THE DEVELOPMENT OF TOOLS FOR THE CONSTRUCTION OF INTERACTIVE INFORMATION SYSTEMS

by

Larry E. Johnson
A thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of Master of Science (Mathematics and Computer Science)

Golden, Colorado

Date 8.31.94

Signed: [Signature]

Larry E. Johnson

Approved: [Signature]

Dr/Jean Bell
Thesis Advisor

Golden, Colorado

Date 8/31/94

[Signature]

Dr. Graeme Fairweather
Professor and Head,
Department of Mathematics and Computer Science
ABSTRACT

An Interactive Information System is a computer information system that allows the user to search through the data interactively. The information is set up according to topic. The user can browse through the data and skip the data subjects that are not of interest. This paper describes a set of tools that can be used for the development of such a system. This set of tools is for the development of a system for the distribution of scientific meta-data and data on CD-ROM (compact disk read only memory). The tool set has built-in functions that allow a developer to easily import text, graphics, and build a hierarchical menu system. These built-in functions allow a scientist to build an Interactive Information System without having to learn the complicated programming language of the multimedia interface builder. The system asks the developer questions and based on the response of the developer, it builds the Interactive Information System. The ease and the reduction of time spent on development are two of the motivating factors of this project.
# TABLE OF CONTENTS

ABSTRACT ................................................................. iii
LIST OF FIGURES ............................................................ vii

Chapter 1: Introduction ................................................. 1
  1.1 Motivation .............................................................. 2
  1.2 Overview of the System ........................................... 3
     1.2.1 Menu System .................................................... 4
     1.2.2 Text Import System ........................................... 4
     1.2.3 Bitmap Windows .............................................. 4
     1.2.4 Full Text Searches .......................................... 5
     1.2.5 Printing Text .................................................. 5
     1.2.6 Converting Graphics to Bitmaps ........................... 5
  1.3 Summary .............................................................. 6

Chapter 2: Interactive Information and Multimedia Systems .... 7
  2.1 A Brief History of IIS ............................................. 8
  2.2 Generations of User Interfaces ................................. 9
  2.3 Uses of Multi-Media Present and Future .................... 12
  2.4 Multimedia Standards .......................................... 14
  2.5 Multimedia Conferencing ...................................... 15
  2.6 Objects and Their Interfaces ................................. 16
  2.7 Code Generators .................................................. 18
  2.8 Mosaic .............................................................. 19
  2.9 Other Multi-Media Applications .............................. 20
     2.9.1 Multimedia and the 1996 Summer Olympic Games ... 20
     2.9.2 Music and Multimedia ..................................... 21
     2.9.3 Arctic Data Interactive .................................... 22
  2.10 Summary ............................................................ 22

iv
Chapter 3: A Description of ToolBook and Image Alchemy ......................23
3.1 A Description of ToolBook .................................................................23
   3.1.1 Contrasting the Types of Development Products ..................23
   3.1.2 Why ToolBook? .................................................................24

3.2 Selection of a Graphics Conversion Package ................................. 26
   3.2.1 Graphic File Format and Conversions Concepts .................... 26
   3.2.2 Comparing Software Products ...........................................28
   3.2.3 The Selecting of Image Alchemy .........................................30

Chapter 4: How to use the Tool Set ......................................................31
4.1 Installation of Applications and the ToolSet ..................................34
4.2 Using the Menu System .................................................................35
4.3 Importing Text Files into ToolBook .............................................39
4.4 Text Search ....................................................................................43
4.5 Printing Text ..................................................................................43
4.6 Importing Graphic Files into ToolBook .........................................44
4.7 Testing the Development Tool Set ................................................47
   4.7.1 Getting the Information out of Arctic Data Interactive .......... 48
   4.7.2 Building the New Arctic Research Plan ...............................48
   4.7.3 Problems Encountered While Testing ..................................49

4.8 On Line Help System .................................................................50

Chapter 5: Research Issues and More About the Code .........................52
5.1 Detailed Information About ToolBook ......................................... .52
5.2 Overview of Code Structure ........................................................56
   5.2.1 Requirements Analysis .......................................................57
   5.2.2 What the Development Tool Set Looks Like .....................58
   5.2.3 Structure of the Finished IIS ...............................................61

5.3 Hierarchical Menu System ..........................................................62
5.4 The Text Import System .............................................................65
   5.4.1 The Asymetrix Text Import System .................................65
   5.4.2 Modifications Made .........................................................66
   5.4.3 A Description of the Code for the Text Import System .......66

5.5 Text Search System ....................................................................69
5.6 Graphic File Conversion System ...............................................69
5.7 Graphic File Import System .......................................................71
LIST OF FIGURES

Figure 4.1 A Main Menu Page ............................................................ 32
Figure 4.2 A Finished Text Page .......................................................... 33
Figure 4.3 First page, Start building book button ................................. 35
Figure 4.4 On line help ..................................................................... 36
Figure 4.5 Make menu button ............................................................. 37
Figure 4.6 Build menu request window ................................................ 37
Figure 4.7 Topic name request window ............................................... 38
Figure 4.8 Topic description request window ....................................... 38
Figure 4.9 Built menu page .................................................................. 39
Figure 4.10 Continuing the menu system ............................................. 40
Figure 4.11 Example of menu structure .............................................. 40
Figure 4.12 Start of import text system ............................................... 41
Figure 4.13 Text file browser window .................................................. 42
Figure 4.14 Import graphic file request window ................................... 44
Figure 4.15 Graphic button caption request window ............................ 45
Figure 4.16 Graphic file browser window .......................................... 46
Figure 4.17 Example of complected text page ...................................... 47
Figure 5.1 ToolBook’s Background, Page and Layers ............................ 53
Figure 5.2 ToolBook’s Page, Background, Book and System ............... 54
Figure 5.3 ToolBook’s Object Hierarchy ............................................. 55
Figure 5.4 First page, Start building book button ............................... 59
Figure 5.5 On line help page .............................................................. 60
Figure 5.6 Page eight of the development tool set ............................... 61
Figure 5.7 Page nine of the development tool set ............................... 62
Figure 5.8 Page ten of the development tool set ................................. 63
Chapter 1: Introduction

This paper describes the development of tools to build an Interactive Information System. An Interactive Information System is a hierarchical menu driven system that contains and displays information in the form of documents containing text, pictures, graphs, sound, animation, data and meta-data. The main advantage of an Interactive Information System is that a user can browse through the information and skip areas that are not of interest.

This research project provides an integrated set of tools for building an Interactive Information System. The type of Interactive Information System that will be built using these tools is for the distribution of scientific data and information related to that data on the CD-ROM (compact disk read only memory) format. These tools will help in the production of data sets for the sharing of data with researchers and other people who wish to gain information in a research area. The tools will make the process of producing interactive data sets on digital compact disc easier and will reduce the cost of putting the information in a useable and convenient form.

There are three major groups of end users of an Interactive Information System (IIS). The first are scientists and researchers, who will have a strong interest in the data and how to retrieve it from the digital compact disc. The second group will be decision makers in industry and government, who are interested in the results of a research project and how it will affect them. The third group will be educators, who will utilize the Interactive Information System (IIS) in teaching new areas of research.

Because of the diverse groups of end users, the IIS will have to contain information for more than one targeted group. To make the building of the IIS easier and cost effective, a set of tools is needed. This thesis project brings together a standard set of tools to be used in the development of IIS's for scientific research.
1.1 Motivation

Interactive Information Systems are an important part of the developing information age. Large amounts of information are available to everyone. Once the information is received, the problem is that it is difficult to make sense of it or to get to only the necessary information. That is where an Interactive Information System comes in. With multimedia features, an Interactive Information System (IIS) can hold text, graphs, pictures, sounds, and animation. With a hierarchical menu system and full text searches, the information that is in the system is easily accessible for the end user. People learn from methods other than reading text; for example viewing a video clip. Interactive Information Systems have the features that allow a user to learn from several different media sources.

The first goal of this thesis project was to find out if a usable set of tools can be developed given the present software that is available for public use. Once this was determined to be feasible, the development of a tool set started. Another motivating factor in developing an IIS tool kit was the desire to provide a standard consistent feel to an IIS. Through this integrated set of tools, consistency can be brought into the building of IIS’s. With these tools, different IIS’s developed by different researchers will have the same feel and continuity among them, even though they were developed by different people.

With the proper tools in place, the development time, and therefore development cost, for an Interactive Information System can be cut down. The biggest benefit is that the developer will not have to learn and program within the programming language of the multimedia interface builder (for example ToolBook OpenScript). The learning curve for the programming language is steep. The tool development set in this thesis project will automatically generate the code for an Interactive Information System (IIS).

In order to eliminate the need of a specialist to do the programming of an IIS, a easy-to-use set of tools were developed. With this easy-to-use tool set, the
researchers that are a part of the research project that is going into the IIS will be able to develop the IIS themselves. A better end product results when developers who are familiar with the information that is going into the IIS, construct the IIS. The scientists who produced the information and data and know best how to describe it.

1.2 Overview of the System

This project collects together the necessary tools to build an Interactive Information System (IIS). IIS is a specialized field that combines database and multimedia areas of computer science. The type of IIS that can be built, as the result of this work, is for the distribution of scientific data and information related to that data. The IIS will contain all the information pertaining to a research project (papers, graphs, pictures, etc.) and the data that was a part of the research project.

Many people were interviewed so that a group of useable tools could be determined. These people included those who have developed IIS's in the past, scientists who have been on large research projects, and people who may become end users. Some of the tools that these people recommended as a basis for a growing tool set are routines for building a menu system, the ability to import text files and the ability to import graphic files.

The PC platform was used. The major software used is ToolBook, an interface and multimedia builder for Windows. After the development of the IIS using the development tool set, the IIS could be distributed on CD-ROM or made available at a FTP site. The PC platform was chosen because of the availability of PCs to our target audience.

ToolBook is a general multimedia interface builder for Windows. ToolBook has advantages and disadvantages. Some of the advantages are: it is inexpensive, it runs on the PC platform, and it is capable of performing all the required tasks for the IIS. The main disadvantage is the steep learning curve for programming ToolBook. Therefore, several tools were set up in ToolBook that meet the needs of
Interactive Information Systems and help speed the development of a IIS. In this section the tools that were developed for ToolBook are explained.

The following is a description of these tools that will be a part of a standard tool set. As work continues in this area and as the need arises, the tools that are a part of this set will continue to grow.

1.2.1 Menu System

The menu system is a recursive routine that allows the developer to build a hierarchical structure for the organization of information according to topic. Each level of a menu system constructed with the tool kit contains a listing of the different subject areas that make up the research project, along with a description of each area. Special small text boxes give a brief description of the subjects and appear when the subject field is entered with the mouse. If the subject field is clicked on, it will enter the user into the first page of the chosen subject.

1.2.2 Text Import System

The text Import system allows a developer to easily import text into an IIS. Text about a research project can be imported from an ASCII file and placed into the text window. This will make adding information about the sample research project easier and more convenient. Buttons are automatically placed on each page that allows the developer and user to move from one page to another within a given paper or return to the menu system.

1.2.3 Bitmap Windows

Bit map windows allow a developer to display figures that are related to the text of an IIS. A bitmap is a graphical representation of a picture. In ToolBook, a picture must be in the format of a bitmap in Microsoft Windows. Within an IIS
constructed with the tool kit, there are windows to display images that help explain the text.

1.2.4 Full Text Searches

Full text searches are a way to find words or word phrases. The development tool set will automatically set up an option so that the user can do a search for text that is within the IIS. This will allow a user to make searching for given information faster. When a word or a group of words are typed in, the system will start searching at the beginning of the IIS constructed with the development tool set. If the user wishes, they can continue the search through the rest of the text within the system. The ability to search is handy if the word or word phrase occurs more than once within the text.

1.2.5 Printing Text

For some people, reading text on a computer screen can become tiring. To solve this problem a routine that prints out the text in the IIS is available. With this routine a user can read the text away from a computer, and send the text to others in the hard copy form.

1.2.6 Converting Graphics to Bitmaps

Scientific visualization is becoming a very important part of scientific work. It allows a person to see a visual representation of collected or simulated data. Many different software packages that do this are already available. The problem is the graphical output of these packages are not in Microsoft Windows bitmaps. Therefore, a program must convert graphic files into Windows-type bitmaps.

There are several products currently available than can do graphic file conversions, from public domain software, shareware and commercially produced
software. Because there are so many products available, it was important to choose the proper package or collection of packages to do the needed conversions. Then the conversion software package was incorporated into the development tool set.

1.3 Summary

As the need for getting scientific information out to more people increases, the need for putting the information in a useable form will become more important. The CD-ROM format seems ideal: it allows six hundred mega-bytes of information to be put on one disk.

The key in cost effective production of an IIS on CD-ROM is to group together the necessary tools that accomplish the job easily and quickly. Since time is costly, a useful set of tools needs to be collected together that will speed up the development of the IIS. Not everyone has the time or knows where to look when they need some software to do a particular task. The tools have been selected so that much of the repetitive programming can be eliminated. Complete documentation will be available for the developers to cut down on the learning curve. These are key factors in making the tool set successful.

The rest of this thesis describes the project in more detail. Chapter 2 presents a review of work that is presently being done within the fields of Interactive Information Systems and Interactive Multimedia. Chapter 3 is a description of the off-the-shelf software products used for this research project. Chapter 4 describes the IIS Tool Set in more detail. Chapter 5 explains the code, and conclusions based on the research project. Chapter 6 sets forth areas for further continued work.
Chapter 2: Interactive Information and Multimedia Systems

Interactive Information Systems is an area of growing interest within computer science. But what exactly, is a IIS? The first thing that needs to be done is to define a group of terms that apply to IIS's.

Interactive- (1) Pertaining to a program or system that alternately accepts input and then responds. An interactive system is conversational, that is, a continuous dialogue exists between user and system. Contrasted with batch. (2) Pertaining to exchange of information between a user and a computer. [IBM Dictionary of Computing, 1994]

Interactive also refers to time-dependent real-time data communications, typically one in which a user enters data and then awaits a response message from the destination before continuing. [Prentice Hall’s Dictionary of Computing, 1992]

Multimedia- In computing, this term refers to the presentation of information on a computer using sound, graphics, animation, and text. [Prentice Hall’s Dictionary of Computing, 1992]

Multimedia is essentially computerized integration of text, sound, graphics and video. Interactive technologies allow the user to instantaneously interface with multimedia devices and utilize the software to its full potential. Using interactive multimedia, users can access and manipulate information in numerous ways. The software is designed to be used in a non-linear fashion, similar to the way people think and interact, with the freedom to branch off into tangents and related subjects. [Graduate Engineer January 1994 p.24]

Interactive Media- Media that derives input from the viewer to determine the content and duration of a message, thus making plausible individualized program material. Synonymous with interactive multimedia. [IBM Dictionary of Computing, 1994]

These definitions are a good place to start. A more concise definition of IIS, as it applies to this paper, is given as: A collection of data in the form of text, picture, sound, and animation that is organized within a computer system where a user can select the different parts of the information interactively.
2.1 A Brief History of IIS

IIS has a very interesting history. Interactive Information Systems really began (within the context of computers) back when computers were first used for the storage of information. Commands could be typed into the computer system and the result of the request was returned. This concept grew into what we know as Database Management.

During the late 70's and early 80's, work was done to use computers as a learning tool where a person could read some text and then be tested on the concepts of the text. Based on the answers given by the user the computer could give the user supplementary material to help in learning the proper concepts. This was the beginning of Computer Based Training.

Also during the late 70's and early 80's, graphical user interfaces were developed. End-user computing is more than 15 years old, as compared to batch style computing. However, recent improvements in hardware technology have improved dramatically, making possible changes to the interface between the computer and end user.

The most notable examples are the growing popularity of graphical user interfaces (GUIs) such as Apple’s Macintosh desk top, Microsoft’s Windows, and IBM’s Presentation Manager. An extension of the interest in GUI’s appears in the increased attention to interactive multimedia. [Reisman, 1991]

Also, GUIs were placed on database systems to make it easier for a user to get to the data within the database system.

