

Rochester Institute of Technology

## RIT Digital Institutional Repository

---

Theses

---

5-18-1989

### Interactive interface design: Graphic Design Archive prototype 2.0

Cathleen Britt

Follow this and additional works at: <https://repository.rit.edu/theses>

---

#### Recommended Citation

Britt, Cathleen, "Interactive interface design: Graphic Design Archive prototype 2.0" (1989). Thesis. Rochester Institute of Technology. Accessed from

This Thesis is brought to you for free and open access by the RIT Libraries. For more information, please contact [repository@rit.edu](mailto:repository@rit.edu).

***Interactive Interface Design  
Graphic Design Archive Prototype 2.0***

**Rochester Institute of Technology**

A thesis submitted to the faculty of  
the College of Fine and Applied Arts  
in candidacy for the degree of

**Master of Fine Arts**

by **Cathleen Britt**

May 18, 1989

## **Approvals**

Advisor: **R. Roger Remington**

Date: *May 18, 1989*

Associate Advisor: **Robert Keough**

Date: *5-18-89*

Associate Advisor: **Dr. Barbara Hodik**

Date: *18 May 1989*

Special Asst. to the Dean for Graduate Affairs: **Philip Bornarth**

Date: *5/24/89*

Dean, College of Fine and Applied Arts: **Dr. Robert Johnston**

Date: *6/5/89*

I hereby grant permission  
to the Wallace Memorial Library of RIT  
to reproduce my thesis in whole or in part.  
Any reproduction will not be for commercial use or profit.

***Special Thanks to:***

**Roger Remington** for direction, encouragement, wisdom  
and the loan of his Macintosh computer

**Mark Collen** and **Steven Kurtz**  
for their patience, technical knowledge and good common sense

**Robert Keough** and **Barbara Hodik**  
for their encouragement and participation in thesis committee meetings

**Tom Wells** for proofreading this report

## **Contents**

Introduction	v
1. A Guide For Interactive Interface Design	1
2. The Organization of the Graphic Design Archive Database	4
3. Prototype 1.0	6
4. Development History: November 29 - January 5	9
5. Development History: January 5 - April 1	10
6. Prototype 2.0	12
7. Conclusion	17
8. Endnotes	18
9. Sources Consulted	19
Appendix A: Prototype 2.0 User's Guide	
Appendix B: Sorting and Programming Documentation	
Appendix C: Flow Charts	
Appendix D: Prototype 1.0 Screen Designs	
Appendix E: February-March Demos, Screen Designs	
Appendix F: Prototype 2.0 Screen Designs	
Appendix G: Prototype 2.0 Evaluation Documents	
Appendix H: Apple Computer, Inc. <u>Human Interface Guidelines</u>	
Appendix I: User Profiles, Pioneers in American Graphic Design, Catalogue Outline, Walker Show: Graphic Design in America	

## ***Introduction***

Recent advancements in personal computing and videodisc technology give us the power to link together and access large bodies of information. A single twelve inch videodisc has the capacity to store over 100,000 still frame images. The computer, provided with the proper syntax, can access any of those images with astonishing speed. The designer of the computer interface holds the key that can unlock the syntactical barriers and open these information networks to all who are interested.

The intent of my thesis project is to provide public access to a small part of the growing Graphic Design Archive. The Archive's purpose is to document the history of graphic design, to provide a tool for graphic design education and to promote the graphic design profession. The Graphic Design Archive database includes approximately 4000 images representative of the history of graphic design. The focus of Prototype 2.0 is the work of twenty designers who worked in New York between 1930 and 1955; a total of 806 images comprise the visual part of the database accessible by using the prototype software.

Although the programming that runs the prototype is lengthy and complex, the concepts behind it are simple. The video images can be accessed by choosing from a list of designers' names (such as Herbert Bayer or Lester Beall) or from a list of the various media used to create the work (such as Graphic Image, Sign or Photographic Image.) The images have also been organized into a tree-like structure according to the type of work they represent, and can be accessed by choosing a branch of the tree that represents, for instance, posters or typography. Once selected by a click of the mouse, the chosen images can be viewed in a matter of seconds, and choices made are automatically recorded and compiled in a list that can be accessed again later on.

In designing the visual interface, I have stressed simplicity and consistency in screen graphics to promote easy understanding of how to use the software. I have sought to make the interface design an unobtrusive accessory to the rich source of information available in the Graphic Design Archive.

## ***A Guide for Interactive Interface Design***

The discipline of interface design combines elements of the fields of semiotics, cognitive science, computer science and the graphic arts. It is a methodical discipline that also draws on one's intuitive powers and one's ability to think creatively. This guide is my response to being unable to find a practitioner's guide to interactive interface design on the shelves of the library. It's the result of an analysis of the many intricate phases of development of the Graphic Design Archive prototypes 1 and 2. I have focused on design principles for iconographic and metaphorically-oriented desktops like those developed by the Apple Computer, Inc., and my recommendations are meant as general guidelines. Later in this report you will find that I have used this guide as criteria for the criticism of Prototype 1.0 and the description of my thesis project, Prototype 2.0.

### ***1. Examine and Organize the Database***

Every database project is governed by preconceived ideas about its ultimate function. Examine the data that you have to work with and make sure that it supports your ideas about the interface. Also be aware that even elegant accessing tools are useless if they allow users to search for non-existent or incomplete data.

**a. Clarify the basic premise** of the project by identifying the reason that it was initiated. Determine whether that reason is still the motivational force behind the project. If there has been a change, determine whether the database can keep up with the changes.

**b. Categorize the data** according to formal similarities. This process may reveal overlapping data structures. Examine the logic of these categories with an open mind. One of the great advantages of creatively organizing a database is that the process can reveal previously hidden relationships between different types of information.

**c. Keep a detailed record** of the organization of the data and any organizational tools that were developed in the process.

### ***2. Develop a user profile***

Because different groups of people are attracted to different types of information, the results of step one will help you to develop a general user profile. Targeting your design to this specific user group will guide you in defining the types of accessing tools needed, but limiting the application to this audience might make it obsolete before its time. Build flexibility into your interface design.

**a. Develop a profile** of the individuals or groups who would most likely take a strong interest in the data. Keep in mind that if the extent of the data does not match the expectations of the users, the system will fail to be accepted as a viable interface no matter how well it has been designed.

**b. Make a list** of groups that would definitely not be interested in the data. (Age may be an important determining factor.) Does eliminating these groups allow greater freedom in the design?

**c. Who falls in between?** What is the likelihood that these individuals might take an interest? How can your design provide for these users?

