedia Editor of the Hypermedia Engine is a very user friendly and easy to use editor. It has no difference from a normal editor. User edits his document normally at first. After this step hyperlink buttons are used for adding hyperlinks to this document.

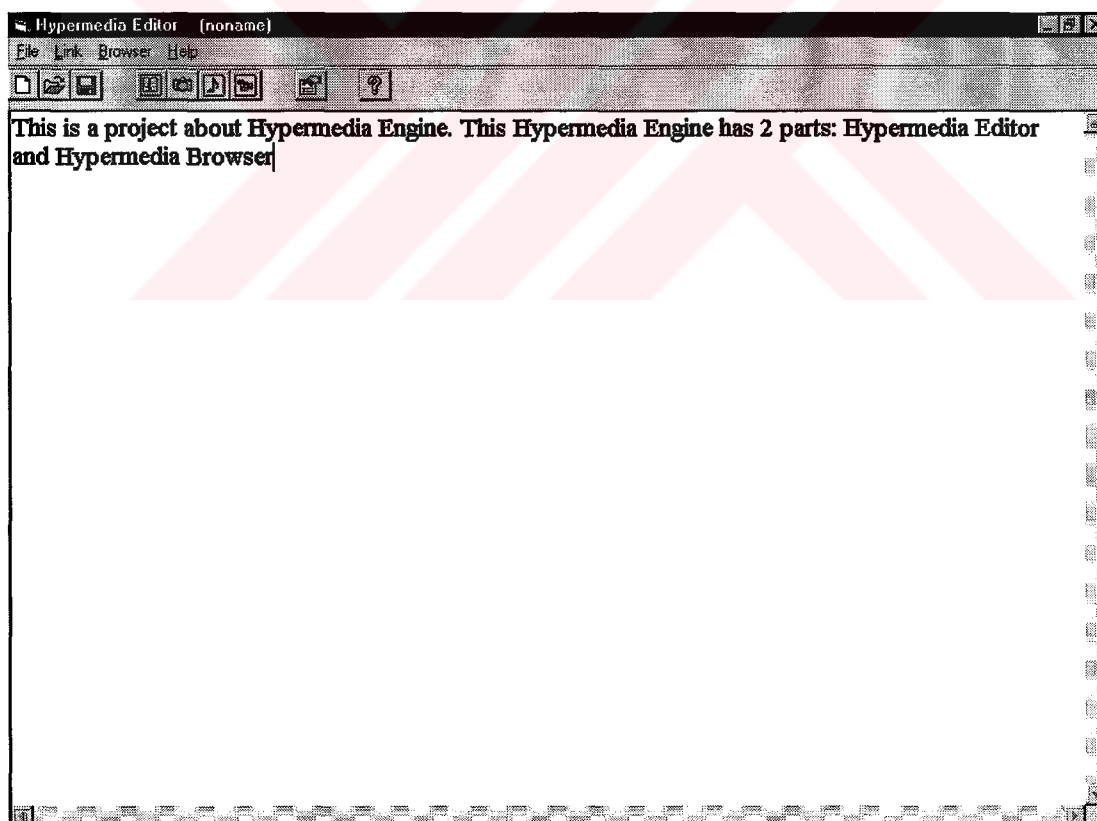


Figure 8.1 Hypermedia Editor

Assume that we want to add links to that sentence: "This is a project about Hypermedia Engine. This Hypermedia Engine has 2 parts: Hypermedia Editor and Hypermedia Browser". The Hypermedia Editor window look like in Figure 8.1.