In the mid to late 80's, a new field called multimedia put together sound, graphics, animation and text. At this time, what was once known as Interactive Information Systems was swallowed up by the growing field of Multimedia. Within the last year Interactive Information Systems are emerging again under the new name of Interactive Multimedia. Within Interactive Multimedia is Interactive
2.2 Generations of User Interfaces

Over the years, within computer science there have been systems developed that help in the building of user interfaces. These systems vary from routines that help manage text to knowledge-based systems that build the interface to code. Hix (1990) stated: David Kasik coined the term 'user-interface management system' in 1982. However, the concept of systems to support the development and execution of the interface existed well before then. What many considered as the first user-interface management system called Newman's Reaction Handler was developed in 1968. ToolBook, the software used in this project, is considered a third generation user-interface management system. In this section, user-interface management systems will be looked at with regard to what user-interfaces management systems have been, where they are going, and the role of multimedia within user-interface management systems.

At this point in time, there are four different generations of user-interface management systems. What are now called first generation user-interface management systems, are predecessors to what became user-interface management systems. The second generation systems allow for development of user-interfaces but are limited in what type of interfaces that can be developed and require large amounts of programming. Third generation user-interface development systems are more flexible in the type of interfaces that can be built and are a move towards a system for the non-programming developer. Fourth generation systems have even more flexibility and ease of use for the developer, and have the beginnings of artificial intelligence use for helping in development. Each of the different generations of user interface management systems will be described in more detail.

First generation user interface management systems are systems that allow for building prototype interfaces or are display managers. The prototype interface
builders are used in the design process to make a mock-up version of the interface. After this is done, programmers do the real development of the interface in some programming language. Display managers do help in some of the interface building; for example, menus or other operations. However, they are very limited in what they can do. These systems tend to be for text based interfaces. An example of a first generation interface management system is Act/1. Act/1 has had more than 100 users and has been widely applied and evaluated in the development of interactive systems. [Hix, 1990]

The second generation of user interface management systems focus on the execution of the interface itself. The ability to do this varies very much from one product to another. These systems do little to support design of the interface, human factors, or the user. These systems are mostly used by programmers but are the beginning of graphical interface systems. A good example of a second generation interface development management system is Kasik's Toolkit. Kasik's Toolkit user interface management system (formerly called Tiger) is part of a larger tool kit for developing highly interactive graphics-based applications. A specialized dialogue language contains declaration and control structures that let a programmer construct a linear dialogue sequence. The Toolkit user interface management system had been used in a variety of applications, including 3D geometry construction and manipulation, drafting and design, and space-station analysis. [Hix, 1990]

Third generation user interface management systems are the beginning of a shift away from programmer development. These systems allow for such features as: windowing, use of a mouse, and object oriented concepts. Third generation systems are themselves interactive and work very well for developing interactive applications. They have more flexibility in the types of interfaces they can construct and contain functionality that is not found in previous generations. A couple of examples of this type of interface management system are Hypercard by Apple and ToolBook by Asymetrix. This project uses ToolBook as one of the off-the-shelf
software products.

ToolBook is interface builder with an available set of multimedia extensions. It can be used for interface and application development and development of multimedia applications. In chapter three, ToolBook is compared to Macromedia Authorware Professional [Macromedia Inc., 1993]. Macromedia Authorware Professional is a multimedia development package and not a user interface management system, Macromedia Authorware Professional can be used to build multimedia applications but it doesn’t have all the features and abilities to connect to other systems that ToolBook has.

The third generation of user interface management systems is the era that has firmly established them as a desirable approach to interface development. The importance of the interface is now well accepted, so the need for tools to support development to the interface is also now well recognized. The increased functionality and usability of the third-generation user interface management systems have let them reach even further into the software-engineering realm and support even more phases of the life cycle. [Hix, 1990]

Fourth generation systems continue to increase functionally over third generation systems and begin the use of knowledge bases, expert systems, and other emerging artificial-intelligence techniques. These systems were created for rapid prototyping and development of each phase of the interface development life. There is a move away from having programmers doing the development of the interface and related systems. An example of a fourth generation user interface management system is the NeXT Interface Builder.

One of the more comprehensive and easy to use systems in NeXT's Interface Builder in NeXT Step. Through it, you can create the interface by direct manipulation of desired interface objects. You can connect objects and make them interdependent to produce direct manipulation interfaces. [Hix, 1990]

Another system that could be considered a fourth generation system is called FOCUS. [Edmonds, 1991] FOCUS is a system that builds user interfaces to existing software systems using expert systems to help in the development of the interface. Because FOCUS builds an interface for existing software, FOCUS uses
an expert system to determine what features are needed in the interface and then builds the interface for the existing software.

ToolBook has all the features of a third generation system, and some of the features of a fourth generation system, but lacks artificial intelligence assistance in development. The direction in both research and industry is to have interface developers be non-programmers: thus, the fourth generation systems have filled this demand. Part of the goal of this project is to move interface development away from the programmer.

User interface management systems should improve the interface developer's productivity. They should shorten the development time and make the developer's job more pleasant by directly supporting the interface production tasks. [Hix, 1990]

In order to do this successfully with a third generation system, a set of easy-to-use tools were developed.

All of the fourth generation interface management systems looked at run on UNIX work stations. None currently run on a PC platform. It was determined that economics would designate the use of the PC platform for many applications. Fourth generation interface management system would help in the development of an IIS, but they are not available for the target hardware platform. Therefore, this research concentrates on the building of easy-to-use tools within a third generation user interface management system.

2.3 Uses of Multi-Media Present and Future

At the present time, most multimedia-based software is used in education. Multimedia has been sold as a way to learn complex information at the student's own pace with the use of pictures, sounds, and animation to help in learning the information. However, the applications of multimedia to business and entertainment are just now being realized. The major reason for the increase of multimedia in these areas are the availability of inexpensive hardware and easy-to-use
development tools. A growing market of users who want to use multimedia products for business, education, and entertainment makes for a promising future for multimedia. Multimedia is changing the way people think about computers and how people learn.

Applications used by physicians and lawyers are two of the most exciting areas of using multimedia as an information tool. For physicians, patient records can be kept in a system that has multimedia features. Because the majority of the information that physicians use are in picture form, (ultra-sound, CAT Scan, EKG, and others) the information can be saved in a multimedia system. Then this information can be retrieved easily by any physician in case of an emergency. Also, a multimedia system containing case studies would help a physician to correctly diagnose a patient. There is also the ability to send information to a colleague across the country to consult on the findings of a test in real time. [Gertman, 1994]

For lawyers, using multimedia may be as common-place as using word processors within twelve years. [Marks, 1993] Documents that lawyers present in the form of text can be enhanced by using multimedia features. A brief to the Court could have, along with text, full-motion video clip, pictures, and graphics that help in presenting the case of the client. These types of documents can be sent via electronic mail to all the people who need a copy of the brief. Video is already being used in the case of wills, so why not add the multimedia feature to legal documents. This will help in better explaining the intent of the people involved. Computers and the software that runs on them are becoming easier to use.

They (multimedia software and computer systems) will be designed so that nontechnical users can easily learn an astounding range of features in a relatively short time, even without tutoring. Almost everyone will find it easy to communicate using multimedia computers. Doing so will be routine. [Marks, 1993]

Multimedia will give the people who communicate for a living a new set of tools to explain the ideas they have. It can help lawyers and other professionals be more efficient in the transfer of ideas and knowledge. [Marks, 1993]

It seems that the majority of the research work being done in the area of
multimedia is either in building easier-to-use multimedia and interface development systems, or making the interface design easier for the end user to understand, use, and learn from. There have been studies on how people learn the best, and how to apply the present technology to help the learning process [Korfhage, 1992]. Some researchers are evaluating the types of user interfaces and looking at what different interfaces are more productive for the end user [Klinger, 1992]. By evaluating multimedia interfaces and how users of such systems learn, developers are able to build better multimedia learning interfaces. Researchers agree that with a fair amount of interaction between the multimedia product and the user, comprehension and learning increases [Lichfield, 1989].

Two basic types of program modes exist in hypermedia (multimedia development). One is linear (like the flow of most books or movies). A user looks at a menu, chooses any of various "lessons" to enter, and emerges from it having learned something specific. This is often the preferred way for industry training or teaching a science course. Nonlinear types are often the most fun. You see connections that you wouldn't necessarily see and get creative leaps. [Adam, 1993]

2.4 Multimedia Standards

With Multimedia still a young and growing field, there is a need for some standard formats that allow for device-independent cross-platform applications. Presently there are two standards for storing video, sound, and other data on compact disk. These standards are: Compact Disk Interactive (CD-I) and Digital Video Interactive (DVI). Each of these standards has a completely different end-user group. The CD-I Format is targeted for non-computer users and mostly for entertainment. But, the DVI format is targeted for the multimedia products running on PC-DOS.

The CD-I product can be defined as: An interactive audio/video/computer system, developed by Sony and Philips for the consumer market, the standards for which are known as the 'Green Book'. [McDaniel, 1994] This machine looks more
like a VCR than a computer, it hooks up to a television and is run through a controller similar to the remote control of a television. It is intended for use by users who may not use a computer, but requires special hardware for running the products.

On the other hand, the DVI format is defined as: An integrated video, audio, and Graphics technology allowing all forms of data - full motion video, still images, graphics and text - to be displayed from any digital source. DVI allows real-time compression and decompression as well as display of digital graphics and full-motion video with audio. [McDaniel, 1994] The market for this format is in the education and business multimedia applications running on multimedia PCs. The main advantage to the DVI format as compared to the CD-I format is the real time compression and decompression. This allows for more video and audio data to be placed on one CD-ROM [Davis, 1989].

These two formats have to do with how the data is stored on the CD-ROM itself. Each of these two formats has its own niche in the market and is motivated by the different industry needs. As far as this research project is concerned, the end product is designed to be run on a PC and should be in the DVI format. The targeted end-user for this project is someone who has a multimedia computer, not an entertainment-based product.

2.5 Multimedia Conferencing

Telecommunications have been an important part of the business world since the common use of telephones started. But with technology changing rapidly and more work done on computers, the idea of holding a meeting via multimedia computer is becoming appealing to business leaders. These systems have to handle video, voice, text, and other data in real-time. A person directs the meeting through the software and each person has the ability to request control of the floor (control of the system to present information to others). Many of the first systems were slow due to hardware constraints. But hardware has changed drastically over
the last couple of years, so usable systems for multimedia Conferencing are more practical than ever.

Some of the first systems developed that could do multimedia Conferencing focused on the software that controlled the system more than hardware problems [Sakata, 1990]. With much of the data and other information being sent over a network, these multimedia Conferencing systems can quickly overload the network. Video images and audio of the person who has control of the floor take up a large amount of band width of the network. Other information has to be sent over the network also. Network configurations, band width, and protocols have to be considered in order to determine which combination will work best of the application [Clark, 1992].

Tools for development of presentations, and storage and retrieval of multimedia applications, also need to be considered. There are more than twenty three products on the market that will do multimedia development [Semich, 1992]. With more and more multimedia applications available on one network system, a database system is needed to store and retrieve multimedia types of files. Object-oriented databases along with relation databases with multimedia extensions are available to help solve this problem [Semich, 1993].

Because of the large size of multimedia applications many problems have been encountered when running multimedia applications on a network system. Advancements in hardware and software have helped solve these problems. But, at this point, there is no single off-the-shelf product that will solve all these problems.

2.6 Objects and Their Interfaces

In Williams (1991) a model of how objects interact in the real world was presented. Mr. Williams proposed how to use his interaction model and apply it to computer-based object-oriented modeling. Because most of the multimedia builders and interface builders available today are object-oriented, the concepts presented
in Williams' paper can be used to build better interface and multimedia development software.

Objects in computer science are a collection of both data structures and the code that operates on those data structures. Other objects or the user don't always have access directly to the data structures. This helps prevent the information in the data structures from being compromised.

In Williams' model, all objects interact with each other through an object interface. This interface defines how the object interacts with a user or with other objects. In this model, the user is considered an object and interacts with the computer through the interface of the keyboard and mouse. Other objects interact through interface connections that the different objects have in common (in ToolBook the interface is called Message passing).

These objects are governed by laws controlling how the object reacts to other objects and each object is considered to have what are called lawful states and unlawful states. When one object interacts with another, it can change the state that the object is in from a lawful one to an unlawful one. When an object is placed in an unlawful state, it is required that the object responds to correct this state. The following quote is presented to illustrate this point.

Consider a man driving a car about to run over a cat. A man is a complex object, but a car is relatively simple, as is the interface between a man and a car. Let us suppose that the sight of the cat in front of the car puts the man in an unlawful state. The laws of the man requires a change to the position of his right foot to render his state lawful. [Williams, 1992]

The way that an object interacts with other objects, the interface between the objects, can be called the object's surface. The point where two objects interact with each other is a point were their surfaces touch. Within an object's surface are all of the routines, procedures and rules necessary for a object to function and interact with other objects. The visible interface to a software product is considered to be the surface of that software that interacts with the user.

Williams' work was undertaken in order to help in the building of better user
interfaces. Much of the work in this area can be applied to multimedia applications also. This model helps in setting up the communication between objects and how they respond to each other.

2.7 Code Generators

Code generators are application specific software that generates lower language code based on options that a developer makes. The development tool set could be considered a code generator. Most code generators available are for developing interfaces or building databases. Developing interfaces in lower languages like C, can produce a large amount of code to do very simple operations. In order to get away from this extensive programming code generators were developed. Some user interface management systems are code generators.

An example of a data base type code generator is ObjectPro by Trinzis Corp [Leach, 1994]. This system allows a developer to build business based data access applications. "ObjectPro, a Windows-based tool, is aimed at developers who want to add business logic to data-access applications." [Leach, 1994] ObjectPro has an object oriented programming language. This language has some feature of both third and fourth generation languages including logic built into it. From ObjectPro’s language the system generates C code. This system can be used for developing applications that interact with Oracle7, Sybase System 10 and Open Database Connectivity Databases.

ProtoGen+ is a Windows application developer that generates C, C++ or Pascal code for Borland, Microsoft, Symantec, or Watcom compilers [Carnahan, 1994]. ProtoGen+ uses the libraries in these compilers to build the Windows interface. This code generator allows developers to quickly develop Windows applications and maintain high performance levels of the application.

ProtoGen+ is a code generator that successfully brings some of the fourth-generation tools and ease of development to Windows programmers who need the performance benefits of a third generation language. Because of ProtoGen+’s comfortable Windows interface,
experienced programmers who have taken the time to learn the product can accomplish routine programming tasks with refreshing speed. [Carnahan, 1994]

This code generator, like most code generators available, is trying to produce applications that have fourth generation programming language ease of development with third generation programming language speed.

2.8 Mosaic

Mosaic is an Internet application that allows a user to access text files, graphic files, and sound files that are on the Internet's world wide web. Connections to different files are made through hyper links in the text files. A user can click on a hyper-text word and have access to files around the world. Mosaic is software the was produced by the University of Illinois's National Center for Supercomputing and is available at no charge. Originally it ran only on UNIX machines, but a version that runs on PCs was made available March 1994. The PC version is difficult to get running, there are several utility programs that need to be loaded properly before Mosaic will work. For PCs that don't have Internet access, a modem and a dial in Internet access provider is needed.

Mosaic is one of the most powerful Internet fount-end Applications. It enables users to easily explore Internet's World Wide Web Hypertext multimedia files. It also brings new life to Internet's Gopher, Archie and Wide Area Information Server data search and retrieval tools. Unfortunately, downloading such data to a PC's hard disk is time consuming and cumbersome. Multimedia documents are particularly slow to download, even over 28.8 - Kbps modems. [Vaughan-Nickols, 1994]

Mosaic is not a inclusive Internet access tool. It can't send and receive electronic mail and is not able to access news groups. But, the features that are provided allow a user to search the world over for information.
When you finally get Mosaic up and running, it resembles nothing you've ever seen online. It's like browsing through a interactive multimedia magazine with endless depth and breadth. And the capability to upload and download both graphics and sound places Mosaic at the forefront of front ends. [Gralla, 1994]

2.9 Other Multi-Media Applications

This section contains examples of some multimedia products. These products are presented here as examples of non-traditional applications for multimedia. The first is a product used to sell a city to the Olympic Board as a possible site for the Summer Olympics. Another, is a real-time music-driven application. The last is a prototype scientific journal.

The Olympics application helped the Olympic Committee in making their decision on having Atlanta host the 1996 Olympics. This software showed how Atlanta with its present facilities and buildings that are to be built by 1996 could host the Summer Olympic Games. The real-time multimedia application, shows how many different parts of computer science can be brought under the name of multimedia. Also, it shows that multimedia can be part of a time-dependent program where speed is of importance.