### ***3. Examine the logic for developing accessing methods***

It is important to examine traditional methods of accessing data. If the tools for data access allude to and are designed from traditional sources, rather than from computer related ones, the chances of alienating and confusing the user will be greatly diminished.

a. **List the different methods of accessing the database;** for example, list the various organizational categories that could be selected (subject, title, author, et cetera.) Refer to the results of step one while creating this list.

b. **Prioritize the list** from most logical methods of access to least logical. This will direct you in deciding where to begin the process of tool development. Keep in mind that several different information retrieval tools may be necessary to make the database completely accessible. The refinement of these tools should be an ongoing process in the remaining phases of the interface design.

#### ***4. Develop a hierarchical flow of operations***

Regardless of the number of different accessing tools under development, the final interface will need to be held together by a central accessing methodology, a hierarchical sequence of events necessary for successfully retrieving information from the database.

a. **Review the steps** it will take to retrieve different kinds of information from the database using the planned tools.

b. **Develop a flow chart** that illustrates the retrieval processes.

c. **Examine the chart for redundancies.** Can some of the processes be simplified or combined? Is there a need for more than one way to get to the same basic type of information?

d. **Revise the flow chart** to reflect any new thinking. Revise the accessing tools as needed.

Once the hierarchical order is established, the process of designing the visual interface can begin. The communication of this hierarchical order, through interpretive screen graphics, will help to put the user in control of operations and prevent the frustration that results from premature or unintended interactions. Test your interface ideas as you develop them. Objective input from one or two test subjects can reveal problems that might otherwise be overlooked.

#### ***5. Use the concept of physical space in screen design***

According to Richard Bolt, author of *The Human Interface*, "People are good at using the space around them for organizing and storing things. They lose this option, though, when they sit down to work with computers. The opportunities and means to use space in dealing with data just aren't there."<sup>1</sup> It is the job of the designer to create an illusion of physical space that allows people to use their organizational skills to keep track of and store their findings. Standard devices useful in this regard include the following:

a. **Metaphors** that allude to an object, a series of objects, a situation, a procedure or a place can communicate a sense of physical space by getting the user to create a mental picture of and identify with the thing to which the allusion is made. The communicative power of the metaphor can also be a drawback. The use of the metaphor can be the cause of unnecessary complexity in screen graphics and unfulfilled or restricted user expectations.

Iconographic and verbal elements of screen design may or may not have metaphoric significance; in either case they can help create an appropriate sense of space, and the following comments apply.

b. **Icons** used as representations of operational procedures are an effective graphic device in interface design. Well designed icons can be informative and save space.

c. **Verbal elements** should be simple, concise and unambiguous.



## **6. Rules for screen graphics**

The Apple Computer Company has published ten fundamental principles of design for their desktop interface.<sup>2</sup> A copy of these principles can be found in appendix H of this report. The following is a combination of some of their principles with some of my own.

**a. The use of a grid system in the design of screen graphics** provides the user with a sense of visual stability. For the designer, a grid system provides a consist template for use in the placement of menus or icons on the computer screen. Designing with a grid system can greatly reduce the amount of time needed for user comprehension and the amount of mouse movement necessary to complete actions.

Construct a basic grid unit from the proportions of a single pixel. This basic unit should, as closely as possible, reflect the size of the typographic unit most commonly used in the interface. Duplicate the unit as many times as necessary to fill the screen. Use the grid to construct a system of margins and axes and to determine the placement of menus, icons and buttons.

There are two books that I have found useful as references for the construction of grids: [Grid Systems in Graphic Design](#) by Josef Muller-Brockman,<sup>3</sup> and [The Grid](#) by Allen Hurlburt.<sup>4</sup>

**b. Proportion the size and style of the type, the amount of text and the size of the graphic elements** to the overall size of the screen and to the basic unit of the grid. Keep the style of buttons and menus consistent throughout the interface.

**c. Determine the hierarchy** of importance of textual and graphic information. Each screen should visibly reflect this hierarchy.

**d. Recognize the graphic and technical limitations** of the software and the computer terminal. The energy spent in trying to get around these limitations might very well be better used in finding and working with the strengths of the available graphics package.

**e. Don't get carried away with special effects.** Keep your graphics simple, direct, concise and legible. "Good design must communicate, not just dazzle. It must inform, not just impress. Graphics are not merely cosmetic. When they are clear and consistent, they contribute greatly to ease of learning, communication, and understanding. The success of graphic design is measured in terms of the user's satisfaction and success in understanding the interface."<sup>5</sup>

**f. Provide consistent visual feedback to user actions,** dialog when necessary and ways to back out of or cancel operations. "People expect their physical actions to have physical results, and they want their tools to provide feedback. Users want topics of interest to be highlighted. They want to see what functions are available at any given moment. If grave consequences might follow from any of those functions, they want to know about them - before any damage is done. They want clues that tell them that a particular command is being carried out, or if it cannot be carried out, they want to know why not and what to do instead."<sup>6</sup>

## ***The Organization Of The Graphic Design Archive Database***

"The history of graphic design has been scattered among the pasts of art, printing, typography, photography, and advertising. Today's increased interest in graphic design among history teachers, scholars, researchers, librarians, and professional designers has led to the recognition of the importance of graphic design history."<sup>7</sup>

The Graphic Design Archive on videodisc contains approximately 4000 images concerned with the history of graphic design, a collection that began with a group of slides from the personal collection of Professor Roger Remington of the Department of Graphic Design, College of Fine and Applied Arts at RIT. The archive on computer disc contains textual data documenting those 4000 video images. The two separate elements have been integrated with Hypercard software to form a unified database for the development of prototypes 1 and 2. Credit for organizing the database structure goes to Professor Steven Kurtz from the College of Applied Computer Science at RIT.

Both prototypes consist of three main sections called stacks. Prototype 1.0 includes the Card stack, the GDA Demo stack and Glossary stack. In Prototype 2.0 the Demo stack is renamed the GDA Start stack and the Reference stack replaces the Glossary.

### ***The Card Stack***

Each screen in Hypercard is called a card. Each card is assigned, by Hypercard, an identification number. There is one HyperCard card for each image on the videodisc that is used in the prototype. It is these cards which make up the Card stack. Each image on the videodisc has an assigned frame number; this number reflects the order in which the images were pressed onto the disc. Each image also has a classification number which corresponds to the position of the image in the Classification Tree, described on the following page.

Areas on the cards that contain verbal or numerical information are called fields. Each card in the Card stack has a field that contains the frame number of the video image that it represents. The cards have other fields that contain pertinent information about the video image, such as the name of the designer, the date the work was created, the location, the medium used, et cetera.