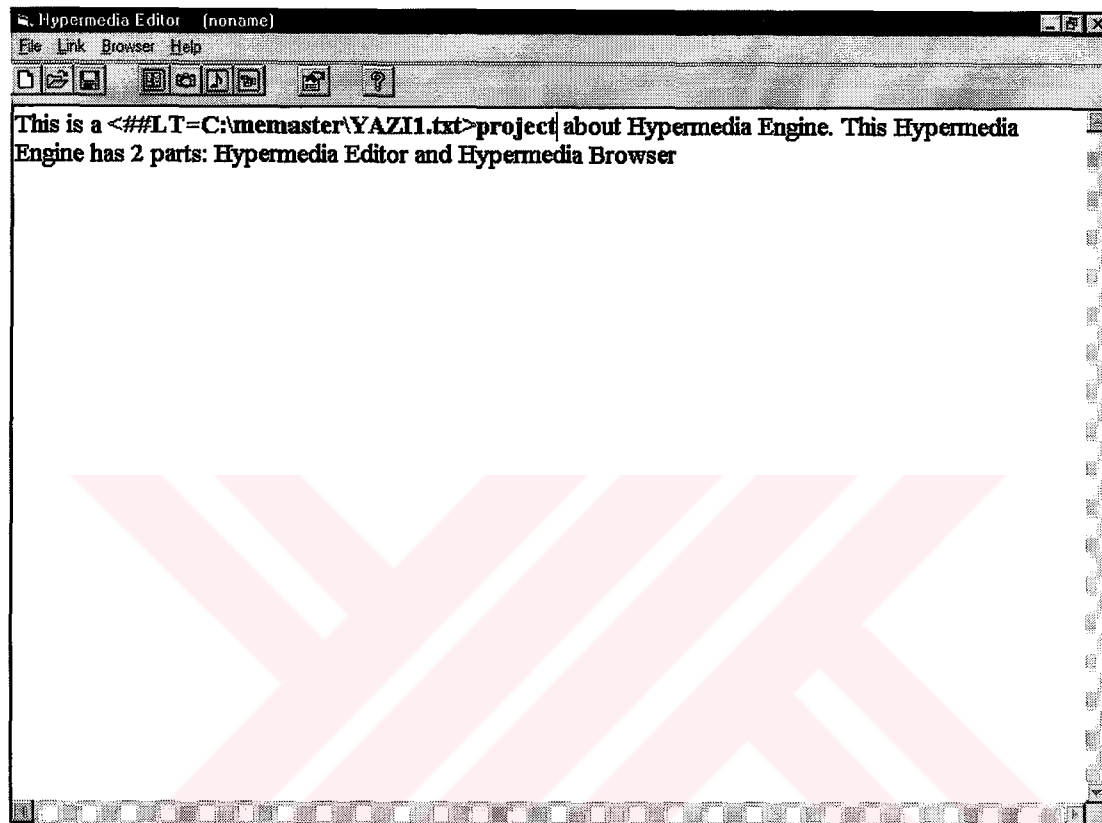


Figure 8.2 Adding Link to Text

Four types of hyperlinks can be added to a document. These are: Text, picture, sound and movie. For adding a hyperlink to a word in the document, user chooses this word by double clicking on it. After that user clicks a button on the toolbar of Hypermedia Editor according to type of the desired link. If user wants to link this word to a text he should choose the Link to Text button.

The specified word and its link's path color are chaged. This is for a user friendly interface. Different link types has different colors. Colors for links are as follows:

- Path for Link to Text: Blue
- Path for Link to Picture: Green
- Path for Link to Sound: Red
- Path for Link to Movie: Purple

After clicking Link to Text (as shown in Figure 8.2) or Link to Picture (as shown in Figure 8.3) or Link to Sound (as shown in Figure 8.3) or Link to Movie (as shown in Figure 8.4) button an opening window occurs on the screen. It helps user to choose the file to be linked. But there limitations at this step. The Hypermedia Editor only accepts the following type of files as a link file.