2.9.1 Multimedia and the 1996 Summer Olympic Games

Fred Dyer, Co-director of the Multimedia Technology Laboratory, and researchers at Georgia Tech developed a system that allows the user to do an interactive fly-over of Atlanta, Georgia. This system allows a user to fly over different parts of Atlanta and view the city as if every thing is ready for the Olympics. At the time that the system was built not all the buildings that would be used as a part of the Olympics had been built. Images of the city were digitized in order to
build a representation of the city within the computer. Then the buildings that were
not built yet had to be added into this computer model. The following is a
dramatization of the system operating.

The dream begins. Flying through the southern skies, you descend
toward the Olympic complex as spectators fill up the newly completed
sporting venues. You take control. With the skyline at your fingertips,
you decide which of Atlanta’s newest architectural triumphs you want
to visit. A flick of your wrist and you are heading to Stone Mountain
to preview the Olympic bicycling and equestrian competitions.
Another twist and you’re flying over the Olympic Village to swoop
down and glide effortlessly inside. Entering the main stadium, you
watch the Golden Athlete stride across the field. [Zepezauer, 1994]

This is a quick overview of what the system can do. Also, it has features
where a user can click on an icon and get more information about given areas of
Atlanta. This is an example of where a multimedia product helped a city get a very
large contract. The multimedia presentation of the city was one of the deciding
factors for the Olympic Committee.

Another project that is related to the Atlanta Summer Olympics is a collection
of multimedia based terminals that help guide visitors to different sites within the
city. Mark Klein is heading a project where a visitor to the city can access tourist
information through touch-screen terminals. The information is available in many
different languages. This system is for helping the many different people that will
be attending the Summer Olympics. [Zepezauer, 1994]

2.9.2 Music and Multimedia

Roger B. Dannenberg has developed a multimedia music performance
system that runs in real time. This system takes input from musical instruments
being played, does some analysis on the data and sends commands to a
synthesizer to play along with the musicians in real time. In addition to the playing
of music the system also does an animated graphical representation of the music
in real time. This type of application is not traditional multimedia but it shows what
can be considered multimedia. This application takes music and animation to the performance level, and deals with problems that do not show up in other multimedia applications. [Dannenberg, 1993]

2.9.3 Arctic Data Interactive

Arctic Data InterActive (ADI) is a prototype CD-ROM science journal containing interdisciplinary scientific information about the arctic region [USGS, 1991]. ADI was produced by the United States Geological Survey as an experiment in getting arctic and global change information to wide number of people, in an easy to use format. The information in ADI is organized according to major topics and sub topics within the major topics. This format allows a user to get to the needed information quickly. ADI has many figures that help in explaining the written text. The format that ADI presented allows users to get information and data about the arctic region easily.

2.10 Summary

Multimedia is a field in computer science that continues to grow rapidly. With multimedia still a young field, there needs to be a set of standard formats that allow for cross-platform operations. There is a trend towards having non-programmers develop multimedia products. Development packages are available to assist with this trend. Different modeling paradigms are being presented to help in the development of these complex systems. Multimedia has changed a lot in the last five years. It is exciting to be a part of the changes that will take place in the near future.
Chapter 3: A Description of ToolBook and Image Alchemy

Within this project, two commercial software products were used. These products are ToolBook by Asymetrix [Asymetrix, 1992] and Image Alchemy by Hand Made Software Inc [Hand Made Software Inc., 1993]. Other products were considered but, based on the requirements of the overall project, these two products were chosen. The following sections describe these software packages and why they were chosen.

3.1 A Description of ToolBook

ToolBook is an interface builder for Windows that allows for multimedia extensions. It has an object-oriented feel and the programming language OpenScript has some object-oriented features in it. External programs can be run from commands within the scripts of different objects. This makes ToolBook good at developing interfaces for other programs and bringing together different software products with a common user interface. More about the structure of ToolBook will be discussed in Chapter 4 and 5.

3.1.1 Contrasting the Types of Development Products

All the multimedia interface builders that are on the market, fall into two general groups, which are: a) right brain and b) left brain:-

Virtually every author tool on the market today can be categorized by the Mind-set of the designer: right brain (focused on pizzazz and surface appeal, with less concern for content) and left brain (emphasizing content and having a user interface that ranges from clunky to nice but never spectacular). [Harriett Hardman, 1993]

An example of a right brain authoring tool is a product called Macromedia Authorware Professional [Macromedia Inc., 1993]. ToolBook by contrast is an example of a left brain authoring tool.
Macromedia Authorware Professional has a nice interface and a user can learn to build applications quickly. But it is very limited; it can't handle large amounts of text, and there isn't a programming language available for redefining the given operations available. Furthermore, it is relatively expensive, at about $4995 for a single machine license.

ToolBook, on the other hand, is difficult for a first time developer to produce a product quickly. Within ToolBook a developer can build an interface by making buttons, fields and graphics in the 'author' mode. The Author mode is similar to a drawing program where a developer can create objects, move the objects or resize the objects. ToolBook also has facilities for handling large amounts of text. With the programming language (OpenScript), a developer can write scripts to accomplish just about any desired process. Data Link Libraries are provided to connect ToolBook to databases, spreadsheets and other commercial products. ToolBook is relatively inexpensive, as multimedia development software goes, at a price of about $695 for a single machine license. The major drawback is that OpenScript is difficult to learn.

It would be nice to have something that is between the two general types of commercial multimedia development tools. Something is needed to handle large amounts of text and multimedia extensions, without being difficult to learn. One solution to this problem is to build some routines within a commercial multimedia development tool so that a developer can make selections within the system that will help in the building of the IIS. That is the approach followed by this research project.

3.1.2 Why ToolBook?

Asymetrix ToolBook began as a Software Construction Set for Windows [Asymetrix, 1993]. It was originally designed to support the construction of interfaces for software and to help in the development of windows applications. Shortly after its original release, extensions were added to ToolBook to provide for multimedia
development. It was one of the first PC-Windows based software-development packages to have utilities for building interfaces, a high level programming language, and multimedia extensions. Even though there were multimedia products available for the Macintosh in the mid to late '80s, equivalent products were not available for the PC until the late '80s to early '90s. ToolBook was selected by the USGS for their first IIS: Arctic Data Interactive (ADI) [USGS, 1990]. The users of ADI were happy with the end results but, because of the expense in programming time of using ToolBook for real applications, plans were not made to produce similar products using ToolBook.

ToolBook has all the features that are needed for the development of the type of IIS that the USGS wants to build. It can hold large text fields, do searches in the text fields, import graphic files, execute external programs from ToolBook, and has the features that allow a menu system to be built. These features make ToolBook an attractive development platform.

ToolBook has many advantages that make it a desirable choice as the development platform for the interactive information system development tool set. The first advantage is that ToolBook's ability to handle large amounts of text. The text fields in ToolBook can store up to ten-thousand characters of text. The second is, ToolBook's ability to run programs that are outside of ToolBook thus makes ToolBook easy to integrate with other software products. Finally, the object-oriented layout of ToolBook helps in developing more structured code.

In spite of all its advantages, ToolBook also has many disadvantages. A person has to go through a painful process in order to learn the programming language. Many programmers, who haven't had any object-oriented experience, have difficulties understanding the layout of ToolBook. The scripting language is at a very high level and there are about 750 key words within OpenScript. This makes for a steep learning curve.

After weighing all of the advantages and disadvantages to Asymetrix ToolBook, the decision was made to use it as the basic development package for the construction of the necessary tools. ToolBook is inexpensive. Previous
products produced with ToolBook ended with a desirable product and ToolBook has the features that will allow tools for an IIS to be produced.

3.2 Selection of a Graphics Conversion Package

The selection of a package for converting graphics file formats began with an investigation of requirements. After talking with experienced people in IIS development, a list of requirements was put together. These requirements are:

1. The ability to convert between many different graphical formats.
2. The ability to resize the graphic picture.
3. The ability to convert without distortion or loss of colors.
4. The ability to convert 24 bit graphic files to 8 bit in order to save space.
5. An easy to use interface.
6. Moderate cost of the software

3.2.1 Graphic File Format and Conversion Concepts

A graphical file is data that is used to display a picture on a computer screen. The format of the data in a graphical file is dependent on what software produced the image and what basic type of graphical format it is. Software companies develop different specific formats depending on issues like color quality, image quality, and file size.

There are two basic graphic file format types that all graphic file formats fall in to. These types are raster images and vector graphics. A raster image is commonly known as a bitmap, where each pixel in a plane is given a location and a color value. Examples of a raster file formats would be: Tag Image File Format (TIFF), ComurServe's 8-bit Graphics Interchange Format (GIF), and Windows bitmap. The vector graphic formats are mathematical representations of lines and objects. Examples of vector graphic format types are: PostScript and Encapsulated
Within a raster graphical file format there is information for each pixel of an image. A pixel is a small dot, thousands of pixels make up a graphic. Pixels can have a location, color, and hue. This information about each pixel is used to build a graphic.

Black-and-white raster images occupy one plane and need only one data bit per pixel to describe the pixel's color values. Color raster images are more complicated. They can be defined with various bit depths: with 8 bits per pixel (Indexed Color, which draws its colors from a set color palette), 16 or 24 bits per pixel (RGB -- red, green, blue -- which uses three planes to express color), or 32 bits (CMYK -- cyan, magenta, yellow, black -- which involves four planes of color). If that weren't enough, a new 48-bit alternative has been proposed. The file size of a raster image is dictated by the numbers of pixels and their bit depth: as a result, hi-resolution color raster images can swell to many megabytes. [Sosinsky, 1994]

The higher quality images in color and resolution produce a large file representation of the image. Some file formats like GIF and JPEG have built in compression algorithms. These algorithms remove redundant information for a group of pixels. For example, several pixels in a row can have the same color and hue, this information is stored once along with the location of each pixel. Other compression routines group together pixels that are almost the same color, but when you compress a graphic file in this way you don't always get the same results back.

Vector graphic file format originated from the first draw and computer aided drafting programs. Each line, circle or other object that is displayed, is represented by a mathematical equation. Each object can have attributes like color and hue. With vector graphic formats, only the objects that make up the graphic are stored and not every pixel. This format is good for computer produced drawings, but if a scanned image is converted into a vector graphic format, the resulting file can become very large. Conversion routines have to make every pixel into an object.
Generally, but not always, vector graphics are more compact than raster images. Sometimes intricate vector graphics require complex descriptions that overwhelm the memory and processing power of your computer and printer. Vector graphics always display or print at the maximum resolution of the output device -- unlike raster images, which print at the resolution of the image itself, meaning that a 72-dpi bitmap looks coarse even on a 600-dpi printer. Vector graphics are sharp and crisp, even when scaled in size. [Sosinsky, 1994]

The Encapsulated PostScript data can be written to a file as text (ASCII) or as hexadecimal (binary). The ASCII format is more widely excepted, but the binary files are smaller, this allows for quicker displaying and printing of the file.

As long as software companies continue to develop graphical file formats for their software there will be a need convert these files. The IIS development tool set has to get all possible file formats into ToolBook. So, there is a need to convert the graphic files into Windows bitmaps.

### 3.2.2 Comparing Software Products

Using the above list of requirements, several graphic packages were considered in order to determine which would work the best for the needs of the project. A group of shareware software products were considered because they were cheap in price however, they couldn't meet all the requirements. They are mentioned here as a learning experience; when you want good software you have to pay for it. The two software packages that met the above requirements the best were HI-JAAK [Inset Systems, 1993] and ALCHEMY [Hand Made Software Inc., 1993].

HI-JAAK runs under Microsoft Windows, and therefore, it has a very good user interface. It comes with interactive user help. HI-JAAK can read in twenty-nine different graphic-file formats and can convert these type of files to eighteen graphic-file formats. It also allows the picture to be viewed and some modifications made to the picture using a paintbrush-style program. Color changes can be made
in the conversion process along with a dithering algorithm, which helps in maintaining the colors during conversions.

The major disadvantage to HI-JAAK is that most of the file types that it can convert between are used only by business applications. For this project, most of the files that are to be converted will come from scientific applications. So, many of the types of files that HI-JAAK can convert may not be useful within this project.

The other product that was considered is IMAGE ALCHEMY. This software product can convert between over fifty different graphic file formats. It is not as limited by the type of files that it can convert to, as HI-JAAK is. IMAGE ALCHEMY also has seven different dithering algorithms. IMAGE ALCHEMY uses a default dithering algorithm which worked well for all the testing that was done on the software product. If converting between some file formats loses some color, a different dithering algorithm could be selected to help maintain the color integrity.

Within IMAGE ALCHEMY, there are five different scaling algorithms. Again, IMAGE ALCHEMY has a default algorithm that worked well during the testing. However, in cases where a small graphic is made very large, some smoothing of lines was needed. To take care of this problem, a different algorithm can be selected to help in the overall appearance of the graphic. The same problems were encountered when trying to scale a very large graphic into a smaller one. These scaling algorithms will be very handy in making a graphic fit within the given graphic windows that are available within the development tool set. For most of these purposes, the default setting works well.

3.2.3 The Selecting of Image Alchemy

There are several advantages to IMAGE ALCHEMY. One advantage is the ability to change 24 bit images into 8 bit images. Since an 8 bit image is smaller in size than a 24 bit image, this can save space within an application if necessary. However, when the colors of the graphic image is important, then it is best to keep it as a 24 bit image. Another major advantage is that it can convert many of the
scientific graphic files into a file format that can be imported into ToolBook.

IMAGE ALCHEMY has two modes of operation to accomplish the same operations: a graphical user interface, and DOS commands typed in at the prompt. All commands for the converting of files can be done through DOS with different options within the command line.

The ability to run IMAGE ALCHEMY through the building of a DOS command-line makes it easier for integration of this graphic conversion system into ToolBook. If a developer selects a graphic file that will not import into ToolBook, a conversion of the file can be done without the developer ever knowing about it. The development tool set checks to see if the file is of a valid graphic file type. If it isn't, the development tool set builds a DOS command-line for IMAGE ALCHEMY, then utilizes IMAGE ALCHEMY to perform the graphic-file conversion more or less automatically.

All of the preceding advantages of IMAGE ALCHEMY make it a desirable software product to do the graphic file conversions for the development tool set. HI-JAAK is a good software product, but integrating it into the development tool set would be very difficult. HI-JAAK can't be executed from a DOS command-line, it has to be run through Windows.
Chapter 4 How to use the Tool Set

This chapter reviews the use of the development tool set from the point of view of the developer. After reading this chapter, a developer should have a good idea how the tool set works. With this information, along with ToolBook's tutor section, a person can develop an IIS. ToolBook tutor is within the ToolBook multimedia development software.

The main purpose of this project was to provide tools for the easy and inexpensive development of an Interactive Information System (IIS). A big motivating factor is that it took over $150,000 [personal communication, Denise Wilshire, USGS, 1994] to develop the first system used for the distribution of scientific data and meta-data for the United States Geological Survey (USGS): the Arctic Data Interactive (ADI) [USGS, 1991]. Since the first project for developing an IIS was expensive, an alternative to this expense had to be found. The tools that have been developed as a result of this research project are the beginning of a software set for the development of IIS's.

With this set of tools, a scientist can develop a system for the storage and distribution of data that was a result of his/her research. The developer (scientist) doing the development of the IIS will not have to learn a difficult multi-media programming language. One of the biggest expenses is the time that it takes for a person to learn the multi-media development software package. In the case of this project, ToolBook Multi-media Interface Builder was selected as the multi-media development software package. Therefore, a group of tools were built so that a developer can build an IIS without spending the time to learn OpenScript, ToolBook's programming language. Within the development tool set, a developer of an IIS can select choices presented, and the tool set will automatically generate the code and objects for the IIS. The developer doesn't have to write the code; the tool set does.

In ToolBook, the end product is called a book. This book is made up of pages and backgrounds. A background can be shared between more that one
page. Both a page and a background can contain objects. An object can be a button, text field or a graphic. All items in ToolBook, including the book itself, can have a script. The script to these items contains the code and data that the object operates on. All items - book, page, background, button, text field and graphic - are used to build an IIS.

From the end user's point of view, when the finished IIS is entered, the user will be placed on the main menu page (see fig. #4.1). In the center of this page is a row of fields that contain topic names. When the mouse cursor is placed in each of the topic fields, a description of that topic will appear in a text box. The user can enter a topic by clicking on the corresponding topic field. When a topic field is clicked on this will bring up another menu page or the first page of text for that topic. Each new menu page would act in the same manner.
1. Background

As required by the Arctic Research and Policy Act of 1984 (Public Law 98–373), a comprehensive Arctic research plan was prepared by the Interagency Arctic Research Policy Committee (IARPC 1987) and submitted to the President, who transmitted it to Congress in July 1987. Section 109(a) of the Act requires a biennial revision to the Plan. This document is the mandated biennial revision.