Cards are stored in stacks. Stacks, cards and fields all can be assigned instructions that are carried out by HyperCard if specific conditions are met. These instructions are called scripts. The Card stack has a script that activates each time a card is accessed or opened. That script instructs Hypercard to get the frame number from the frame number field on the card that has been accessed and commands the videodisc player to find and display that frame number on the video monitor. In this way the user is able to simultaneously view the video image and its corresponding card with related textual information.

The numerical order of the images on the disc is not necessarily the most desirable order in which to access them. Controlling the order in which these images can be accessed is made possible by sorting the Hypercard cards in different ways. The card stacks for prototypes 1 and 2 have each been sorted a total of twenty times. For illustrative purposes, I will describe the first sequence of five sorts and the effect of this sorting order on the organization of the cards.

**Sort by date** - so that the images are represented in chronological order.

**Sort by medium** - the cards are now categorized by medium, each medium is in order by date.

**Sort by location** - categorized by location, each location by medium, each medium by date.

**Sort by designer** - categorized by designer, each designer by location, each location by medium, each medium by date.

**Sort by classification number** - categorized by classification number, each classification number by designer, each designer by location, each location by medium, each medium by date.

After every sequence of five sorts, the resulting order of card identification numbers was recorded. This process was repeated with the sorting order changed to organize and record five different lists of card identification numbers. These lists are used in accessing the cards in conjunction with the Query card, the Stack Builder card and the Classification Tree (see below.) Complete documentation of the sorting process can be found in appendix B.

### ***The GDA Demo Stack and The GDA Start Stack***

These stacks contain the entry or Welcome card, the Query card (prototype 1) or the Stack Builder (prototype 2) card, and the Classification Tree. It is in these stacks that the sorted lists of card identification numbers from the card stack are stored. With the Query card or the Stack Builder card the user is able to indicate (by choosing from a list of designers names, a list of media, or a list of locations) what he or she would like to see. If for instance, the user indicates that he or she would like to see all the work done by Lester Beall, the computer finds the previously stored designer list, and pulls out of it the card identification numbers that correspond to Beall. The computer sends that list, with the user, to the card stack and allows access to only those cards that are on the list (the cards that represent work done by Lester Beall.)

The Classification Tree, a system developed by Roger Remington, organizes the images in the prototype database according to the type of work that they represent. There are two major branches to this tree. Design Work includes all the design applications and Designers' Archive includes things like photographs of the designers, their surroundings and their correspondences. The tree has a total of 249 branches. Each branch of the tree has been assigned a classification number. This number is assigned also, to each image in the prototype that has been classified under that branch of the tree. By choosing a branch of the tree, the user instructs the computer to access the classification number list, allowing access to all the images that have the same classification number. The deeper into the tree one navigates, the more specific the choice is and the smaller the number of associated images.

### ***The Reference Stack and The Glossary Stack***

The Reference stack consists of a series of cards that contain biographical information about the designers in the database, definitions of important terms used in their biographies, and a listing of bibliographical sources. Extensive cross-referencing encourages and facilitates study of the manifold influences on the designers and the historical contexts of their work. The Reference stack can be easily accessed from the other two stacks.

## ***Prototype 1.0***

The following is a critique of Prototype 1.0 based on the principles outlined above in the Guide for Interactive Interface Design. This critique was made so that the problems of the first prototype could be recognized and the strengths could be used as a foundation for Prototype 2.0. Printouts of the screen designs from Prototype 1.0 can be found in Section 1 of appendix D.

### ***The Database***

Prototype 1.0 consists of the first 601 images found on the videodisc. These images were chosen simply for expediency. The development team needed images and data in a hurry so that they could test new accessing tools. As a result, the prototype accesses a wide range of graphic imagery spanning some 2000 years, many of the video images are of poor quality, and the data in the card stack is incomplete and, in some cases, inaccurate.

The Glossary stack was contributed by Ellen Lupton, Curator at The Herb Lubalin Study Center of Design and Typography at The Cooper Union. The development team recognized the potential of this stack as a vehicle to access the database and as a source of support material for the archive. The Glossary was visually redesigned, a few cards were added for explanatory purposes, and Mark Collien revised the scripts to integrate the new stack with the three stacks (GDA Demo, Cards, and Help) that were already in existence. A problem remains, however, in that the information included in the original Glossary doesn't complement the images on the videodisc or the data in the Card stack of Prototype 1.0.

### ***User profiles***

Because of the experimental nature of the prototype (the first of its kind at RIT), user profiles were developed from ideas of what the archive might someday become and did not take into consideration the nature of the information that actually existed in the database. User profiles ranged from the scholar and the researcher to the student and the non-designer. A copy of these user profiles can be found in appendix I. In retrospect, it is apparent that the data and imagery in Prototype 1.0 is simply not extensive enough or cohesive enough to support research.

### ***Logic in developing accessing methods***

There are six different ways to access the prototype database: by choosing a designer, a location or a medium, by accessing a previously saved file, by querying the database for more information, or by choosing a branch of the classification tree. Five of these six accessing methods are provided to the user through the use of the Query card.

### ***Query card description*** (screen design can be found in Section 1 of appendix C)

The functional and visual theme of the Query card is mathematical set notation or Venn diagrams. The user is presented with two large circles (list A and list B,) and can decide which information is to be included in each list. The process for selecting information for list A begins when the user clicks anywhere in that field to activate it (a bold outline indicates that the field is activated.) Then the user must click on the Select A button and choose from 5 options: read a file, choose designer, choose location, choose medium, or current list. Choose designer, choose location and choose medium bring up itemized lists of these topics. Clicking on one of these items will put the selection into list A and indicate how many images it represents (for instance there are 42 images in the database designed by Herbert Bayer.) By repeating this process, the user can then select, for example, Germany for field B. To this point there have been eight clicks of the mouse, and the user has yet to decide how to relate the two fields of information. The choices are represented by 8 different Venn diagrams at the bottom of the screen, each with a verbal description:

**All Items In either list A or list B**  
**All the Items from list A**  
**Only the Items from list A that are not in list B**  
**Only the Items that are in both list A and list B**  
**Only the Items from list B that are not in list A**  
**All the Items from list B**  
**Only the Items in list A or list B, but not in both**  
**No Items are selected**

After deciding which relationship to use, (by clicking on the correct Venn diagram,) the user must click on the data card icon at the top of the screen and go to the Data Cards. There the user can finally see the chosen images.