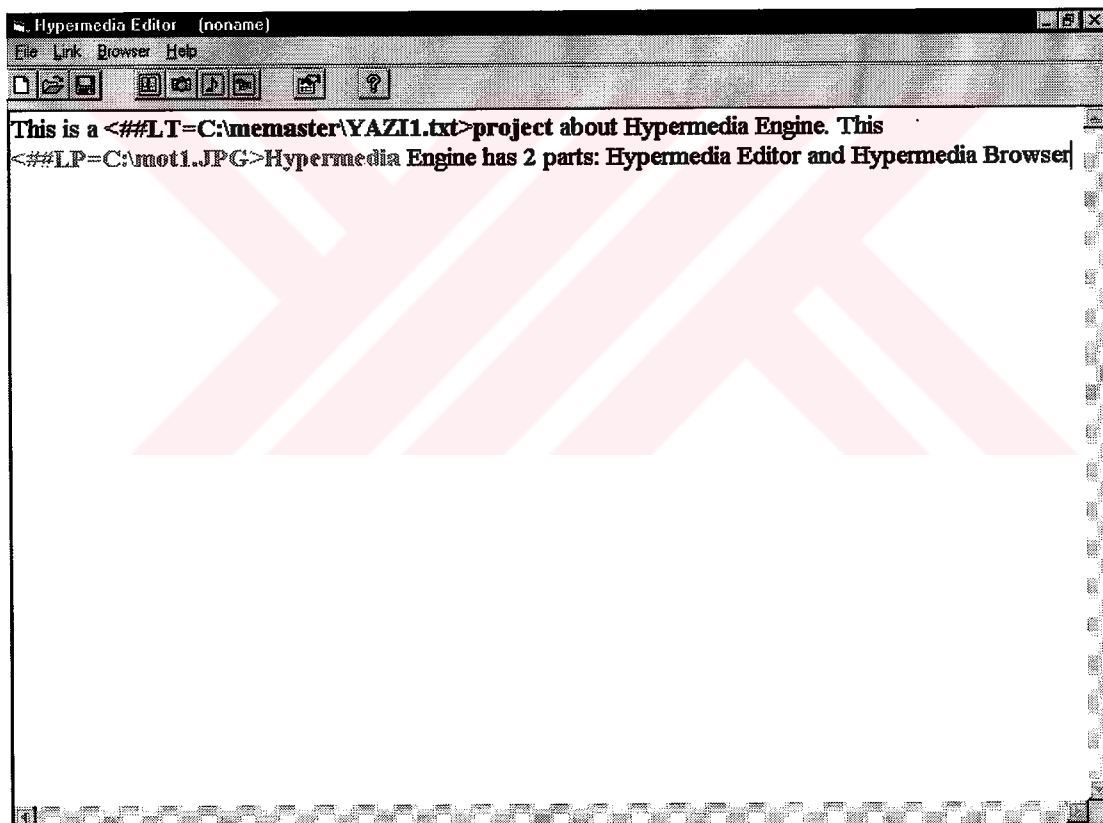


Figure 8.3 Adding Link to Picture

- Files with “txt” extension for Link to Text
- Files with “jpg, bmp, gif, ico” extension for Link to Picture
- Files with “wav” extension for Link to Sound

- Files with “avi” extension for Link to Movie

When the user finishes editing of hypermedia document he should save his work. To save a hypermedia document Save Document button should be clicked. After clicking a saving window appears and user save his work with a specified name and specified location.