United States research in the Arctic and for this biennial revision is governed by the goals and objectives agreed upon by the Interagency Committee on February 3, 1986, which include supporting research to implement national policy of protecting security interests, promoting rational development while minimizing adverse effects, and contributing to the knowledge of the environment best studied in the Arctic.

The Act did not provide separate additional funding for Arctic

Figure 4.2 A Finished Text Page

A finished text page has a large text field with a slider bar to the right (see fig. #4.2). All of the text in this field can be read using the slider bar. On the right side of the page there may be a row of buttons that display figures which corresponds to the text in the text field. A figure is displayed by clicking on the proper button. The image can be dismissed by clicking on the image itself. If there are additional pages of text to this topic there will be buttons on the lower part of the page that help in turning to the next or previous page. In figure 4.2 note the button on the lower right corner, when clicked on this button will return the user to the calling menu page.

In the rest of this chapter, everything is discussed from the point-of-view of the developer of a new IIS product. The following sections describe the process that a developer would go through in order to build an IIS. Parts of the development tool
set that ask for information (such as topic name, description, number of topics, filename, figure name and, number of graphic files) that is only part of the developer’s view. Once any part of the IIS is built, that part of the IIS acts in the same way as it does for end users. Within the sections of the IIS already built, the developer moves around the IIS in the same way as an end user.

4.1 Installation of Applications and the Tool Set

In order to use a developed IIS the user needs the following hardware and software. A PC computer with a 286 or higher processor and Windows 3.0 or higher is required to run a developed IIS. Depending on the method of distribution, a CD-ROM reader may be necessary. If the IIS is distributed on a CD-ROM, the ToolBook application program (the developed IIS with the '.tbk' extension) and the ToolBook execution file 'tbook.exe' should be in a subdirectory 'D:\IIS'. From the windows file management select 'run program' from the menu and type in 'D:\IIS\TBOOK.EXE D:\IIS\??.TBK', where '??' would be the name the developer used to save the IIS.

To do development with the development tool set, a PC computer 386 or higher processor and Windows 3.0 or higher is required. Also, a copy of ToolBook Multimedia Development Software version 1.5 or higher (3.0 is recommended) and Image Alchemy graphic conversion software must be installed on your development computer system. On the 'C:' drive of the development computer system a subdirectory called 'C:\IIS\GRAPHIC' needs to be created. All converted graphic files go into this subdirectory. It is suggested that all other graphic files related to the IIS go into the subdirectory 'C:\IIS\GRAPHIC'. Copy the development tool set file that is on the disk in the back of this thesis to 'C:\IIS'. The development tool set has the name 'MENU1_5.TBK' or 'MENU3_0.TBK', these are for ToolBook versions 1.5 and 3.0 respectively. Copy one of the development tool set files to the subdirectory 'C:\IIS'. Access the development tool set from the ToolBook program to start building the IIS. When development is done, copy the ToolBook 'run time
files’ into 'C:\IIS' (see ToolBook manuals); all of this subdirectory structure and the files in it make up the finished IIS. The subdirectory structure and the files in it can be used to make a CD-ROM and distributed to end users.

Figure 4.3 First page, Start building book button

4.2 Using the Menu System

The hierarchical menu system is the backbone of all the tools that are a part of the development tool set. It allows a developer to organize the information in a way that makes it easy to get to. The information can be divided into different subjects. For example, the subjects may be chapters of a book or report. Each of those subjects can then be broken up even further. This process can be repeated to provide as many hierarchical levels as the developer desires. The process of
building the hierarchical menu system is recursive. Providing more than two or three levels, however, may make the system confusing for the end user.

Before a developer starts building the menu system, they should go through a process of designing the layout of the system. Clearly defined categories and subcategories need to be set up to make the process of building the IIS easier.

When a developer first enters the development tool set they are placed on a page with a button that starts the building of the menu system (see fig. #4.3). On line help is available for each part of the development tool set (see fig. #4.3 and fig. #4.4). After the developer clicks on the 'Start Building Book' button the development tool set asks for the name of the book, this would be the name of the research paper that the IIS is based on. The development tool set then makes a page and places the button that contains the menu system on the page (see fig. #4.5).
Do you wish to build a menu system?

Yes  No

When the 'Build Menu' button is clicked on this activates the menu system. The first thing that the menu system does is to ask the developer if they wish to build
a menu system (see fig. #4.6). If not, then the development tool set removes the menu system button and gets a copy of the import text system. If the menu system is to build a menu, then it asks what the first topic name is: for example, Chapter 1 (see fig. #4.7) and a brief description of the topic (see fig. #4.8). With this information, the development tool set can build one topic on this level of the menu system. A field is placed on a page with the topic name in it. When the mouse curser is moved into the topic field the description of the topic appears (see fig. #4.9). The reason for this is so a end user can get an idea of what different sections are about before ever entering that section.

After the first topic field is in place the 'Add a Topic' button is copied on to the page. This allows the developer to place as many topic fields on this menu page as there is room for. If the developer clicks on the 'Add a Topic' button and selects 'No', the 'Add a Topic' button is removed from the page (see fig. #4.9).

A click on a topic field will select that topic for the developer. Once in the new topic, the developer can build a new menu in the same way as described
This is chapter one of IISDTS.

This is chapter one of IISDTS.

Figure 4.9 Built menu page

previously or start importing text (see fig. #4.10). This allows for the recursive part of the menu system. If the developer chooses to build another menu, they can go thorough the previously described actions which will allow them to build a second level of the menu (see fig #4.11, sub. #2 and sub. #4).

4.3 Importing Text Files into ToolBook

Most information is available in the form of text. A generic ASCII format is the basic format acceptable to all modern word processors. Scanners can convert the text of a printed paper to the ASCII format. This development tool set utilizes ToolBook’s capabilities to allow ASCII text files to be imported into text fields. The ASCII conversion process removes bold, italics, different fonts or underlined text
Figure 4.10 Continuing the menu system

Figure 4.11 Example of menu structure

features of a paper. However, within ToolBook these features can be added back to the text.

Before starting the process of importing text files, all the text that is to be imported must be subdivided into individual files that are associated with the
different divisions of the menu system. Each text file should relate to a single topic or subtopic within the menu system.

Once this process is completed, the text files may be imported into the IIS. Within each subtopic of the menu system there is a button than when clicked on, will start the text importing process (see fig. #4.12). When the text importing process is activated for the given subtopic, a browser window pops up (see fig. #4.13) and the developer has access to any text file within the computer system that the development is being done on. After the appropriate text file is selected, the text file import system then tells the developer how big the file is.

Because slower computers scroll through text fields very slowly, large text files are broken down into smaller pieces. The text import system of the development tool set automatically breaks the text up and puts about three written pages of text in one text field. The text import system continues to make pages with text fields until all of the text file is imported into the IIS. As the text import system
is creating these ToolBook pages, it puts buttons on each page that allow the user to go to the previous and next page within the subtopic. Another button is placed on each text page that will allow for key word text searches. In addition, on each page a button is placed that allows the user and the developer to go up to the calling level of the menu system. The calling level of the menu system is the page where the developer or user can click on a topic field that goes to the first page of text for that topic.

The IIS development tool set is designed for building a system that provides close integration of text and other information related to that text. Therefore, each subtopic has to contain some text. Subtopics with only pictures and graphs are not allowed.
4.4 Text Search

If a person is looking for a specific word or word phrase, finding it within the menu system can be time consuming. A routine is provided to help those who want to search for a word or word phrase. This is activated by clicking on the 'Text Search' button. This button is located on each menu page and text field page.

When this text search button is clicked on, a window appears which asks for the word or word phrase that the end user wishes to search for. The search begins at the beginning of the IIS and stops at the first occurrence of the word or word phrase. If the string was not found the system responds with an error window. The user has the option to go to the next occurrence of the word or word phrase until the end of the document. When the routine gets to the end, the user is notified of a search failure.

The capability of searching for text within the text field pages adds to the usability of the overall system. The user can scan through all the text, and directly locate each occurrence of a word or word phrase. This routine provides a short cut for the people who want to find specific information quickly.

4.5 Printing Text

Many people prefer to read text on paper, because reading from a computer screen is tiring. Routines for printing will help distribute information from within the IIS. Printed text can be used when there is a need to read the information later, send the information to someone else, or to read when a computer is not available.

At the bottom of each text field page is a button that activates the text print system. When this button is clicked on, it sends all the text of the entire IIS to the printer.
4.6 Importing Graphic Files into ToolBook

An IIS should provide a close relationship between text and pictures (graphic files). Therefore in an IIS, graphic files need to be made available for each page of text. The graphic files may represent graphs, maps, illustrations, graphic representations, and pictures that help to describe the ideas that are in the text. The IIS development tool set provides routines for easily importing graphic files into the IIS as it is being developed.

The development tool set has available on each text field page a button that allows the developer to import graphic files to the text field page. When the developer clicks on the 'import graphic files' button, the graphic file import system asks if they wish to import a graphic file (see fig #4.14). When the 'no' choice is selected, the system removes the import graphics file button from that page and ends. This allows for those cases where no graphic files are associated with a text field page. If the 'yes' choice is selected, the system starts the graphic import process.

As the graphic file import system continues, it asks what name should be on the button that represents the first graphic file on that page. An example of a name for the button might be 'figure 1.1' (see fig. #4.15). When the text says 'see figure 1.1', the user can click on the button with the label 'figure 1.1' and have the graphic
Once a button is created and labeled, the appropriate graphic files need to be connected to the corresponding graphic buttons. This is accomplished by clicking on one of the graphic buttons. The development system asks for what graphic file to use by activating a file browser window (see fig. #4.16). After the graphic file is selected, the graphic file is displayed. If the wrong graphic file has been selected or if the position on the screen where it originally appears is to be changed, the developer can click on the respective graphic file button with the right mouse button and the developer is allowed to make those changes.

Once the developer has the graphic files set up for the given text field page, the graphic file buttons have to be reset to prevent future users from changing which files are represented by which buttons. However, the development tool set allows for any mistakes that the developer may make while importing the different graphic files into the given text field page. The developer continues to have the option of making changes to which file each graphic button is pointing to. The 'set graphic files in stone' button is displayed, on the lower part of the screen. When the 'set graphic files in stone' button is activated it takes away the function to change
location and files for each of the graphic file buttons (see fig. #4.17). It also removes the 'set graphic files in stone' button from the page.

After any graphic has been displayed, the end user or the developer can modify or close the display window. Possible modifications include changing the window location or miniaturizing the window. The location of the window can be changed by clicking on the top bar of the window and moving it to a different location, in the same way as all Windows type windows do. The window may also be miniaturized. When it is miniaturized, the window is closed and an icon representing the graphic file is displayed on the lower part of the screen, in the same way as Windows icons are displayed. It can be redisplayed by clicking on the icon.

The graphic display window can be closed by two different ways. First, if the end user clicks directly on the graphic display itself, it will disappear. Second, if the end user selects the menu bar on the graphic file display window and selects the 'close' option, the display window will disappear.
This paper describes the process of developing some of the necessary tools to build an interactive information system. An interactive information system is a hierarchical menu-driven system that contains and displays information in the form of documents containing text, pictures, graphs, sound, animation, data, and meta-data. The main advantage of a interactive information system is that a user can browse through the information and skip areas that are not of interest.

This research project provides an integrated set of tools for building an interactive information system. The type of interactive information system that will be built as a result of this work is for the distribution of scientific data and information related to that data.

4.7 Testing the Development Tool Set

A portion of the CD-Rom Arctic Data Interactive (ADI) was selected to test the development tool set. The section of Arctic Data Interactive that was chosen was Arctic Research Plan (ARP). This part of the ADI has natural breakdown of information that corresponds nicely to the menu system’s capabilities. The Arctic Research Plan was set up with five topics and several subtopics. This type of information organization is an excellent way to test the building of a menu and capability of the development tool set. Furthermore, this project was inspired by the earlier work done for the ADI, thus it appeared logical to conduct the testing of development tool set on a part of ADI.
4.7.1 Getting the Information out of Arctic Data Interactive

The process to get the data out of ARP was difficult and time consuming, it took over eight hours. Even though ADI allows for parts of the information to be saved to a disk, it is saved in the ToolBook format. The format that is needed to get the text into the development tool set is in text form or ASCII. The best known way to accomplish the conversion from ToolBook to ASCII was to copy each text field to the clipboard in Windows, then go to a Windows word processor and paste what is in the clipboard to the word processor. Then save the text in the word processor in the ASCII format with a '.TXT' extension. This process leaves a file with only text in it. Now, it can be imported into the development tool set.

The pictures and graphs were extracted from ARP using HI-JAAK to do screen captures. Different file formats were selected in order to properly test the file conversion package. This process was completed without any problems.

Within ARP, the information was already organized with a hierarchical menu. The major topic headings formed the top level of the IIS menu system, and sub-topics became the second level of the menu system. The number of topics and the different titles of the topics in ARP were used directly for building the equivalent menu system in the development tool set.

4.7.2 Building the New Arctic Research Plan

The building of the new Arctic Research Plan took less than three hours using the development tool set. There were only minor problems encountered during the building process, which will be discussed further in the following section. A great deal of time was saved in developing the new version of the ARP using the development tool set as compared to programming it from scratch. It took six weeks to program the Arctic Data Interactive (ADI). ARP is about one fourth of ADI, so it can be assumed that it took about 1.5 weeks, or 60 hours. Therefore the three
hours required using the IIS development tool set corresponds to an approximately 20 times saving in time.

The Arctic Research Plan has five general topics with as many as six subtopics under the general topics. There are many pages of text within each subtopic. Within the text there are references to eight different pictures and tables. These pictures and tables were imported using the graphic import system.

Getting the text into the new ARP was generally easy. However it was discovered that problems arose whenever a wrong text file was selected for a given topic. To correct this mistake, many pages had to be deleted from the book and a new title page for that topic had to be created. It would be a nice additional feature to allow the developer to check the text file before it is imported into ToolBook.

The tables and pictures extracted from ARP were saved in a format other than Windows bitmap. This was done in order to make sure that the graphic conversion system would work. There were relatively few graphic files as compared to the number of text pages, so most of the text pages had no graphic buttons on them. Creating the links between the text and graphics worked well. The feature of being able to confirm and if necessary, to change the relationships between the graphic files and graphic buttons was especially useful, since no guessing was required to confirm that the correct graphic file was selected by each button.

4.7.3 Problems Encountered While Testing

Most of the problems encountered involved the extraction of information from the existing ARP. One problem was found which relates directly to the future role of the development tool set. While designing the tool set, the titles for different topics were used in the menu system as names for different objects that form the menu system. ToolBook allows the object names to be up to thirty-two characters long. The development tool set used these topic names to name buttons and pages. Some of the topic ARP headings were more than thirty-two characters long and resulted in a ToolBook error.
This problem was resolved by programming default type names for each object within the tool set other than using topic titles. Topic titles are only used as labels in topic fields and are not used as a part of object names. Topic titles can now be longer than thirty-two characters long without resulting in an error.

During the first test of the development tool set, it became obvious that the development tool set was not user friendly. If a developer made a wrong choice or canceled an operation, the development tool set didn't operate in an expected way. The development tool set would continue operations even though the developer tried to cancel an operation. Also, the computer system would sometimes lock up when a wrong operation was done. These problems were resolved in the latest version of the development tool set. Every time the developer does an operation within the development tool set, the development tool set asks if the information is correct. The developer has a chance to go back and correct errors or cancel the operation. All parts of the development tool set respond in the same way as standard Windows type applications respond.

4.8 On Line Help System

In the development tool set there are buttons that activate the on line help system (see fig. #4.4). The on line help system is made up of six different parts, which are:

1. About the Development Tool Set
2. Before You Start
3. Navigation
4. Building a Menu
5. Importing Text
6. Importing Graphics

The different help buttons in the development tool are context sensitive. The button on the menu page brings up the 'building a menu' part of the help system. On each page of the help system is a text field that contains a description of what
a developer should do (see appendix H). Once in the help system a person can access any part of the help system by using the buttons on the lower part of the page. When done, the 'Click to Dismiss Help' button will return the development tool set to the page where the help system was activated.
Chapter 5: Research Issues and More About the Code

Chapter four provides a description of how to use the development tool set from the point of view of the IIS developer. In this chapter, the code itself will be presented. How the different tools were integrated will be discussed, along with what problems were encountered during the development of the tool set. So that others may better understand the development tool set, a detailed description of the code is included. Appendices A-C provide listings of the code. Comments within these listings are included so further maintenance of the code is possible.