### **Query card problems**

Successful use of the Query card requires an understanding of set notation and Venn diagrams. This tool functions well when it is used by individuals with a thorough knowledge of mathematics or computer science. Unfortunately, for most others, the query language (the verbal descriptions of the Venn diagrams) is counter-intuitive:

In query language, "and" indicates an intersection of two sets. This conflicts with the intuitive sense of "and" as "give me everything in both sets."

In query language, "or" indicates the union of two sets, eliminating any duplicates. Intuitively, "or" means "give me one or the other, but not both."

The real power of the Query card is in the underlying concept of a current list. Every time the user retrieves information from the database, it is in the form of a list of card identification numbers, stored in a hidden field on the Query card. This list is what is referred to, by the Query card, as the current list. By using the current list as a choice for one of the fields on the Query card, the skilled user could continue to refine that list, tailoring it to his or her specific needs. However, the concept of the current list is hidden to most users, greatly diminishing the power of the Query card as a database accessing tool.

The Query card requires of the user a minimum of 10 different decisions in the process of selecting information to view (some of these processes may have to be repeated.) The effect of such a complicated and lengthy process is that the Data Cards and images are pushed into the background in terms of importance. The Query card process takes precedence over the data.

**The Classification Tree** (screen design can be found in Section 1 of appendix C)

The power of the classification tree as a database accessing tool is greatly reduced because of the failure to communicate the idea of the current list. When returning to the Query card from a specific branch of the tree, all the images that fall under that branch are automatically put into the current list. To use the tree information in a database query, the user needs to recognize the existence of the current list. Without an understanding of the current list, a user could select images of magazine ads and of designs by Paul Rand, but might have trouble selecting magazine ads by Paul Rand.

### ***Hierarchical flow of operations***

Prototype 1.0 was the first of its kind at RIT and, consequently, a learning experience for everyone involved. The emphasis was on tool development and database organization, rather than interface design. No flow charts exist for this prototype.

There are three different accessing tools in the prototype (the Glossary, the Classification Tree and the Query card) that access the same basic information. They all have their advantages, but are independent and unrelated (no visual or systematic hierarchy), causing confusion about what happens to information collected with one tool when switching to another. The concept that is supposed to unite these three accessing tools- the current list- fails to accomplish the task.

### ***Physical Space in Screen Design***

The use of metaphor to communicate a sense of physical space is evident in the screen designs of Prototype 1.0. The query card, the data cards, the glossary and the help stack, display the characteristic Macintosh grey border and shadow, which give the user the sense of an actual paper card, with a directed source of light. A map of the Classification Tree with a flashing marker creates a sense of direction for the user as they navigate through the tree. The Help stack incorporates the use of tabs to give the idea of an easily accessible file system. The only instance where the use of metaphor impedes user progress is on the Query card. The Venn diagrams do not represent a universally accepted ideology, and they fail to provide a visible manifestation of the current list or a visible record of user progress. More generally, the metaphors used in the prototype are not subtly represented; they have been illustrated as literally as possible, resulting in screen graphics that are awkward and oversized. Also, the verbal signs used in Prototype 1.0 are generally verbose and often ambiguous.

The database accessing tools and the navigational tools of Prototype 1.0 are all represented through the use of icons. They are located in the upper right corner of every card in the interface. The Glossary, Query card and Classification Tree icons seem appropriate and communicative. The icon that represents the Data Cards is an illustration of a scrollbar, which is an element on the data cards but is not representative of the data cards themselves. This causes confusion when the user tries to choose the correct tool to access the data cards and the video images.

### ***Screen graphics***

The graphic interface of Prototype 1.0 was designed without the use of a grid system. The size of buttons and icons are not consistent with, or proportional to, the size of other elements on the screen. Margins, axes and rules are non-existent, disturbing the user's sense of visual stability, especially in the transition from one stack to another.

The screen graphics are not organized according to any sort of visual hierarchy. They do not communicate or even hint at a hierarchical order when, in fact, certain events do need to happen in a specific order and certain elements do have more importance than others.

### ***Prototype 1.0 Critique Conclusion***

If the success of an interface design is measured in terms of user satisfaction and understanding, then Prototype 1.0 falls short. The failings of the first prototype were taken into consideration in preparing Prototype 2.0.

## ***Prototype 2.0 Development History: November 29 - January 5***

### ***New Stack***

Early in the winter quarter I met with Roger Remington and we discussed the possibility of targeting the new prototype towards the audience of the Graphic Design in America show scheduled for October 1989 at the Walker Art Center in Minneapolis, Minnesota. Roger made a list of the important pioneering American designers and after checking the database I found that twenty of those designers were represented by a total of 1287 images currently on the videodisc. I then compiled and sorted the cards representative of these images, creating a new stack that could be accessed with the query card in Prototype 1.0. I reviewed the new stack one card and one image at a time, deleting those that were of poor video quality or duplicates. After several editing sessions the stack was down to 890 cards and all the images were of good visual quality. (A copy of the catalogue outline for the Graphic Design in America show and a copy of the original list of pioneering American graphic designers compiled by Roger Remington can be found in appendix I of this report.)

Data entry was the next major project. Roger seemed to be the only one who knew where to find the necessary information about the images. So while he worked on the data entry, I began working on tool development.

### ***Tool Development***

I spent the early part of December working on tools that had been left unfinished by the development team of Prototype 1.0. My thinking was that I should get these tools in working order as quickly as possible, integrate them into the prototype and then begin the process of redesigning the entire interface.

The camera tool was developed to give the user the ability, while scrolling through the data cards, to record specific images and save them as a separate file. That file could be accessed as often as necessary by the Query card. This tool could be a valuable resource for a teacher developing a course or a researcher needing repeated access to the same information.

A front end **browse** tool was developed to cater to the uncommitted user, such as a museum visitor who has no knowledge of computers and does not want to put effort into learning how to use the prototype but is interested in the information contained in the database. The user would be required to click on the browse button which would bring up a list of designers' names. After selecting one of these names, the user would then be able to **scan** the corresponding images (at one second intervals) or move through them at their own pace using the scrollbar.

### ***Flow Charts***

By the middle of December, it became clear to me that the development of a flow chart of the entire project was a necessity. I was losing sight of the overall flow of the prototype and was unable to make decisions about how the new tools should function in relation to those already in existence. Copies of the five flow charts developed for Prototype 2.0 can be found in appendix C.