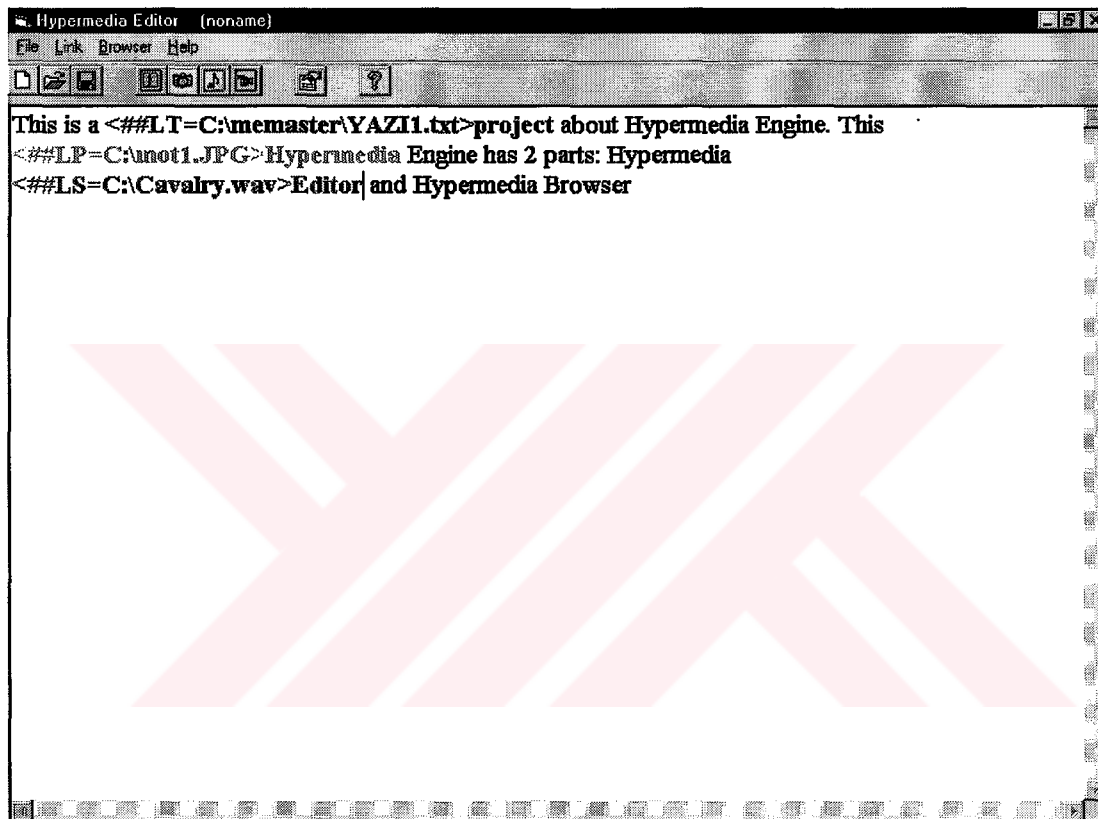


Figure 8.4 Adding Link to Sound

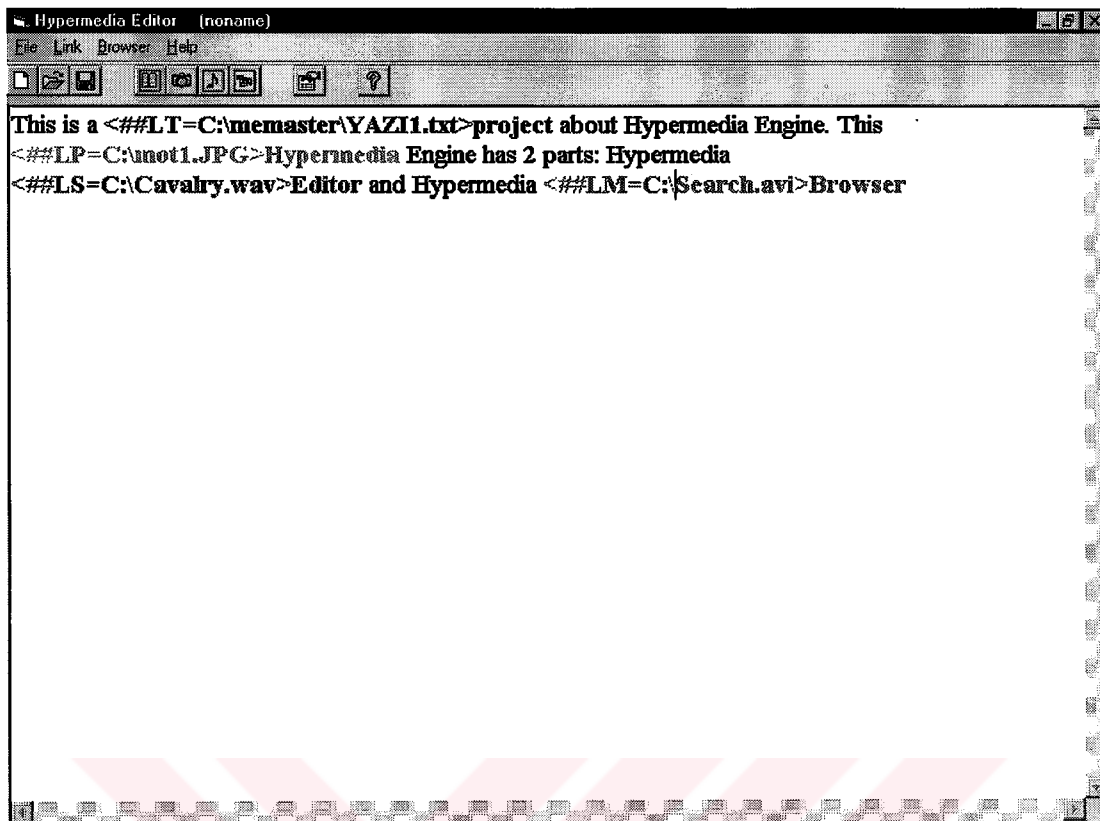


Figure 8.5 Adding Link to Movie

8.2 Using Hypermedia Engine

Hypermedia Browser can be run by clicking Browser button on toolbar of Hypermedia Editor.

To open a hypermedia document user should click open button on toolbar of Hypermedia Browser. An open window helps user to choose the hypermedia document.

After opening hypermedia document user can use this document. If a word in the document has a link than it has different color than black and its font type is bold. For example when the user click on a blue word, a text appears in TextBox. Different colors show different type of links.