Most of the development tool set was developed in ToolBook. Two of the systems that were developed in ToolBook, were built from code that is available at an anonymous ftp called 'ftp.asymetrix.com'. This ftp site has many different pieces of code for ToolBook developers to use. The text import system and the graphic file import system were built from code found at 'ftp.asymetrix.com'. The rest of the systems were developed from scratch, except the graphic file conversion system, which was a commercial product.

5.1 Detailed Information About ToolBook

ToolBook is an object-oriented interface builder for Windows. It is called object-oriented because it has some of the features that are found in the Object Oriented Paradigm of programming: the use of objects and data encapsulation [Rumbaugh, 1991]. Having the development tool set organized into objects allows for very modularized code. The modularization of the development tool set made modifications to the code easier. There was no worry that changing the code of a system would affect the behavior of other systems.

In ToolBook, the end product is called a book. This book is made up of pages and backgrounds. A background can be shared between more that one page. Both a page and a background can contain objects. An object on a page or a background can be a button, text field or a graphic. All items in ToolBook,
including the book itself, can have a script. The script to these items contains the code and data that the object operates on. All items - book, page, background, button, text field and graphic - are used to build an IIS.

The objects that can be on a page are: a graphic, a text field, or a button. Each of these objects have different properties. For example, a button can be one of several types of buttons, it can have different colors, sizes, location, etc. These properties are in the data encapsulation that is a part of ToolBook.

Several objects on a page or background can be put together to form a group. Within a group, each object can have a script, but also, the group, as a single unit can have its own script. The grouping ability of ToolBook allows many
objects to share the same script.

A background is a part of the book that contains objects to be shared between several pages. The page is made up of a background and a foreground. The transparent part of a page that holds objects specific to that page is called the foreground (see fig. #5.1 and fig #5.2). These figures show that both background and foreground are composed of different layers, where each layer contains one object (a button, a field, a graphic, etc.).

In ToolBook the code is activated by the passing of messages from one object to another. Each object in ToolBook can have a script (programming language in OpenScript) that will perform different operations based on messages
When a message is sent to an object, it travels up what's called the object hierarchy until the message locates a handler for itself or until the message reaches the ToolBook system.

Figure 5.3 ToolBook's Object Hierarchy

that it receives. A script can have several handlers, each handler is for a different message that can be passed to an object. For example, the mouse cursor is considered an object. When a user clicks the mouse this will send the message 'buttonUp' to the object that the mouse is over. The ToolBook system looks for a 'buttonUp' handler within the script of the object that was clicked on. Then the code that is within the 'buttonUp' handler will be executed. Scripts are associated with the book, the pages, the graphics, the text fields, and the buttons.

A message can be passed directly from one object to another or it can be sent up the object hierarchy. What ToolBook calls its object hierarchy is based on the object-oriented concept of aggregation [Rumbaugh, 1991]. When a message is sent up the object hierarchy, the first object that the system encounters with a script that has a handler for the message is the one that gets executed (see fig. #5.3). This figure shows the path that a message would take if sent up the object hierarchy. For example, if an object on a page didn't have a 'buttonUp' handler and a user clicked on that object with the mouse, the message would first go to the object, then to the page, then the background of the page, then the book and finally
the ToolBook system. The message would stop anywhere a proper handler is found during the journey of the message up the object hierarchy. In this way a background can have script routines that can be used by all of its pages; and a page can have script routines that are used by all of the objects on the page.

5.2 Overview of Code Structure

The IIS development tool set provides an integrated set of routines used for building an interactive system for the distribution of scientific research papers and data. A developed end product, or IIS, for any project requires a large amount of ToolBook code. Any project-specific IIS can be written by an experienced ToolBook programmer. However, the development tool set is actually a set of programs that will write the necessary ToolBook code for any IIS application, based on decisions that a developer makes. The development tool set is a collection of routines or systems that utilizes ToolBook’s programming language, OpenScript, to produce project-specific IIS products.

The development tool set develops ToolBook objects reflecting decisions made by the developer. It creates new pages, copies objects on to the pages, and builds the scripts for the objects. The development tool set has two different backgrounds that end up a part of the IIS. One background is for the pages that make up the menu system. The other is for the pages that contain the text. From these two backgrounds new pages are made within the menu system or the text import system. When a new page is made, the development tool set goes to one of three different pages to copy objects and fields that contain pieces of code that can be assembled to make up a script for the objects. In this way, a new page is made, objects are placed on it, and scripts are built for the objects. The application-specific code is built without the developer having to learn the programming language.
5.2.1 Requirements Analysis

Based on the definition of the problem in Chapter 1, a usable and expandable tool set had to be developed. Because the USGS wanted a tool set that was capable of building IISs like the Arctic Data Interactive (ADI) scientific journal, design features were taken from ADI. First, the ADI concept of organizing information in a hierarchy, was used as a model for the menu system in the development tool set. In ADI the information is organized according to topics and sub-topics. Several pages in ADI have names of topics on them and access is available to the topic by clicking on the name with the mouse. A user can leave a topic by clicking on a button which returns them back to a topic page. Second, ADI has a close relationship between the text in the IIS and figures and pictures used in explaining the ideas in the text. Access to both text and graphics would need to be in the finished IIS.

During the analysis phase of the project, several people were interviewed in order to determine what features would be the best to go into the development tool set. These people were multimedia developers, people who may be users of products made with the development tool set, and scientists. A broad base of people were sought for their ideas because of the many types of people who may be involved with the tool set. Potential developers were concerned with easy development features. The user types were concerned that the end product be easy to use.

Of those that were talked to, three people made suggestions that became systems. Gerry Lebing of the USGS in Reston Va., who programmed the original ADI, wanted a way to get text into the IIS easily. Edward McFual of the USGS in Reston Va., had worked on multimedia CD-ROM projects and was interested in tools that import pictures, sounds and animation. Mike Crane is a part of the GIS lab for the USGS; he said the problem that they deal with involves getting graphical representations of a digital map into a document. This problem is due to the different file formats for computer generated pictures. The following was determined
to be a starting point for features that should be in the development tool set:

1. The importing of text
2. The importing of graphics
3. Graphic file conversions
4. A way to access text by subject
5. Search for key words
6. Print out information

Each of these features became a system in the development tool set.

The development tool set was constructed so that the product it produced responds to the user like the Arctic Data Interactive IIS. The development tool set was designed so that it would respond in the same manner that other Windows based applications would. In this way the developer would have a chance to correct errors and confirm operations before they were put into affect. These features help in making the development tool set user friendly.

5.2.2 What the Development Tool Set Looks Like

The development tool set is made up of ten pages. Each of these pages has buttons, fields or graphics that are used to build the IIS. From the first page of the development tool set a user can start building an IIS. Pages two through seven contain the on line help system. The eigth and ninth pages contain objects used by the development tool set to build an IIS. Finally, page ten is a template used for the text field pages.

The first page of the development tool set contains the button that starts the IIS building process (see fig. #5.4). This button takes care of all the preliminary operations before the menu system is activated. The name of this page is 'makeMenu'. Also there is a button that activates the on line help system.

The next six pages are part of the on-line help for the developers. Each of these six pages are the same, except for the text that is in the field, and the title name of the page (see fig #5.5). Figure 5.5 is the first page of the on line help
system and contains an overview of the development tool set.

The eighth page is used for storing buttons and fields used by the text import system and the graphic import system (see fig. #5.6). The developer doesn’t see this page during the development of the IIS. The fields contain text that is used for the scripts of text field pages when graphic files are imported to the text field page. Each of the buttons that are used from this page contain a script. The button that activates the text import system is stored on this page. The scripts for these buttons are not built, because these buttons contain scripts that don’t need to be modified based on decisions that the developer makes.

Page nine is used for storing all the fields, buttons, and graphics necessary for the building of the menu system, and the main button for activating the text import system (see fig. #5.7). Some of the fields contain text that is used for
Building a Menu

Importing Text

Importing Graphics

Figure 5.5 On line help page

building scripts for buttons that are a part of the IIS’s menu system. The background of this page is used by all of the menu pages. The background is named 'theBackground' while the page is named 'makeMenuPage'. The items on this page are copied and put together in order to build a menu page. The developer doesn't see this page during the development of the IIS. Instead the development tool set uses this page to store objects used in the building of the menu system and the button that activates the text import system.

The foreground of page ten contains no objects; however, the background is used by the text field pages. The background contains a field that holds text for the text field pages, some graphics, and a button that contains the code for the text search system. The text import system uses this page as a template for all the text field pages (see fig. #5.8). Other objects are placed on a copy of this page to build
a text field page.

5.2.3 Structure of the Finished IIS

The final IIS is divided into two parts. The first part contains all of the pages that are part of the menu. The second part contains all the pages that hold the text of the IIS. Windows that display the graphic files are created when a user clicks on a graphic button, therefore there are no pages in the ToolBook book itself that contain the graphical pictures. The pages of the IIS are setup physically one after another, just as are the pages of a printed book. The logically setup is different however, reflecting how a user or developer moves around the built IIS.

The title page is the first page of the built menu. The title page contains
several buttons that contain pointers to other menu pages, or to the first page of text of a topic. Menu pages other than the title page contain further subdivisions of the topics into subtopics. Each leaf node of the hierarchical menu contains the first page of text for that topic.

The pages of text within a topic are physically set up in a sequential order. However, the topics may not be in their physical order. The physical order of the topics depends upon the order of importing of the text files. The logical order of the text topics is maintained by the pointers from the pages of the menu.

5.3 Hierarchical Menu System

Since the hierarchical menu system is the tool that other tools are built upon,
it has to be flexible and easy to integrate with other tools. Some development of the other systems had been done before the work on the menu system ever started. Therefore, when the menu system was being developed, plans for how other systems would fit in were made so that integration would be easier.

It was assumed, at first, that an IIS (Interactive Information System) couldn't exist without a menu system. Then the idea was presented of an IIS with only one topic. In this case there wouldn't be a menu system. When entering the IIS, it would put the user on the first text page. With the possibility of having to develop a single topic IIS, the code had to allow for this special case.

In the current version of the tool set, the building of the IIS begins with the activating of the 'Build a Book' button. The script for this button is responsible for getting the name of the IIS and setting up the first page of the IIS (see appendix A
part 1). It asks for the name of the book. This would be the name of the research project or paper that the IIS is based on. ToolBook has a command called 'ask' that gets information from a user (the developer of the IIS in this case) easily. The name of the book is confirmed with the developer to make sure that it is correct. A new page is created and a copy of the menu system button, from the page 'makeMenuPage', is placed on the new page along with the help button and 'theTopic' field. The field 'theTopic' holds the name of the book that was retrieved from the developer earlier.

When the 'Make Menu' button is activated, the system asks if the developer wants to build a menu system (see appendix A part 2). If the developer responds yes, the system then asks for the name of the first topic. A 'No' response gets a copy of the button that activates the text import system and places it on this page. After this the menu system for this topic ends.

If the menu system gets a 'Yes' response it asks for a topic name, the topic name is confirmed, the system then asks for brief description of the topic and the description is confirmed. When all this information is received from the developer the system goes to the page 'makeMenuPage' and gets a copy of three fields containing scripts (see appendix A part 6) that are used for constructing the scripts of the finished IIS, one field that will hold the topic name ('topicField') and the button 'addTopic'. The field 'topicField' is given a default name, the script is built for it (see appendix A part 4), and it is positioned on the page. After this a new page is created that will be the first page of the new topic, a copy of the menu system button is placed on it, and a copy of the button 'goUp' is also placed on it. The script for the 'goUp' button is built from the text in two fields (see appendix A part 5); this button allows a user or developer to return back to the calling menu page.

The 'addTopic' button allows the developer to add more topics to a menu page. If the 'No' option is selected by the developer when activating the script for the button 'addTopic' then this button is removed and no more topics can be added to this menu page. The 'addTopic' button's script is very much like the script of the menu system button, (see appendix A part 3) in that it gets topic information from
the developer, builds a topic field and places it on the menu page, and creates a new page for the topic.

The developer can continue placing topics on a menu page with the 'addTopic' button and is only limited by the space available on the menu page. The developer can continue building the menu system from each new page that was created from a particular level of the menu system. The ability to continue building the menu system with the same code is what makes the menu system recursive.

Most of the problems encountered with development of the menu system happened with the recursive concept of the system. Originally the top level of the menu system was the one that took care of the preliminary operations. The first tool set had two menu systems, one that took care of the top level and one for every level after that. The top level menu system contained routines that got preliminary bookkeeping information from the developer. In order to make the menu system truly recursive, these operation were move out of the menu system and placed in the 'Build a Book' button on the first page of the development tool set. The present tool set has only one menu system.

5.4 The Text Import System

The text import system allows ASCII text files to be easily inserted into the IIS. The system was built from a set of routines that were available at the anonymous ftp site 'ftp.asymetrix.com'. The supplied code had to be modified so that it would integrate with the menu system and the other systems. This system can import into ToolBook any ASCII text file available on the development computer system. It automatically breaks the text file up into pages of fixed size and places the proper buttons on each page to allow the user to navigate through the IIS.

5.4.1 The Asymetrix Text Import System

The text import system from Asymetrix that was integrated into the
development tool set supported three operations. The code from Asymetrix took care of the following complex problems:

1. Opening the correct data link libraries in order to communicate properly with Windows
2. Copying the text into ToolBook fields and breaking up a long text file in many ToolBook pages
3. Placing the proper buttons on the page to allow the user to navigate through all the text

All of the code that is in the Asymetrix text import routine was kept. More code was added to help with integration.

5.4.2 Modifications Made

While the text import routine developed by Asymetrix had all of the features necessary for the text import system, integrating with the rest of the development tool set was required. The following additional features were added to the Asymetrix code:

1. Set up a pointer to point to first page of text
2. Add a button to each page that returns to the calling menu page
3. Add a button to each page that starts the graphic file import system
4. Place a copy of the text search system on the background of each text field page

5.4.3 A Description of the Code for the Text Import System

When a developer clicks on the text import system button, the first thing that this system does is to open some data link libraries (see appendix B part 1). The
data link libraries allow ToolBook to communicate with Windows operations. Two data link libraries are used in the text import system. The first data link library is used to find out what file is to be opened and read in. The other data link library is used to find out how large the file is, so that the system knows when it is at the end of the file. The text import system asks the developer what text file is to be opened using the 'OpenDLG' data link library; this library opens the Windows file browser.

The next thing the text import system does is to create the first page and set a point from the menu to the first text page. The process of setting a pointer to the first text page is complicated. Within the menu system, when a topic field is clicked on to enter that topic, the code jumps to the proper page within the book. This is done with a ToolBook command called 'go to page'. So, in the menu system, when a new page is made for a menu, it is automatically given a name, for example 'page10'. When the lowest level of the menu is reached, the developer is looking at a named menu page and clicks the 'no' response to make another level of menu. The 'no' response automatically activates the text import system. The menu page where the text import system was activated will be used as a temporary "staging area" while the text pages for this topic are made. The menu page will not be needed after the text is imported, because it doesn't contain a real menu; the page will be renamed 'temp' during its use as a staging area, and will be deleted entirely after the text is imported. The name the menu page had been assigned, for example 'page10' will be given to the first text page instead. In this way, the calling menu will correctly point to the first page of text for the given topic.

After the text import system has created a new text field page, it activates the Windows file browser so the developer can select which file is to be opened. After the developer selects a text file, the development tool set enters a loop. In this loop, the text import system first goes to the page where the text import system was activated, which was renamed 'temp'. On the page 'temp' the text import system gets a copy of the title field and the 'goUp' button which is needed to return to the calling menu page. Then the system returns to the text field page being created and places the title field and 'goUp' button text field page. Next, the text import system
gets ASCII text from the text file and places it in the Text field on the text field page until the text field is full. Then, if the page is not the first page of the topic, the system adds the 'previous' page button and if the page is not the last page of the topic, the system adds the 'next' page button. The button for the import graphics system is also copied to each new text page. At the end of the loop, if there is more text left to be imported, the text import system decrements the file size counter, creates a new text page and continues within the loop. After all the text has been imported for this topic and the loop terminates The system then removes the page that was named 'temp' using a routine called 'killPage'.