The first three charts (dated Dec. 15, Dec. 18, and Jan. 2) were attempts to determine exactly how my newly developed tools would integrate with Prototype 1.0. I knew that the query card had to be redesigned but I still saw the Classification Tree, the Query card, and the Glossary as separate ways of accessing the same data. My frustration grew as I became aware of the enormous task involved in trying to reorganize the flow of the prototype. It was not the kind of thing that could be quickly sketched out and put to a test. I was intimidated by the extent of and complexity of the programming. Several people had been involved in setting up the first prototype, and much of the scripting had not been documented. There were external functions, hidden fields and global variables. How was I ever going to make any sense out of this?

### ***The Turning Point***

On January 5 Nicholas Negroponte, the Director of the MIT Media Lab spoke at Xerox. Mark Collien and I went to the afternoon lecture. Negroponte discussed the need for more levels of communication from the desktop interface. He compared a first time user to someone visiting a foreign country and unable to speak the language. At a breakfast table, with everything laid out in the open, the foreigner would be able to indicate what they wanted by pointing or gesturing.

On January 6th I heard a tape of a WXXI radio broadcast of The National Press Club Luncheon featuring James Billington, the Chief Librarian at the Library of Congress in Washington D.C. He discussed the development of optical disc technology and its ability to provide easy public access to large amounts of information and how access to information is one of the great strengths of the technical age. "The founding fathers sensed early what others are just discovering now, that both a body of knowledge and access to it must aspire towards universality if you are to aspire towards true democracy."<sup>8</sup>

These two lectures made something click. I suddenly realized that instead of trying to force Prototype 1.0 to work, I should use it as a foundation for a new prototype. I should begin working on a new version of the Query card and access to the images should be a priority instead of an end product. There should be a menu of choices always visible to the user instead of hidden behind a mouse click. I should give the user a place to store and access any findings and all of this should take place on one card so that the user's sense of process is not interrupted. There should be one central list-building card from which the user is able to access other applications and then return to with information to add to his or her list.

### ***Development History: January 5 - April 1***

#### ***The Central List Building Card*** (flow chart can be found in appendix C)

I began working on a flow chart to illustrate my idea of the Central List Building Card. I wanted to give the user complete control over the building of a list. I wanted to tell the user exactly how many images were being selected and give him or her the opportunity to scan the selections before going to the data cards. I wanted to keep the lists itemized so that the user could trace steps backward and review items that he or she had seen before or remove ones that no longer were desired.

#### ***January 12th Meeting***

The Graphic Design Archive team had been meeting every Thursday to report new developments. At the January 12th meeting Steve Kurtz announced that the American Video Institute had provided some release time for him so that he could work on the Graphic Design Archive. He wanted to begin meeting weekly with Mark Collien and me for brainstorming sessions. We decided that the first one would be Friday, January 13th. Also at this meeting Susan Preston-Mauks, a student of psychology from the College of Liberal Arts, was introduced. She was interested in initiating an evaluation system that would measure the effectiveness of the new prototype. We decided that we would run an evaluation session the first week in February to test my ideas. Now the pressure was really on!

Later in the meeting I presented a flow chart that illustrated my idea of the Central List Building Card. I was asked to describe what it would actually look like on the computer screen and was unable to do so. I realized that it was going to take more than a flow chart to convince Steve Kurtz, Mark Collien and Roger Remington. I needed to quickly become a programmer and build my ideas into real applications that could be presented and tested.



### ***First Brainstorming session***

Steve, Mark and I agreed that most of the metaphors used in Prototype 1.0 were either too abstract or too literal, and that we needed to find a more efficient method of communicating the simple idea of a list building system. We decided that the metaphor of a Stack Builder was a good one because it reflects the manner in which Hypercard stores information. Steve and I both agreed to build a demo over the weekend. He was going to try to revise the current Query card and I was going to make a first attempt at a Stack Builder card. We would present our results at the meeting next Thursday.

### ***January 20th Meeting***

I presented my first working version of the Stack Builder card. On the left of the card was a menu that included designer, location, medium and file. On the right was a simple rectangle labeled "The Stack." Clicking on designer, for instance, brought up a list of designers' names. Selecting a name from that list would put it into the Stack along with the number of images on the disc that were done by that designer. Those images could then be scanned by clicking on the scan button. This simple demo finally convinced Steve that I was on the right track. He never again mentioned revising the Query card from Prototype 1.0.

We discussed the need to develop a Glossary stack that was tailored to the information in the new card stack. Roger mentioned that he would ask for volunteers from his class of seniors to research and write the biographies of the twenty designers represented by the database. We also discussed the need to find a new name for the Glossary, and Roger announced that he had completed the data entry for the new card stack. After the meeting I examined the stack and initiated the sorting process.

### ***Finding The Right Query Tool***

The Stack Builder card was a great beginning but at this point nowhere near a final product. The evaluation scheduled for early February was postponed, and so was the one scheduled for late March, as I struggled to build the power of the old Query card into the new Stack Builder card. I wanted my prototype to be able to search the database for specific information such as all the Bayer work done in Germany or all the tourism posters designed by Herbert Matter. This kind of database accessing required the use of a query language; there was no getting around it. The trick was to develop a way to communicate this language to the user.

February and March were spent building demos and watching them fall apart in our brainstorming sessions. Mark, Steve and I devoted many hours to these sessions, especially over the spring break when we met two or three times a week. It was not uncommon for these meetings to last five or six hours.

(Appendix E includes printouts of screen designs of three of the demos built between early February and late March, as well as printouts of the final demo described below.)

### ***The Final Demo***

In late March I had finally come to a solution that we all could live with. I had to quickly begin the process of visually designing the Classification Tree, the Reference stack (the new name for the Glossary) and the Data Cards. I had four short weeks left before the thesis show to pull the final prototype together. Susan and I scheduled another evaluation session for the first week in April. (Remaining developments of the prototype made during this period are described in the next section.)

## ***Prototype 2.0, Final Version***

The following description of my thesis project is organized according to the Guide for Interactive Interface Design on pages 1-3 of this report, and is presented as evidence to justify the design decisions made in the development of Prototype 2.0. The following description also offers the reader an opportunity to compare Prototype 1.0 (discussed on pages 9-12) to Prototype 2.0 using consistent critique guidelines.

### ***The Database***

The purpose of the Graphic Design Archive hadn't changed in the process of developing the second prototype: to document the history of graphic design, to provide a tool for graphic design educators and to promote the graphic design profession.<sup>9</sup>

Prototype 2.0 consists of 806 images (on videodisc) created by a group of twenty pioneering graphic designers who worked between 1930 and 1955. Each video image is linked to a text frame that lists the name of the designer, the country where the work was created, the medium used, the date of the piece, the title and occasionally a comment. A stack of Reference material includes biographies of the designers, definitions of specific terms used in the biographies, and bibliographic information.