User can use ScrollBar to scroll hypermedia document. It scrolls hypermedia document line by line.

Assume that we want to browse the hypermedia document we created in Hypermedia Editor section. We open this document by pressing Open button in toolbar. Hypermedia Browser looks like in Figure 8.6.

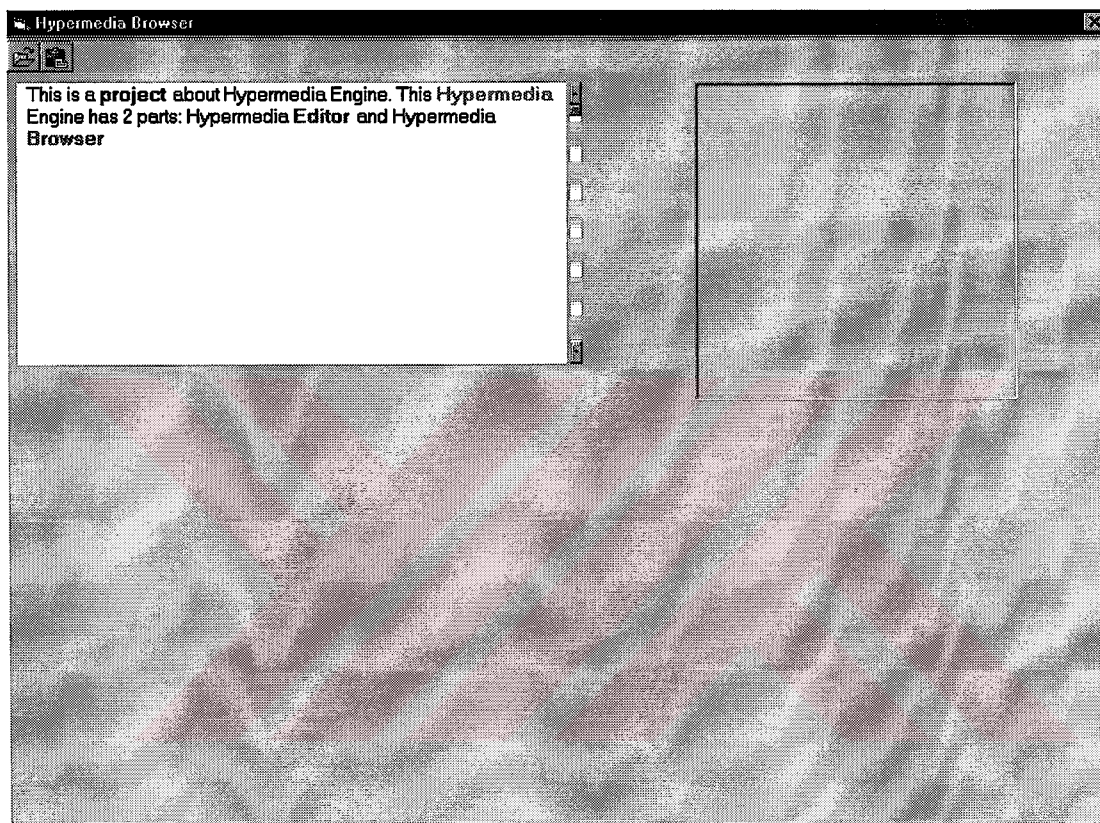


Figure 8.6. Hypermedia Browser

When we click on the word "Hypermedia" it links to a picture. This picture was specified by us in previous section. After this step We screen looks like in Figure 8.7. When we click on the word "Browser" it links to a movie and screen looks like in Figure 8.8.