The 'killPage' routine is told what page to delete and what page to go to afterwards (see appendix E part 1). The 'killPage' routine is in the script on the book level. 'killPage' needs two system variables to be set before it can function properly. These two variables are 'pageToKill' and 'whereToGo'. The routine goes to the page that is to be deleted and selects the entire page. Then it sends a message to the ToolBook system to remove that page. The other thing that this routine does is to go to the page name that is in the variable 'whereToGo'. The reasoning behind placing this routine at the book level is the anticipation that the 'killPage' function will be needed by other systems that are added to the development tool set.

In the first version of the development tool set, there were two text file import systems. One was used when the menu system is built, and the other was used when there is no menu system. These two systems were brought together into one system that handles the special cases by checking to see if there is a menu system built. The two old text import systems were the same except that the button that returns the user up the menu system, 'goUp', was not needed on the non-menu system text pages. The part of the text system that copies the 'goUp' button was put inside a 'if - then' statement that checked to see if a menu system has been built. Having only one text import system saves on the amount of code to be maintained in the future.
5.5 Text Search System

The text search system allows the IIS user to search for key words or phrases that are within the text fields of ToolBook. The user can click on a button set up for doing word searches. This button is located on each of the menu pages and each of the text field pages. When clicked on with the mouse, a window pops up that asks for a word or phrase to search for. It finds the first occurrence of the word or phrase within the text fields. After the word or phrase is found, then the system asks if the user wants to continue searching using the same string. This system assists users to find information quickly and easily.

All of the programming for a text search system has already been done in ToolBook. ToolBook has an OpenScript command called 'search' that searches through all the text fields within a ToolBook book. This command retrieves a string from the user and then search all text fields within the book for the given text string. If the string is not found then the system stops and reports to the user 'String Not Found'. If the system finds the string within the text fields then it asks the user if they want to 'Continue Search?'. If the user answers 'yes', then it goes to the next occurrence of the given string.

With the command 'search' available within ToolBook, the development of the text search system was easy, a one line script (see appendix D). Integration was also easy. During the integration process, all that needed to be done was to copy the button containing the script of text search system. From any menu page or text field page, a user can do a text search of the text field pages.

5.6 Graphic File Conversion System

Scientific visualization software produces a great variety of graphic file formats and there is a need to get these files into an IIS. ToolBook supports the direct import of only a limited number of graphical file formats. Because of the large
number of different graphic file formats, a routine was needed to convert the graphic
files into a type that can be imported into ToolBook.

Most of the work with the development of a graphic file conversion system
had to do with integrating this system into the graphic file import system. The
backbone of the graphic file conversion system is an off-the-shelf product called
Image Alchemy. This product converts numerous different graphic file formats to
other formats.

Image Alchemy is a stand alone commercial product. It can be used by itself,
or as an integrated part of the development tool set. As a stand alone product, the
developer can convert and resize many graphic files. The developer has all the
features that are part of Image Alchemy available to use outside of the development
tool set (see Chapter 3). These features allow the developer to make changes in
the conversion process that may help in producing a more refined graphic picture.
The graphical user interface for Image Alchemy is very user-friendly and takes little
time to learn.

Integration of Image Alchemy into the development tool set was very simple,
since Image Alchemy can be executed from a DOS command line. ToolBook has
a feature that allows it to run other programs from a script by creating a DOS
command line. Within the graphic file import system, the ToolBook script checks to
see if the graphic file is a type that can be imported into ToolBook. If it is, the
graphic file import system continues as normal. However, if the file is one that can
not be directly imported into ToolBook, the system builds a DOS command line that
calls Image Alchemy. Image Alchemy then converts the selected file into a format
that can be imported into ToolBook. Because this is automatically taking place, the
developer doesn’t have to know what has happened. Therefore, it appears to the
developer that any graphic file format can be imported into the IIS. The graphic file
conversion system was integrated into the graphic file import system. A more
detailed description of the graphic file import system and how the graphic file
conversion system was integrated is given in the next section.
5.7 Graphic File Import System

The graphic file import system allows the developer to import into the IIS graphic files that are related to the text. This system allows the developer to connect graphic files on the computer system to buttons on the text field pages of the IIS. The developer has the ability to place buttons on each text page as required to display the contents of desired graphic files.

Most of the graphic file import system was developed by Asymetrix. They had a button that a developer could copy into a book, and the button could be connected to a bitmap in a file external to the ToolBook book (the IIS). If this button wasn't connected to any file, it would ask for a file. The developer could change the file that it points to with the right mouse button. All of the code in the button 'bitln' was developed by Asymetrix.

The major drawback was that this code also would allow the end user to change the file with the right mouse button. The end user should not have the ability to change the IIS. Therefore, the system was developed so that the developer can make sure that the graphic buttons are pointing to the proper graphic files. Then, after all the buttons on a page are pointing to the proper files, the ability to change the name of a file identified by a button must be taken away. A routine was developed called 'Set Graphics in Stone' that took care of this problem.

The Asymetrix code was modified so that a number of graphic buttons can be placed on a given text page. With the basic routines that Asymetrix developed and the modifications that were made, a system was developed that allows the developer to import graphic files into the IIS and not have to do any programming in the OpenScript language.

The graphic file import system is made up of scripts that are in four different buttons. Those buttons are 'graphicln', 'bitln', 'addGraphic', and 'setInStone' (see appendix E parts 1 - 4). There is also some code at the page level that closes bitmap windows when entering or leaving a page. The scripts in 'graphicln', 'setInStone', 'addGraphic', and at the page level are not a part of the original
Asymetrix code. The code at the page level is the handler that takes care of changing which file a graphic button points to.

The 'graphicIn' button contains the script that places the first graphic button on a text page and sets up the script of that page for graphic file importing. The first thing that this script does when activated is to ask the developer if there are to be any graphic files on this page with the 'request' command. The 'request' allows the system to get predetermined responses from a user (for example Yes or No). If the developer answers 'no' then the system removes the 'graphicIn' button. Otherwise, the system sets the script of the page to the text in field 'script6', (see Appendix A part 6). Then the developer is asked what the name of the figure is, for example 'fig. #1.3'. The name for each graphic button shows up in the button's caption.

When a graphic button 'bitln' is clicked on, if it isn't connected to any file then the button sends a message to the page that opens the Windows file browser. The developer can select the proper file. If the file is a bitmap file then the file is displayed in its own window. If the file isn't a bitmap file the file is converted into a bitmap file using the graphic conversion system. When the bitmap is open, the window can be closed by clicking on the actual image with the mouse.

The button 'addGraphic' button allows additional graphic buttons to be placed on the text page. The script of this button is similar to that of 'graphicIn' except it doesn't change the script of the page, because the script of the page is set up for importing of graphics only once. The 'addGraphic' button allows as many graphic buttons to be placed on the text page as the developer wants, the only limit is the amount of space available on the page for graphic buttons.

The 'setInStone' button takes away the ability to change which file the graphic button is point to by setting the script of the page to the text in field 'script7' (see appendix A part 6). The 'setInStone' button also removes itself and the 'addGraphic' button. This is a overview of the code; the following is more detailed description of what the code is doing.

The 'graphicIn' button copies a script, for handling graphic operations, to the text page where the 'graphicIn' button was activated. There are four different
handlers within the script: 'leavePage', 'closeBit', 'enterPage', and 'getBit'. The 'leavePage' handler makes sure that no bitmap is open when leaving the given page and going to another. The main command that accomplishes this is the 'tbkBitmap('close dibFile')'; this is the command that closes a open bitmap file. An open bitmap window that is not related to the new page of text can become confusing for an end user. The 'closeBit' handler is activated when a user clicks on the bitmap with the mouse to close the bitmap window, and when a bitmap is already open and a user tries to open a new one. Because of the way the scripts are set up, two bitmaps can not be open at the same time. It is possible to write code for more that one bitmap to be open at one time, but each graphic button would need different names for the system variables that control the opening and closing of the bitmap window. To do this, would require much programming.

The 'enterPage' handler takes care of a system variable called 'openBit'. This variable has a true/false value and lets other scripts know if there is an open bitmap on that page. The 'enterPage' handler is activated when entering the page, and the value of 'bitOpen' is set to false.

The largest handler in this script is called 'getBit'. This handler connects a graphic button to its associated graphic file. This handler does two main things: gets a name of a bitmap, and it defines the location on the screen where the developer wants this bitmap to be placed when it is opened. The 'set graphics in stone' button removes this handler, so a end user doesn't change which file a graphic button is point to.

5.8 The Text Print System

The text print system allows the text in the text field pages to be printed. With the routines available in ToolBook's OpenScript the script to this system is only two lines long (see appendix G). These two commands are: 'set printerfields' and 'send printReport'.

The integration of the Print Text System was done by placing the button on
the background of the text field pages. This way, the system can be activated from each text field page.

5.9 Other Scripts

Two other scripts that are part of the development tool set are the 'PrevPage' button, and the 'NextPage' button. The buttons 'PrevPage' and 'NextPage' are found on the text field pages and are used to flip pages, as in a book. These very short scripts send messages 'next' and 'previous' to the ToolBook system, which bring up the next or previous pages respectively. These buttons were part of the original text import system from Asymetrix.

5.10 The Help System

Within the help system there is three parts: buttons located throughout the development tool set that activate the help system, buttons in the help system that move a developer to different topics, and a button in the help system that returns the developer to the page where the help system was activated (see appendix I). The buttons that activate the help system do two things: go to the proper section of the help system and set a variable that tells the help system where to return to. Buttons that activate the help system are context sensitive. For example, when a developer clicks on a help system button on the page where the text import system button is, the help system goes to its text import section. Once a developer is in the help system they can move to other topics by using the topic buttons. The 'Click To Dismiss Help' button returns the developer back to page of the IIS where the help system was activated.

5.10 Summary

Before development was started on the different systems of the development
tool set, many people were interviewed to discover which features would be desirable. Both developers, as well as end users, were interviewed. The suggestions given formed the basis of the development tool set design.

As was expected, the learning curve for the programming language OpenScript was very steep. However after the major points of the language were understood, programming became easier.

Of all the problems encountered, the biggest was with the development of the menu system. In order to allow the developer to build a menu system for any combinations of topics and subtopics, it had to be recursive. The two menu systems were put together into one system and the preliminary book keeping operations were moved out of the menu system.

There were many routines that were already written which made development of some of the systems in the development tool set go faster. Modifications had to be made to them in order to give them the desired functionality and to integrate them into the overall development tool set.
Chapter 6 Areas of Continuing Work and Conclusions

The original goal of this project was to discover if the technology was available for developing a set of tools that could be used to build an IIS. Assuming the technology exists, a second goal was to determine what features would be desirable to make a usable IIS tool set for a developer. These tools should be easy to use for both the developer and end user. The tool set that is presented in this thesis represents the basis for such a tool system.

6.1 Overview

This research demonstrated that there is technology available to develop a set of tools for the construction of Interactive Information Systems. A choice had to be made about what products to integrate into a desirable end product. The decision was made to develop several tool sets within ToolBook and to use the Image Alchemy product for graphics file conversions. The systems that were developed by this research are:

1. Hierarchical Menu System
2. Text File Import System
3. Text Search System
4. Graphics Conversion System
5. Graphics File Import System
6. Text Print System

Different people doing IIS development have different preferences as to the features that should be in a development tool set. The biggest demand was for a system that is easy to use. The integration of all the different products into ToolBook allows the features of the development tool set to be used easily. Another benefit of integration within ToolBook is that the different systems appear to be one
system to the developer. The development tool set is seen as consistent across the
different systems. Once a developer gets comfortable with one part of the
development tool set, then other parts of it also become easy to use.

The initial IIS tool set developed during this project represents a good
beginning, but there is more work that needs to be done. Several features can be
added to the development tool set to improve its overall useability. This chapter
presents some ideas concerning continuing work and project conclusions. The
following sections describe some of the more desirable features to be added to the
development tool set.

6.2 What can be Changed

This tool set is just a beginning of a growing set of tools. The present tool set
was build from a functional point of view and not from an artistic style point of view.
Some one may find the background images of the present development tool set less
that appealing. The background images can be changed. A developer can enter
and edit the background images using ToolBook Author mode. This provides a
paint brush style editor where a person can draw whatever they want. The color
and texture of most objects can be changed within the author mode, also. A
developer can change, edit, update, or screw up this development tool set in any
way that they feel fit. The development tool set was intended to have features
added to it, so flexibility is built in.

6.3 Hot Words

Presently the development tool set activates the displaying of figures related
to the text through buttons on the page. A nice feature would be to have these
figures displayed by hot words in the text. In ToolBook hot words are objects and
can have a script, like all other objects in ToolBook. Therefore, the ability to have
figures displayed by clicking on a hot word as compared to a button can be done.
Hot words can also be used to build links between related topics. Using this feature of hot words allows a user to further wander through the text of an IIS interactively. This is an additional way to access information in the related topics within an IIS.

6.4 Print Out Graphic Files

An important feature that will improve usability of the development tool set is the ability for users to print out graphics files. The text print system allows a user to produce a 'hard copy' of the text in the IIS. However, the text is only one part of the information, pictures help in describing information. The ability to print out graphic files will augment the hard copy of text in the IIS.

6.5 Import of Sound Files

Adding a system to the development tool set that would allow the playing of sound files is not necessary for the distribution of scientific data, but it may be a useful feature for some scientific data sets. A large variety of different sounds could be helpful in describing something. For example, playing the sounds that some animal makes may help describe animal behavior or the activities of animals that living in an area. Adding a system to import sound files to the IIS development tools is not a high priority. However, some developers may find a need for it.

6.6 Import Animation Files

Scientific visualization continues to expand in the areas of three-dimensional visualization and animation. There is an important need for a system within the development tool set to import animation files. With an animation file import system, a developer will be able to display information about data in a way that an end user could easily understand. For example, animation would be helpful in showing the
variations in snow or ice coverage of a given area over an annual cycle.

6.7 Routines for the Extraction of the Data from Disk

The type of IIS that is to be developed ultimately by this development tool set will contain not only the results of research and descriptions of the data, but also the actual data. Therefore, it is necessary to have a set of routines that will extract the data from the CD-ROM and copy the data to a disk file. The user can then import that file into other software products in order to further analyze the data. The Data Extraction System will accomplish this.

A large research project may easily generate more than 500 megabytes of data. Many researches may require only a subset of the data. A set of routines is needed to subset the data, so that a researcher using the IIS can retrieve only the required data. These routines subset the data by geographical area or retrieve only those values that fall into a given range. These routines will be part of the Data Extraction System.

The National Oceanic and Atmospheric Administration (NOAA) has developed a data extraction system [David Clark NOAA/NESDIS NGDC Boulder, CO]. They have a set of standard routines that are available to the public which will accomplish the necessary functions of the Data Extraction System. Some additional work will be required to interface this set of routines with ToolBook, but this work appears to be minimal. ToolBook's ability to run other software from a DOS command line will facilitate the integration of this software into the development tool set.

6.8 Update an Existing IIS

After an IIS has been built with the development tool set and distributed to users, there may be the need to put additional information into the IIS. The ability to take a existing IIS and add to it, or modify it would make the development tool set
more flexible for developers. This ability would extend the life of an IIS as more research is done in the area that the IIS is based on.

6.9 Remove Development Tool Set Pages From IIS

In the present development tool set, the pages where objects are stored that the development tool set uses to build an IIS are not removed from the IIS. These pages could take up valuable disk space and computer memory. A developer can remove these pages by going to the author mode in ToolBook. It would be nice to remove these pages with a routine to prevent a developer from removing the wrong pages.

6.10 Conclusions

The original objective for this project was to find out if available technology within the PC environment can produce a set of tools for the construction of an IIS. This research project found that the technology is available for the development of a set of IIS tools within the PC environment. The IIS tools that were developed in this thesis project were integrated to form a single development package. Although the present set of tools is only a prototype, the features in the current IIS development tool set will allow the developer to produce an IIS that is easy to use and appealing to the user. This set of tools will help keep the development cost of IIS low, thereby providing products to users more quickly.

The literature search for this research suggested that no one else appears to be doing the same type of work. There were descriptions of many multimedia products, low level development packages, and architectures of different products. However, no reports were discovered which describe the development of products by taking commercially available products and building a set of tools to make development easier. It may be that tool sets are being constructed as various applications are being developed. However, these tool sets are not distributed
because Multimedia and IIS developers are making money by applying these tool sets. Individual developers have, over time, developed sets of standard routines that allow them to do the development quickly and do not want other people to have the tool set.