### ***User Profile***

Roger and I decided early in the winter quarter to target the new prototype towards a museum audience, specifically the audience for the Graphic Design in America show at the Walker Art Center. This decision influenced the choice of designers whose work would be included in the prototype database.

In developing a user profile, I assumed that the members of the museum audience would be able to read, that they had at some time used a library system to access information, and that they would range from teenagers to senior citizens, from high school students to scholars.

Having worked previously in designing museum exhibits, I was aware that I had only a few moments to pique the curiosity of the museum visitors, and I could not assume that they would have experience using a computer. The prototype had to be simple, legible and direct, and give the user quick and easy access to the images and the data.

### ***Logic for developing accessing methods***

While struggling through the tool development stages of Prototype 2.0, I kept referring back to traditional methods of data accessing. I imagined myself searching for the work of a specific designer in a library, using the card catalogue, walking through the stacks, having it all there in front of me organized in such a way that I could easily remember where things were located. I could pull a book off of a shelf and flip through it, set it aside on a table and go to another shelf to look at something else. This was the kind of flexibility and organizational clarity that I wanted to build into the prototype. I needed to illustrate for the user the differences in the types of information that could be found in the database and the ability he or she had, through the use of the accessing tools, to get to that information.

The accessing options in Prototype 2.0 are identical to those in Prototype 1.0: by choosing designer, location, medium, by accessing a previously saved file, by querying the database for more specific information, or by choosing a branch of the Classification Tree. What has changed is the way in which the data is stored by the computer, the central accessing methodology and the documentation of this process for the user in the form of interpretive screen graphics. What has also changed is that now all of the accessing methods are available to the user through the use of one card, the Stack Builder.

The Reference stack, previously the Glossary stack, is no longer used as an accessing tool because the results are the same as choosing from a list of designers' names. The Reference stack now functions simply as support material to the information contained on the data cards.

To build consistency into the prototype, Roger and I determined the hierarchical order of the accessing tools by examining the information contained on the Data Cards. It seemed logical that the first thing the user would want to know about an image was the name of the designer, then the date, the location, the medium, title and comment. This hierarchy was used to organize the menu of accessing tools displayed on the Stack Builder card.

### ***Hierarchical flow of operations***

The considerations noted above gave rise to the following operational system. A flow chart (dated April 29) that illustrates the hierarchy of operations for Prototype 2.0 can be found in appendix C.

### ***Stack Builder*** (screen designs can be found in appendix F)

The Stack Builder card functions as the card catalogue of the database and also as the table top on which items can be set aside for later use. On the left of the card is a menu of the data accessing tools. On the right is a rectangle labelled "The Stack." All of the information gathered by the accessing tools is put into the stack in the form of an itemized list. For example, if the user's first choice was "Matter, Herbert" from the list of designers' names, "Matter, Herbert" would appear on the first line of The Stack. To the right of this entry would be a number that refers to the number of images done by Matter contained in the database. The next selection would appear on the next line of The Stack, and so on. Once an item has been selected and is listed in The Stack, the user has the ability to scan the images, remove them from The Stack, save them as a file or go to the data cards for those images.

**The first three tools** on the menu are the simplest ways to access the database. They represent three different categories of information: designer, location and medium. Users can access all the work done by a certain designer by choosing from a list of designers' names. Similarly, they can access all the work created in a specific location or done in a specific medium.

**The File button** gives the user access to a file saved during a previous use of the prototype. The information collected in The Stack can be saved as an itemized list or as a unified list with no duplicates and later, through the use of the File button, be put back into The Stack in the format in which it was saved.

**The Classification Tree** is accessed from the Stack Builder card by clicking on the Tree button. From the first card of the tree users can navigate along the different branches, scanning images along the way. When they arrive at a branch that interests them, they can click on the return button and they will be asked if they want to add the information included in that branch to the stack. If they click yes, then they will return to the Stack Builder card and the name of tree branch will appear in the Stack. It can then be treated like any other item.

**The Query function** allows users to be more specific in their search for information. It was designed to combine the unique accessing power of the Classification Tree with the tools that access information categorically (designer, location or medium.) A click on the Query button brings up a series of five rectangular boxes to the right of the menu item (designer, location, medium, file and tree). The user can then proceed to select items just as before. These selections will appear in the of boxes to the right of the menu instead of in The Stack rectangle.

For example, the operations used to access all the Herbert Matter posters designed in Switzerland are as follows. First, after clicking the Query button and selecting the designer list, choose "Matter, Herbert" from the list of designers' names. "Matter, Herbert" will appear in the box to the right of the Designer button. Then choose "Switzerland" from the list of locations and "Switzerland" will appear in the box to the right of the

Location button. Click on the Tree button and navigate to the branch that is labelled "Posters." Click the return button and answer "yes" to the dialogue box that asks if you would like to use the poster information for a query constraint. You will be returned to the Stack Builder card and "Poster" will appear in the box to the right of the tree button. Click on Query Search button and the computer will search the database for the information that matches these constraints. The result will appear in a field that covers the constraint boxes, and along with this field will appear an Add To Stack button. Click on this button and you will be asked to type in a name for the information that was found. Once this is done, click OK and the Stack Builder card will return to its original state with the addition of your typed entry to The Stack. You are now able to scan Matter's Swiss posters, save them as a file, access their data cards, or remove them from the stack.

**The overall structure** of the prototype resembles that of a rimless wheel with a hub and three spokes; the hub is the Stack Builder card, the spokes are the ClassificationTree, the Data Card stack and the Reference stack. The user can access any of the spokes from the Stack Builder card but must always return there before going somewhere else. There is an outer rim that allows the user to access the Reference stack from the Data Card stack, but that is the only outer connection between the spokes and it is unidirectional. The user cannot access the Data Card stack from the Reference stack unless the Data Card stack was the original point of departure. Every card in the Reference stack, the Card stack and the Classification Tree has a Return button in the lower right corner. After arriving at any one of these destinations from the Stack Builder card, the Return button will return the user to the Stack Builder card.