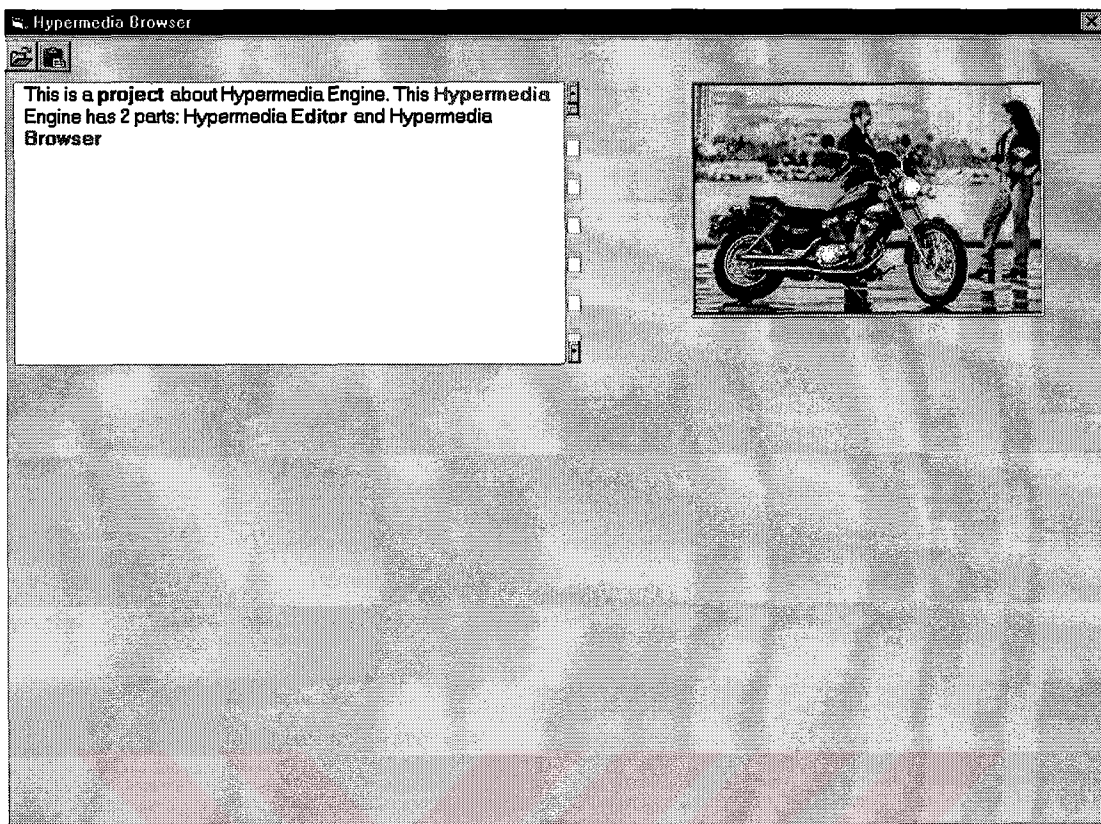


Figure 8.7. Hypermedia Browser with Link to Picture

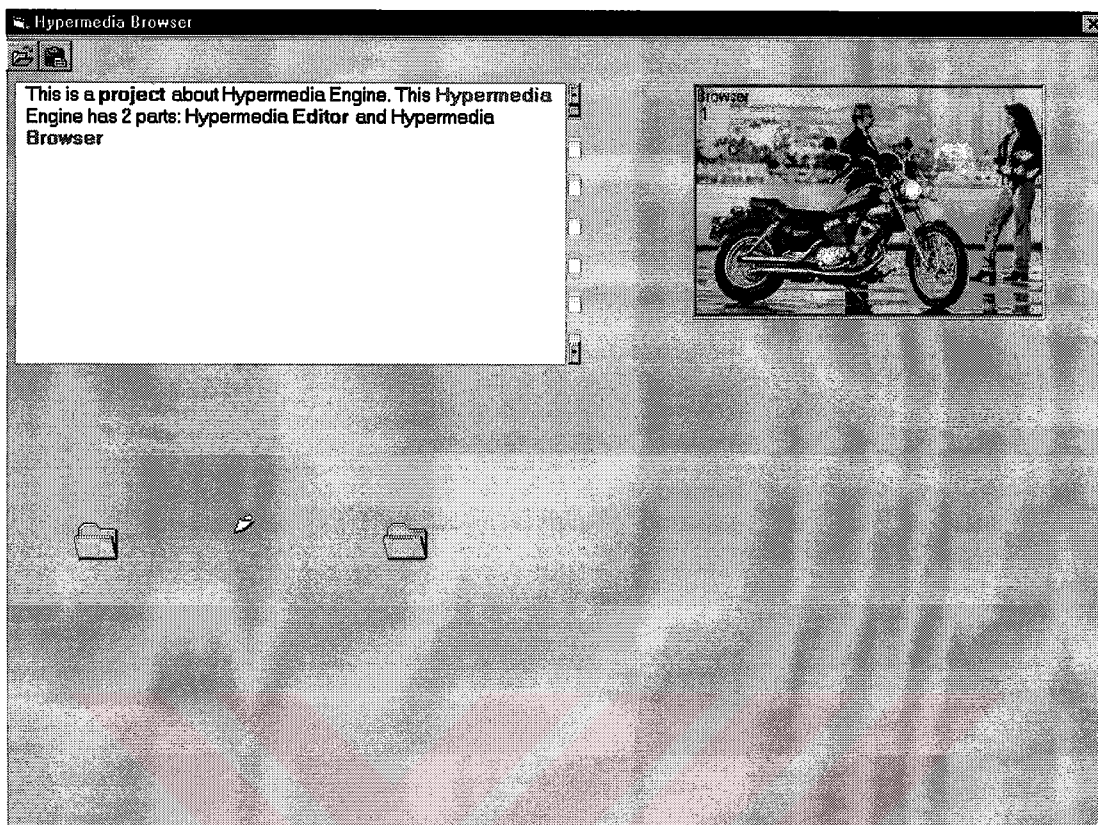


Figure 8.8. Hypermedia Browser with Link to Movie

REFERENCES

Scott Jarol (1994). Visual Basic Multimedia. Coriolis Group Books.

[Http://www.gris.uni-tuebingen.de/gris/proj/guis/Papers/DISS/node7.html](http://www.gris.uni-tuebingen.de/gris/proj/guis/Papers/DISS/node7.html)

[Http://cbl.leeds.ac.uk/nikos/tmp/hypermedia/section2_7_2.html](http://cbl.leeds.ac.uk/nikos/tmp/hypermedia/section2_7_2.html)

[Http://web.signnet.com.sg/~fredling/pg0000007.htm](http://web.signnet.com.sg/~fredling/pg0000007.htm)

[Http://wwwis.win.tue.nl/2L670/cgi/get.cgi/none/chapter0.html](http://wwwis.win.tue.nl/2L670/cgi/get.cgi/none/chapter0.html)

[Http://www.web.co.za/mg/pc/history/multimedia-pcs.html](http://www.web.co.za/mg/pc/history/multimedia-pcs.html)

[Http://the-duke.duq-duke.edu/notes/week_5/hypr1.htm](http://the-duke.duq-duke.edu/notes/week_5/hypr1.htm)

[Http://wwwx.cs.unc.edu/~barman/HT96/P46/HT96BapatEtAl.html](http://wwwx.cs.unc.edu/~barman/HT96/P46/HT96BapatEtAl.html)

T.C. YÜKSEK ÖĞRETİM KURULU
DOKÜMAN YAYIN MERKEZİ