This research makes available a set of public-domain IIS development tools which may be used by multimedia developers without programming experience. It is believed that this tool set will help make research information more readily available to the public. In this growing information age, it is important to put information more in a form that is conveniently accessible to a broad range of people. This tool set helps scientists to develop easy-to-use information systems and is a step in the right direction for easy-to-use development packages.
REFERENCES CITED


Macromedia Inc. 1992 Authorware Professional (software) copyright 1992, Macromedia Inc., 6000 townsend St., San Francisco, CA 94103


USGS (United States Geological Survey) 1991. Arctic Data InterActive (software) A Prototype CD ROM Science Journal


APPENDIX A

Part 1 – Code for top level of the menu system

-- button 'BuildBook'
-- Script for button that does initial setup to start building the book
-- It is on the first page of the book
-- All of this script was programmed by larry e. johnson

to handle buttonUp
  system startTheMenu
  system presentWorkingPage
  system topicTemp, topicButton
  set topicButton to null
  if syserror is -1
    break to system
  end

  set syslockscreen to true
  set atopic to null
  set ifOk to false
  while not ifOk
    ask "What is the name of this book?" with atopic
    set atopic to it
    if sysError = "cancel" then
      break to system
    end
    request "Is " & atopic & " the name you want for this book?"\ with "Yes" or "No" or "Cancel"
    if it = "Yes" then
      set ifOk to true
    end
    if it = "Cancel"
      break to system
    end
  end

  set the caption of this book to atopic
  go to last page of background "theBackground"
  send newPage
  get the topicNumber of this book
  if it = null then
    set the topicNumber of this book to 1
  end
  set thisTopicName to "topic" & topicNumber of this book
  set the topicNumber of this book to the topicNumber of this book + 1
  set the name of this page to thisTopicName
  set the firstPage of this book to thisTopicName
  go to page "makeMenuPage"
  select field "theTopic"
  extend select button "menuButton"
extend select group "makeMenuHelp"
send copy
send back -- last page
set focus to null
send paste
set text of field "theTopic" to atopic
set fontSize of text of field "theTopic" of this page to 24

Part 2 — Code for menu building button

-- Script for recursive menu system builder
-- This button is placed on the menu pages
-- this script was programmed by larry e. johnson

to handle buttonUp
    system startTheMenu
    system presentWorkingPage
    system topicTemp
    if syserror is -1
        break to system
    end

    set syslockscreen to true
    request "Do you wish to build a menu system? 'No' will get things"
        && "ready for importing of text for this topic." with "Yes" or
        "No" or "Cancel"
    if it = "Cancel" then
        break to system
    end
    if it = "Yes" then
        get the topicNumber of this page
        if it = null then
            set the topicNumber of this page to 0
        end
        set topicTemp to null
        do
            ask "What is the topic name of section" && the topicNumber
                of this page + 1 & "?" with topicTemp
            if sysError = "cancel" then
                break to system
            end
            set topicTemp to it
            request "Is " && topicTemp && " the correct name for this"
                && "topic?" with "Yes" or "No" or "Cancel"
            if it = "Cancel" then
                break to system
            end
        while topicTemp is not null
end
until it = "Yes"

set presentWorkingPage to name of this page.
set description to null

do
ask "Give a brief description of topic" && topicTemp & "." with description
if sysError = "Cancel"
    break to system
end
set description to it
request "Is " & description & " the proper description" && "for the topic" && topicTemp && "?" with "Yes" or "No" or "Cancel"
if it = "Cancel" then
    break to system
end
until it = "Yes"

go to page "makeMenuPage"
select field "script1"
extend select field "script2"
extend select field "script3"
extend select field "topicField"
extend select button "addTopic"
send copy
send back
set focus to null
send paste
set the script of this page to the text of field "script3"
get the topicNumber of this book
if it = null then
    set the topicNumber of this book to 1
end
set thisTopicName to "topic" & topicNumber of this book
set the name of field "topicField" to thisTopicName
set the text of field thisTopicName to topicTemp
set the topicNumber of this book to the topicNumber of this book + 1
set the script of field thisTopicName to the text of field script1 \ & description & the text of field script2
move field thisTopicName to 2500, (450 * topicNumber of this page)\ + 1000
set the topicNumber of this page to the topicNumber of this\ page + 1
go to last page of background "theBackground"
send newPage

set the name of this page to thisTopicName
go to page "makeMenuPage"
select field "theTopic"
extend select field "script4"
extend select field "script5"
extend select button "menuButton"
extend select group "makeMenuHelp"
extend select group "goUp"
send copy
send back -- last page
set focus to null
send paste

set text of field "theTopic" to topicTemp
set the script of group "goUp" to the text of field "script4"
   ;& presentWorkingPage & the text of field "script5"
clear field "script4" -- clean up
clear field "script5"
go to page presentWorkingPage

clear field "script1" -- clean up
clear field "script2"
clear field "script3"
clear button "menuButton"

else
   -- copy in text import system
   go to page "makeMenuPage"
   select button "ImportButton"
   extend select group "importTextHelp"
send copy
   send back
   set focus to null
   send paste
clear button "menuButton"
clear group "makeMenuHelp"
end:

end:

Part 3- Code for the button 'addTopic'

-- Script for recursive menu system builder second level on
-- This button is placed on the pages that the first level point to
-- and every other menu page from that point on

to handle buttonUp
   system startTheMenu
   system presentWorkingPage
   system topicTemp
   if syserror is -1
break to system
end

set syslockscreen to true

request "Do you wish to add a topic? 'No' will remove the 'Add A " & "Topic' button." with "Yes" or "No" or "Cancel"
if it = "Cancel" then
  break to system
end
if it = "Yes" then
  set topicTemp to null
  get the topicNumber of this page
  if it = null then
    set the topicNumber of this page to 0
  end

  do
    ask "What is the topic name of section" && the topicNumber of this page + 1 &"?" with topicTemp
    if sysError = "cancel" then
      break to system
    end
    set topicTemp to it
    request "Is " & topicTemp & " the correct name for this" & & "topic?" with "Yes" or "No" or "Cancel"
    if it = "Cancel" then
      break to system
    end
  until it = "Yes"

  set presentWorkingPage to name of this page
  set description to null

  do
    ask "Give a brief description of topic" && topicTemp & "." with description
    if sysError = "cancel"
      break to system
    end
    set description to it
    request "Is " & description & " the proper description" & & "for the topic" & topicTemp & "?" with "Yes" or "No" or "Cancel"
    if it = "Cancel" then
      break to system
    end
  until it = "Yes"

  go to page "makeMenuPage"
  select field "script1"
extend select field "script2"
extend select field "topicField"
send copy
send back
set focus to null
send paste

get the topicNumber of this book
if it = null then
    set the topicNumber of this book to 1
end
set thisTopicName to "topic" & topicNumber of this book
set the name of field "topicField" to thisTopicName
set the text of field thisTopicName to topicTemp
set the topicNumber of this book to the topicNumber of this book + 1
set the script of field thisTopicName to the text of field script1 \ 
    & description & the text of field script2

move field thisTopicName to 2500, (450 * topicNumber of this page)\ 
    + 1000
set the topicNumber of this page to the topicNumber of this\ 
    page + 1
go to last page of background "theBackground"
send newPage

set Name of this page to thisTopicName

go to page "makeMenuPage"
select field "theTopic"
extend select field "script4"
extend select field "script5"
extend select button "menuButton"
extend select group "makeMenuHelp"
extend select group "goUp"
send copy
send back -- last page
set focus to null
send paste

set text of field "theTopic" to topicTemp
set the script of group "goUp" to the text of field "script4"\ 
    & presentWorkingPage & the text of field "script5"
clear field "script4" -- clean up
clear field "script5"
go to page presentWorkingPage

clear field "script1" -- clean up
clear field "script2"

else
    -- remove the addToipic button
    request "Are you sure that there are to be no more topics on"
Part 4 – Example code of topic button

-- Script of field on a menu page that controls what topic to go to
-- the code that generated this code was programmed by larry e. johnson

to handle buttonUp
    set syscursor to 4
    set tempPage to name of self
    hide field "topicDescription" of the background "theBackground"
    set my fillColor to my fColor
    fxZoom to page tempPage
    set syscursor to default
end

to handle mouseEnter
    set sysCursor to 37
    set my fColor to fillColor of self
    set fillColor of self to "120,50,100"
    set text of field "topicDescription" of this background to "An over view of the Hydrochemical Data Base of the Death Valley Region."
    show field "topicDescription" of this background
end

to handle mouseLeave
    set fillColor of self to my fColor
    set sysCursor to default
    hide field "topicDescription" of the background "theBackground"
end

Part 5 – Example code of button goUp

-- Script for go up menu button
-- Is placed on each menu page and text field pages
-- the code that generated this code was programmed by larry e. johnson

to handle buttonUp
system topicButton
set topicButton to null
go to page "Test"
end

to handle buttonDown
    send buttonDown to button "thisButton"
end

Part 6 – Text that is in the fields used for building scripts

-- Text that is in field script1
-- Script of button on a menu page that controls what topic to go to
to handle buttonUp
    system topicButton
    set fillColor of self to 120,50,100
    if not (topicButton = null) and not (topicButton = name of self) then
        set fillColor of button topicButton to 180,37.625,100
    end if
    set topicButton to name of self.
    forward
end
to handle gotoTopic
    system isASingleClick, topicButton
    set isASingleClick to false
    set syscursor to 4
    set temp to "checkMark" & topicButton
    show group temp
    set tempPage to topicButton
    set topicButton to null
    fxZoom to page tempPage
    set syscursor to default
end
to handle showTopicDescription
    set text of field "topicDescription" of this background to "
end

-- Text that is in the field script2

" show field "topicDescription" of this background
end
to handle invertSelf
    system topicButton
    set fillColor of button topicButton to 180.37.625.100
end

-- Text that is in field script3

-- Script for page on the menu system traps single and double clicks

to handle enterPage
    system isASingleClick, topicButton, buttonColor
    if not (topicButton = null) then
        set fillColor of button topicButton to 180.37.625.100
    end
    set isASingleClick to true
    hide field "topicDescription" of this background
end

to handle buttonDoubleClick
    system topicButton
    send gotoTopic to button topicButton
end

to handle buttonUp
    system isASingleClick, topicButton
    if isASingleClick and not (topicButton = null) and not (the name of this page = topicButton) then
        send showTopicDescription to button topicButton
    end
end

-- Text that is in the field script4

-- Script for go up menu button
-- Is placed on each menu page and text field pages

to handle buttonUp
    system topicButton
    set topicButton to null
    go to page *
-- Text that is in the field script5

end

to handle buttonDown
   send buttonDown to button "thisButton"
end
Code and related items for the text import system

Part 1  --  Code for text import system

-- Text import system for use with menu system
-- Button is placed on each leaf page of the menu system tree

--- Button Script to load a file into a record field using as many
--- pages as necessary

--- I tried to do things in stages to avoid any one long wait.
--- Once I have the filename I create the new page and set the
--- title, show that much to the user, then load the file.

-- most of this script was programmed by Asymetrix Corporation, it is noted otherwise

to handle buttonUp

-- Variables for max text in a single record field

  local IMaxSize
  local IBiteSize
  local presentWorkingTopic -- name of topic for naming pages
  system pageToKill, whereToGo
  set IMaxSize to 10000 -- Maximum number of characters in any field
  set IBiteSize to 5000 -- Size of chunk to read in (must be a factor of IMaxSize!)

-- Link the dlls

  linkDII "TBKDLG.DLL"
    STRING openDlg(STRING,STRING,STRING,STRING)
  end linkDLL

  linkDII "TBKFILE.DLL"
    LONG GetFileSize(STRING)
  end linkDLL

-- Get the filename to open

clear syserror

  get OpenDLG(".*\.*\.*.txt","Choose a text file for this topic.*","Import")
set IFileName to uppercase(it) -- upper for aesthetics only

if syserror is -1
    break to system
end

set syslockscreen to true

set presentWorkingTopic to the name of this page -- this section by larry e. johnson
set the name of this page to "temp"
go to last page of background "Procedures"
send newPage
set the name of this page to presentWorkingTopic

-- Import the file

-- check the size:
set IFileSize to getfilesize(IFileName)
request "File" && IFileName && "is" && IFileSize && "bytes."

-- Create a first page for the file
set syscursor to 4 -- sand glass

set IPageNum to 1 -- initialize the counter

-- Open the file:
clear syserror
openfile IFileName
if sysError is not null
    request "Error opening file:" && syserror
    break to system
end

-- count the filesize down to 0:
while IFileSize > 0:

go to page "temp" -- this section by larry e. johnson
select group "goUp" -- button to go up the menu system
extend select field "theTopic"
send copy
send back -- last page
set focus to null
send paste
get charcount(text of record field "LoadField")
while (it < IMaxSize) and (it < IFileSize)
    Clear it
    readfile IFileName for IBiteSize
    put it after text of record field "LoadField"
    clear it
    get charcount(text of record field "LoadField")
end while

decrement IFileSize by charcount(text of record field "LoadField")

if IPageNum is not 1 -- Add the "Prev" button if not first page for this file
    go to page "Objects"
    select button "PrevPage"
    send copy
    send back -- last page
    set focus to null
    send paste
end if

-- Add the "Graphics import" button: -- this section by larry e. johnson
    go to page "Objects"
    select button "graphicIn"
    send copy
    send back -- last page
    set focus to null
    send paste

if IFileSize > 0 -- Meaning there is still text left in the file
    -- Add the "Next" button:
    go to page "Objects"
    select button "NextPage"
    send copy
    send back -- last page
    set focus to null
    send paste

    -- flash the page so the user knows something's happening
    set syslockscreent to false
    set syslockscreent to true

    -- Create and go to the new page:
    send newPage
end if

increment IPageNum

end while

set syscursor to 1 -- restore the arrow cursor

closefile IFileName
set pageToKill to "temp" -- this section by larry e. johnson
set whereToGo to presentWorkingTopic
send killPage -- clean up

end buttonUp
APPENDIX C

Code for the graphics import system

Part 1 — Code for import graphics button

-- Script for import graphics file button on text page
-- Puts 1 to 10 buttons on the page to connect to graphics files
-- this script was programmed by larry e. johnson

to handle ButtonUp

request "Do you wish to import graphic files?" with "Yes" or "No"
if it = "Yes" then
    do
        ask "How Many Graphic Files? 1,2,3,4,5,6,7,8,9,10"
        until "1,2,3,4,5,6,7,8,9,10" contains it
        set num to it
        set count to 0
        set syslockscreen to true
        go to page "Objects"
        select field "script6"
        send copy
        send back
        set focus to null
        send paste
        set the script of this page to the text of field "script6"
        clear field "script6"
        set place to 0
        do
            set count to count + 1
            go to page "Objects"
            select button "bitln"
            send copy
            send back
            set focus to null
            send paste
            ask "What is the name of graphic" && count && "?"
            set bitName to it
            set the caption of button "bitln" to bitName
            set the name of button "bitln" to bitName
            set atemp to topicTemp
move button bitName to 7190, place + 925
set place to place + 380
until count >= num
go to page "Objects"
select button "setlnStone" -- for removing the ability to change graphic files
send copy
send back -- last page
set focus to null
send paste
end

clear button "graphicln" -- clean up
end

Part 2 -- Code for the graphics display button

-- Script for button that controls the import and opening of graphic files
-- This button is placed on text field pages
-- this script was programmed by Asymetrix Corporation, unless noted otherwise

to buttonUp
    system s_bmpHandle, openBit, bitNumber
    local temp, temp2

    get bitNumber of this book
    if it is null
        set bitNumber of this book to 1
    end

    get my fileName -- file name that button is pointing to
    if it is null
        send rightButtonup
    end
    if my fileName contains ".bmp" then

        if openBit then
            send closeBit to this page
        end
        set openBit to true

        -- open this button's bitmap with a pop up window
        set sysCursor to 4
        get tbkBitmapchk("open" && my fileName && "alias dibFile style overlapped parent" && sysWindowHandle, 1, 1)
        if my x is not null
get tbkBitmapchk("window dibFile position" && my x && "," && my y,1)
end
get tbkBitmapchk("window dibFile state show",1)