### ***The concept of physical space in screen design***

"It is an old saying that 'form follows function.' This means that the shape of an object is defined by the work it has to do. After a million years of trial and error, nature has produced well functioning shapes, but human history is much too short to compete with nature's richness in creating functional forms. Nevertheless the ingenuity of man has brought forth excellent results in every period of his history when he understood the scientific, technological, esthetic and other requirements."<sup>10</sup>

Two months of concentrated effort in the development of the central accessing logic and methodology of Prototype 2.0 resulted in the implementation of a system that communicates to the user the functions of the software and provides a place that is tailored to accommodate those functions. In an effort to focus on the development of the software I had eliminated all unnecessary graphic marks, including icons. The accessing tools and navigational buttons took shape as terse verbal elements in an effort to avoid the debate that always seemed to revolve around the development of new icons. The profile of the user, the museum audience, seemed to indicate that these verbal elements may be the most effective form of communication. The hierarchical order of the accessing tools, determined by examining the information contained on the data cards, helped to enhance the development of a functional metaphor.

The idea of a central list building device evolved into the Stack Builder card. As mentioned earlier, it would function as the card catalogue of the database and the table top on which items could be set aside for later use. The illustration of the table top idea was accomplished with a simple rectangle labeled "The Stack" and situated on the right side of the card. The card catalogue idea was represented by a menu of buttons and situated on the left side of card. The arrangement of these visual elements from left to right helps to communicate the idea that a linear, multi-stepped process must be followed to select items and put them into The Stack. I eliminated the gray border and shadow that is characteristic of Hypercard cards to make a screen large enough to accommodate the selection process and to push the perimeters of the functional metaphor out into the user's physical space.

The visual standards of the Stack Builder card were carried through, for consistency, to the Classification Tree, the Data Card stack and the Reference stack. Each of these applications has within it unique visual elements, but the overall interface is held together by the theme of linear process that helps the user to read the visual elements from left right and ultimately brings them to the Return button in the lower right corner.

**The Classification Tree** and the **Data Cards** draw upon essentially the same metaphors used in Prototype 1.0. The tree is illustrated by a map with a flashing arrow to indicate to the user the current position within the tree. To the right of the map, the user is presented with the name of the current branch in a bold reversed out bar. To the right of that, in a linear form that reflects the form of the map, the user is presented with the navigational options (the branches that are attainable from the current place in the tree.)

On the Data Cards the scrollbar is a tool to allow movement through a list of images and is a unifying visual element that runs horizontally across the top of the card. Hanging underneath the scrollbar is the textual information about the images. The format of the Data Cards makes use of indenting and of different sizes and weights of type to illustrate the hierarchy of the information and to ultimately carry the eye of the user to the Return button in the lower right corner of the card.

Visual effects that occur when navigating between stacks reinforce the culmination of one process and the beginning of another. When going from the Stack Builder card to any other application the visual effect is that of an iris opening up to reveal a new card and a new process. When clicking the return button the visual effect is that of a closing iris that reinforces the idea of a return to a previous place.

### ***Screen graphics***

**The grid system** used to develop the screen graphics for Prototype 2.0 can be found in appendix F.

A single unit of the grid is proportionate to the size of one screen pixel and the height of an uppercase letter of nine point Geneva type. The final placement of the visual elements on the Stack Builder card, as a result of the use of this grid, provides the axes and margins for the entire interface. The size and the placement of the navigational buttons are identical from card to card and application to application. The scrollbar on the Data Cards is in the form of a black bar placed horizontally along the top of the cards. This visual element is repeated on the Stack Builder card and the Reference cards and is used to highlight the title of the cards and to label certain areas of the cards (such as The Stack on the Stack Builder card.) The use of the grid in developing the visual elements has contributed greatly to a sense of visual stability throughout the interface, has significantly reduced the required amount of mouse movement necessary to navigate through the prototype, and has made the software easy for the user to learn.

Because of the small size of the Hypercard window and the large amount of information that I needed to display on each screen, I used nine point type for most of the text and all of the navigational buttons in Prototype 2.0. I used Geneva because it was the most legible of the sans serif fonts that were available.

**The graphic capabilities of Hypercard version 1.2.1** are limited to one bit per pixel which means that each pixel is either on or off, black or white. In addition, the size of one pixel is quite large in relation to the size of the screen. These graphic limitations result in jagged diagonal lines, awkward circular forms and visually disturbing gray values. Considering the nature and the extent of the information that I had to convey and the graphic limitations of Hypercard, I decided to limit my screen graphics vertical and horizontal lines and black and white elements. I used gray in the prototype only to indicate that certain options are not currently available.

**To provide consistent visual feedback** to the user throughout the prototype system, I established the following rules:

1. Only the options that are currently available are visible in black outlined forms. Unavailable options are either represented in gray or completely hidden.
2. Every button and every line in a field, once selected, becomes highlighted.
3. If the computer takes longer than a few moments to complete an operation then the cursor changes to a watch icon to indicate that that the command is being executed.
4. Dialogue boxes are used to alert the user to any questionable actions attempted. For example, if the user tries to scan, remove, or save an item without first selecting it from The Stack, a dialogue box appears and tells him or her to select something first. When returning from the tree a dialogue box asks whether the user wants to take the information included in the current branch and add it to the Stack or use it as a Query constraint.
5. The user is allowed at any time to cancel an operation, to quit the program completely or to start over.

## **Conclusion**

### **Prototype 2.0 Evaluation**

The Graphic Design Archive Prototype 2.0 was evaluated in a formal session on April 6, 1989. The test subjects, volunteers from Professor Remington's senior studio design class, were organized into groups of three so that each individual could have access to a workstation (a Macintosh II or SE computer, a videodisc player and a video monitor.) After a brief introduction to the Archive and a demonstration of the prototype software, the subjects were asked to complete a worksheet that required them to find different kinds of information using the prototype's database accessing tools. After completing the worksheet the test subjects answered a series of questions designed to evaluate the performance of the prototype and the design of the interface.

The test subjects quickly understood the functions of the software, even the more advanced functions involved with the Query procedure. One subject in particular, who had no experience with a Macintosh was able to complete the worksheet with the correct information before the two others in her group. The overall response of the test subjects to the prototype was favorable and provides evidence that the design principles used to build Prototype 2.0 (those stated in the Guide for Interactive Interface Design on pages 1-3 of this report) are effective. A complete report of the evaluation results written by Susan Preston-Mauks, the project evaluation consultant, can be found in appendix G.

### **A Few Thoughts For The Future**

Prototype 2.0 stretches the limits of Hypercard in the way that it access and stores information, and would become too cumbersome and too slow for a larger database. As the next prototype is developed changes will need to be made in the database software, probably resulting in the use of software other than Hypercard.

The extent of the textual data in prototype 2.0 dictated the level of complexity attainable in the database accessing tools. The addition of more accessing tools would create redundancies in the type of information retrieved. The development of future prototypes will depend on the collection and thorough recording of more extensive textual data about each video image.