--set the bitmap's window handle and start translating window messages
set s_bmpHandle to tbkBitmap("status dibFile window")
translatewindowmessage for s_bmpHandle
before 2 send bmpWindowClose to self --destroy window message
after 514 send bmpButtonDown to self --buttonUp message
end
set sysCursor to 1
else
-- do a graphic file conversion using Image Alchemy
-- place converted file into 'iis\graphic\fig???.bmp
-- this section was programmed by larry e. johnson
get my fileName
set temp2 to it
set temp to "alchemy\alchemy.exe" && temp2 && "iis\graphic\fig" &
bitNumber of this book & ".bmp" && "-w"
request "Converting Graphic File." with "OK"
run temp
set my fileName to "iis\graphic\fig" & bitNumber of this book & ".bmp"
set bitNumber of this book to bitNumber of this book + 1
end
end

--dismiss the bitmap when they click it
to handle BMPbuttonUp
system s_bmpHandle

--we use a system variable for the bitmap handle here since a timer
--notification handler might send this message, and the timer notification
--handler doesn't have access to the normal window handle container
untranslateAllWindowMessages for s_bmpHandle
get tbkBitmap("close dibFile")
end

--start a timer when they close the window directly (Alt-f4)
to handle bmpWindowClose

--this lets window finish closing before sending a timer message
--which in turn sends a bmpButtonDown message to close the file
--and untranslate all windows messages
get tbkTimerStart("single",500,100,self)
end

to handle tbkMMTimer
send bmpButtonDown
end
to handle rightButtonUp
    system buttonName
    set buttonName to name of self
    send getBit to this page
end

Part 3 — Code for button set graphics in stone

-- Script for button set graphics in stone on text page
-- Takes away the ability to change graphic files
-- this script was programmed by larry e. johnson

to handle buttonUp
    set syslockscreen to true
    go to page "Objects"
    select field "script7"
    send copy
    send back
    set focus to null
    send paste
    set the script of this page to the text of field "script7"
    clear field "script7"
    clear button "setlnStone"
end

Part 4 — Text of fields used for the building of scripts

-- Text to the field script6

-- Script for page that includes changing of bit maps
-- What to do when leaving page
to handle leavePage
    system openBit
    --close bitmap that might be open
    get tbkBmpMap("close dibFile")
    set openBit to false
    forward
end

-- how to close a bit map
to handle closeBit
system openBit
if openBit then
    get tbkBitmap("close dibFile")
    set openBit to false
end
end

-- what to do when entering the page

to handle enterPage
    system openBit
    set openBit to false
end

-- handler for changing a bitmap

to handle getBit
    system buttonText
    get OpenDlg("*.", "Choose a bitmap file for this button. (.dib or .bmp
extensions)", "Choose Bitmap File")
    if it is null
        clear fileName of button buttonText
        clear x of button buttonText
        clear y of button buttonText
        break to system
    else
        set fileName of button buttonText to it
        clear sysError
        if x of button buttonText is not null
            get x of button buttonText & "," & y of button buttonText
        else
            get "0,0"
        end
        ask "Would you like to specify the position of the window? If so, please enter 2 numeric coordinates." with it
        conditions
        when sysError is "cancel" or it is null
            clear x of button buttonText
            clear y of button buttonText
            break
        when itemCount(it) <> 2
            request "Invalid number of coordinates."
            break
        end
        step i from 1 to 2
        if item i of it is not "0" and not isNumber(item i of it)
            request "Coordinate number " && i && " is not a number."
            break
        end
        set x of button buttonText to item 1 of it
        set y of button buttonText to item 2 of it
-- Text to the field script

-- Script for page that doesn't allow for changing of bit maps
-- Is the script of the page after set graphics in stone

to handle leavePage
    system openBit
    -- close bitmap that might be open
    get tbkBitmap("close dibFile")
    set openBit to false
    forward
end

-- how to close a bit map

to handle closeBit
    system openBit
    if openBit then
        get tbkBitmap("close dibFile")
        set openBit to false
    end
end

-- What to do when entering the page

to handle enterPage
    system openBit
    set openBit to false
end

-- needs to be here for the right button command from graphics button
-- does nothing

to handle getBit
end
Code for the text search system

Script for doing Word searches of the text fields

This button is on text field pages and menu pages

This script was programmed by Larry E. Johnson

to handle buttonUp
    send search
end
APPENDIX E

— Graphic file Import system

— Part 1

—'graphicln'
- Script for import graphics file button on text page
- Puts the first button on the page to connect to graphics files
- this code was programmed by larry e. johnson

to handle buttonUp

request "Do you wish to import graphic files?" with "Yes" or "No"
or "Cancel"
if it = "Cancel"
    break to system
end
if it = "Yes" then

    if graphicNum of this page is null
        set graphicNum of this page to 0
    end
    set bitName to null
    do
        ask "What is the name of graphic" && graphicNum of this page + 1
            &"?" with bitName
        if sysError = "cancel"
            break to system
        end
        set bitName to it
        request "Is" && bitName && "the name for graphics button" && graphicNum of this page + 1
            &"?" with "Yes" or "No" or "Cancel"
        if it = "Cancel"
            break to system
        end
    until it = "Yes"
    set syslockscren to true
    go to page "Objects"
    select button "bitln"
    extend select field "scripts"
    send copy
    send back — last page
    set focus to null
    send paste
    set the caption of button "bitln" to bitName
    set the name of button "bitln" to bitName
    set the script of this page to the text of field "script6"
    clear field "script6"
    move button bitName to 7190 , graphicNum of this page * 380 + 925
- Part 2

-- 'bitln'
-- Script for button that controls the import and opening of graphic files
-- This button is placed on text field pages
-- this code was written by asymetrix

to handle buttonUp
    system s_bmpHandle, openBit, bitNumber
    local temp, temp2

    get bitNumber of this book
    if it is null
        set bitNumber of this book to 1
    end

    get my fileName --file name that button is pointing to
    if it is null
        send rightButtonup
    end

    if my fileName contains ".bmp" then
        if openBit then
            send closeBit to this page
        end
        set openBit to true

        --open this button's bitmap with a pop up window
        set sysCursor to 4
        get tbkBitmapchkfopen" && my fileName && "alias dibFile style overlapped
        parent" && sysWindowHandle, 1,1)
        if my x is not null
            get tbkBitmapchkfwindow dibFile position" && my x & "," & my y,1)
        end
get tbkBitmapchk("window dibFile state show",1)

--set the bitmap's window handle and start translating window messages
set s_bmpHandle to tbkBitmap("status dibFile window")
translatewindowmessage for s_bmpHandle
    before 2 send bmpWindowClose to self --destroy window message
    after 514 send bmpButtonup to self --buttonUp message
end
set sysCursor to 1
else
    -- do a graphic file conversion using Image Alchemy
    -- place converted file into \\is\graphic\fig????.bmp
get my fileName
set temp2 to it
set temp to "alchemy\alchemy.exe" && temp2 && "\is\graphic\fig" && bitNumber
of this book & "\bmp" && "-w"
request "Converting Graphic File." with *OK*
run temp
set my fileName to "\is\graphic\fig" & bitNumber of this book & ".bmp"
set bitNumber of this book to bitNumber of this book + 1
end

--dismiss the bitmap when they click it
to handle BMPbuttonUp
    system s_bmpHandle
    --we use a system variable for the bitmap handle here since a timer
    --notification handler might send this message, and the timer notification
    --handler doesn't have access to the normal window handle container
    untranslateAllWindowMessages for s_bmpHandle
    get tbkBitmap("close dibFile")
end

--start a timer when they close the window directly (Alt-f4)
to handle bmpWindowClose
    --this lets window finish closing before sending a timer message
    --which in turn sends a bmpButtonDown message to close the file
    --and untranslate all windows messages
    get tbkTimerStart("single",500,100,self)
end

to handle tbkMMTimer
    send bmpButtonup
end

to handle rightButtonUp
    system buttonName
    set buttonName to name of self
    send getBit to this page
end
- Part 3

-- 'addGraphic'
-- Script for import graphics file button on text page
-- Puts additional buttons on the page to connect to graphics files
-- this code was written by larry e. johnson

to handle buttonUp

    request "Do you wish to import a graphic file?" with "Yes" or "No" or "Cancel"
    if it = "Cancel"
        break to system
    end
    if it = "Yes" then
        if graphicNum of this page is null
            set graphicNum of this page to 0
        end
        set syslockscreen to true
        set bitName to null
        do
            ask "What is the name of graphic" && count &"?" with bitName
            if sysError = "cancel"
                break to system
            end
            set bitName to it
            request "Is" && bitName && "the name for graphic" && "button" && count & "?" with "Yes" or "No" or "Cancel"
            if it = "Cancel"
                break to system
            end
        until it = "Yes"
        go to page "Objects"
        select button "bitln"
        send copy
        send back -- last page
        set focus to null
        send paste
        set the caption of button "bitln" to bitName
        set the name of button "bitln" to bitName
        set atemp to topicTemp
        move button bitName to 7190 , graphicNum of this page * 380 + 925
        set graphicNum of this page to graphicNum of this page + 1
    end
- Part 4

-- 'setlnStone'
-- Script for button set graphics in stone on text page
-- Takes away the ability to change graphic file pointers
-- this code was written by larry e. johnson

to handle buttonUp
  request "Are you sure that all graphic buttons are set to the"
     &&"correct graphic files" with "Yes" or "No" or "Cancel"
  if it = "Yes" then
    set syslockscreen to true
    go to page "Objects"
    select field "script7"
    send copy
    send back
    set focus to null
    send paste
    set the script of this page to the text of field "script7"
    clear field "script7"
    clear button "setlnStone"
    clear button "addGraphic"
  end
end
Appendix F

- Other Code

-- Script to the book
-- this script was programmed by larry e. johnson

-- script for removing a page in the book

to handle killPage
    system pageToKill, whereToGo
    go to page pageToKill
    select this page
    send clear
    go to page whereToGo
end

-- script that is activated when entering the book

to handle enterBook
    send reader
    if the caption of this book is not null then
        go to the page (firstPage of this book)
    end
end
Appendix G

-- 'printText'
-- Print Text Button on text field background
-- Sends the text that is in the text field to the printer
-- this code was written by larry e. johnson

    to handle buttonUp

        set printerfields to "LoadField"
        send printReport

    end
Text in the on line help system

About the Development Tool Set

The development tool set is a collection of systems integrated together that build an interactive information system (IIS). An IIS is a computer system for distributing the information in a research paper, research project or a large documentation project. The information is broken up according to topic so, an end user can access only the topics that they are interested in. Each topic is given a name and has a brief description. The name and description help the end user in finding the information needed.

The development tool set writes the code that becomes the IIS based on decisions made by the developer. The developer enters the needed information for building the IIS interactively. As parts of the IIS are built the developer moves around the IIS in the same way that the end user would.

The development tool set contains some code that was written by Asymetrix Corporation. The rest of the code was developed by Larry E. Johnson. The development tool set was a part of the requirements for obtaining a masters of science degree from Colorado School of Mines. Support for the development of the development tool set was provided by the United States Geological Survey Information Systems Division Denver. You can make as many copies of the development tool set as you wish. You can modify it in any way that you wish, just keep the acknowledgment to Asymetrix Corporation. There is no warranty for the development tool set, if it destroys any part of your computer system or other software it is your fault.

Before You Start

The development tool set is a software product that runs under ToolBook by Asymetrix Corporation. It is necessary, in order to build an interactive information system (IIS) with the development tool set, that a copy of ToolBook with multimedia extensions is on your development computer.

There is a system within the development tool set that automatically converts graphic files to Windows bitmaps. The graphics conversion system requires a software product called Image Alchemy by Handmade Software Incorporated.

Both ToolBook and Image Alchemy need to be on the 'C:' hard drive of the development computer system. A subdirectory called 'C:\iis\graphic' needs to be created. The directory 'C:\iis\graphic' is where all converted graphic files are placed. This directory needs to be copied with the final IIS (?.tbk) file and the 'runtime' files for ToolBook (see ToolBook manual) to make a usable end product.

Navigation

As the interactive information system (IIS) is being by the development tool set, the developer will move around the IIS in the same way as an end user would. There are two different type of pages (or backgrounds) within the IIS.

The first type of page is a menu page. A menu page has a list of topic names and an overview for each topic. This page allows an end user or a developer to navigate between topics. When the mouse curser is placed over a topic name, the description of that topic will be displayed. If the topic
name is clicked on by the mouse, the system will go to the topic page of that topic name. There can be more than one menu page, for example, sub topics within a given topic. The ability of the development tool set to break down information into topics and subtopics allows the end user to find information quickly. On all pages, except for the first menu page, there is a button in the lower right corner when clicked on will return the user or developer to the calling menu page.

The second type of page is the text field page. The text field page contains some amount of text for a topic. This page contains a field that stores the text, the field has a scroll bar at the right to help in reading all of the text in the field. If there is more that one page of text for a topic the 'next' and 'previous' buttons are place on the pages to help move around within a topic's text. A button that returns a user to the calling menu page is also on each text page. If there is figures that the text refers to then there will be one or more buttons to the right of the text field. These buttons display a graphic file that contains the figure. Buttons for printing out text and searching for a given word or word phrase are on each text field page.

Building a Menu

To make a level of the menu system you need to have all the text files broken up according to topic. A name and a brief description is needed for each topic or subtopic. To build a menu page the 'Make Menu' button needs to be clicked on. When the make menu button is clicked on it asks for a topic name and description. After the first topic is built, additional topics can be placed on the page by clicking on the 'Add a Topic' button. Selecting the 'No' option within the 'Add a Topic' script will remove the ability to add additional topics on that menu page.

Entering into a topic by clicking on the topic name will allow you to build a new menu page that has sub topic or to import the text file for the topic. Each level of the menu building system works in the same way as previous levels.

Importing Text

The text import system allows you to bring a text file into the interactive information system (IIS). It automatically breaks the file up into smaller pages and places the correct buttons for changing pages and returning to the menu system. The text file has to have the '.txt' extension. This system allows you to browse through the different files available on the computer system in order to select the correct text file. Each topic of text should be in it's own text file. This way each topic within the IIS contains only that topics text.

Importing Graphics

The Import Graphic Files button will allow you to import pictures and figures in the form of graphic files. The system converts most graphic files to the bitmap format. You will need a name for each figure or picture, for example "fig. 1.1".

The Graphics Import System allows you to place graphic buttons on a given text page. The graphic buttons bring up a graphic file that contains figure, graph or picture. This is used to represent some figure that is talked about in the text, for example (see figure 1.1). Initially the graphics button isn't connected to any graphics file. The first time the button is clicked on by the mouse, the file browser comes up. With this you can select a graphic file. If the graphic file isn't a windows bitmap then the system converts the graphic file into a bitmap and places it in the subdirectory
C:\IIS\GRAPHIC. After this the system asks for a location on the screen where the graphic file is to appear. The graphic file can be displayed with a click on the button with the mouse. A click on the graphic display with the mouse makes the graphic display disappear. If the wrong graphic file is selected the file can be changed by clicking on the graphic button with the right mouse button. This command brings back the file browser.

The Set Graphics in Stone button takes away the ability to change which file a graphics button is connected to. Before this button is clicked on, all the graphics buttons on this page should be connected to the correct graphic files.
Appendix I

-- Code to the on lin help system
-- all of this code was programmed by larry e. johnson

-- Code to the buttons that actavate the help system

-- Code that enters the first page (introduction) of the help system
to handle buttonUp
    set bookMark of this book to the idNumber of this page
    go to page 1 of background "help"
end

-- Code that enters the fourth page (Building a menu) of the help system
to handle buttonUp
    set bookMark of this book to the idNumber of this page
    go to page 4 of background "help"
end

-- Code that enters page five (Import text) of the help system
to handle buttonUp
    set bookMark of this book to the idNumber of this page
    go to page 5 of background "help"
end

-- Code that enters page six (Import graphics) of the help system
to handle buttonUp
    set bookMark of this book to the idNumber of this page
    go to page 6 of background "help"
end

-- Code for navigation buttons within the help system

-- Code to get to page one
to handle buttonup
    go page 1 of this background
end

-- Code to get to page two
to handle buttonup
    go page 2 of this background
end

-- Code to get to page three
to handle buttonup
    go page 3 of this background
end

-- Code to get to page four
to handle buttonup
    go page 4 of this background
end

-- Code to get to page five
to handle buttonup
  go page 5 of this background
end

-- Code to get to page six
to handle buttonup
  go page 6 of this background
end