The assignment of key words to each image could add an important intuitive search capability to the database. Imagine being able to access all the posters or all the letterheads in the database without having to wind through the branches of the Classification Tree. It would also be a valuable archival resource if the whereabouts of the actual piece was included on each Data Card.

### **The Project**

Prototype 2.0 is the result of an extended period of time (including Thanksgiving, Christmas and Spring breaks) during which I did nothing but concentrate on the interface design and during which I had unlimited access to Roger Remington's Macintosh SE computer. It is also the result of the valuable and weekly input of Roger Remington, Mark Collien and Steve Kurtz. I feel very fortunate to have had the opportunity to work with these individuals and fortunate to have had the freedom to focus on the complex problems involved in the field of interface design.

## ***Endnotes***

**1**Richard Bolt, The Human Interface (Belmont, California: Lifetime Learning Publications, 1984), 1.

**2**Apple Computer, Inc., Human Interface Guidelines. The Apple desktop Interface (Reading, Massachusetts: Addison-Wesley Publishing Co., Inc., 1987), 1-17.

**3**Josef Muller-Brockman, Grid Systems in Graphic Design (Niederteufen, Switzerland: Arthur Niggli LTD., 1981.)

**4**Allen Hurlburt, The Grid (New York: Van Nostrand Rienhold Co., Inc., 1978.)

**5**Apple Computer, Inc., Human Interface Guidelines. The Apple desktop Interface (Reading, Massachusetts: Addison-Wesley Publishing Co., Inc., 1987), 9.

**6**Ibid., 4.

**7**Barbara Hodik, ed., "Introduction" in The First Symposium on The History of Graphic Design (Rochester, NewYork: Rochester Institute of Technology, 1983), 5.

**8**James Billington, Librarian of Congress, Speaker (Washington, D.C.: National Press Club Luncheon, January 1988.)

**9**Barbara Hodik, ed., "Introduction" in The First Symposium on The History of Graphic Design (Rochester, NewYork: Rochester Institute of Technology, 1983), 5.

**10**L. Moholy-Nagy, Vision in Motion (Chicago: Paul Theobald and Company, 1969), 33.



## ***Sources Consulted***

- Apple Computer, Inc. Human Interface Guidelines. The Apple Desktop Interface. Reading, Massachusetts: Addison-Wesley Publishing Co., Inc., 1987.
- Billington, James, Librarian of Congress, Speaker. Speech about expanding the Council of Scholars of the Library of Congress to bring world scholars together to discuss world issues. Washington, D.C.: National Press Club Luncheon, January 1988.
- Bolt, Richard. The Human Interface. Belmont, California: Lifetime Learning Publications, 1984.
- Goodman, Danny. The Complete Hypercard Handbook. Toronto, New York, London, Sydney, Auckland: Bantam Books, 1987.
- Hodik, Barbara, ed. "Introduction" in The First Symposium on the History of Graphic Design. Rochester, New York: Rochester Institute of Technology, 1983.
- Hurlburt, Allen. The Grid. New York: Van Nostrand Rienhold Company Inc., 1978.
- Kerr, Stephen T. "Instructional Text: The Transition From Page To Screen." Visible Language, XX4 (Autumn 1986): 368-392.
- Moholy-Nagy, L. Vision in Motion. Chicago: Paul Theobald and Company, 1969.
- Muller-Brockman, Josef. Grid Systems in Graphic Design. Niederteufen, Switzerland: Arthur Niggli LTD., 1981.
- Nadin, Mihai. "Interface Design and Evaluation - Semiotic Implications." in Advances in Human - Computer Interaction, Volume 2, ed. H. Rex Hartson and Debora Hix, 45-100. New Jersey: Ablex Publishing Corporation, 1988.
- Shafer, Daniel. HyperTalk Programming. Indianapolis, Indiana: Hayden Books, 1988.



## ***Appendix A: Prototype 2.0 User's Guide***



## **Graphic Design Archive**

### **| Prototype 2.0**

#### **| User's Guide**

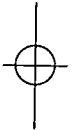
Prototype and User's Guide  
Designed by **Cathleen Britt**  
Rochester Institute of Technology

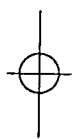
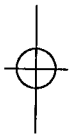




## **| Contents**

<b>Preface</b>	<b>1</b>
<b>System Requirements</b>	<b>2</b>
<b>GDA Stacks</b>	<b>3</b>
<b>Opening Prototype 2.0</b>	<b>4</b>
<b>Set Player</b>	<b>5</b>
<b>Introduction</b>	<b>6</b>
<b>Stack Builder</b>	<b>7, 8, 15</b>
<b>Data Cards</b>	<b>9</b>
<b>Reference</b>	<b>10</b>
<b>Query</b>	<b>11, 12, 14</b>
<b>Classification Tree</b>	<b>13</b>
<b>Saving A File</b>	<b>16</b>





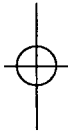


## **Preface**

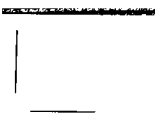
The Graphic Design Archive contains approximately 4000 images pertinent to the history of graphic design, a collection that began with a group of slides from the personal collection of Professor Roger Remington of the Department of Graphic Design, College of Fine and Applied Arts at the Rochester Institute of Technology. The archive on computer disk contains textual data documenting those 4000 video images. The two separate elements have been integrated, using a Macintosh computer and HyperCard software, to form a unified database for the development of Prototype 2.0.

Prototype 2.0 accesses 806 of the images on the Archive's videodisc. These images are from the works of a group of twenty pioneering graphic designers who worked between 1930 and 1955. Each video image is linked to a text frame that lists the name of the designer, the country where the work was created, the medium used, the date of the piece, the title and occasionally a comment. A stack of Reference material includes biographies of the designers, definitions of specific terms used in the biographies, and bibliographic information.

The purpose of the Graphic Design Archive is to document the history of graphic design, to provide a tool for graphic design education and to promote the graphic design profession.







## **System Requirements**

### **Hardware**

**Macintosh Plus, SE, or II with 1 megabyte of memory and a hardisk drive**  
**Macintosh II monitor must be set to two colors - black and white**

#### **Laser videodisc player:**

Pioneer LDV-6000  
Pioneer LDV-4200  
Hitachi 9550  
Sony LDP-1000A  
Sony LDP-1200  
Sony LDP-1500  
Sony LDP-2000

**Monitor:** any quality NTSC video monitor or a standard television set with RF input (not all disc players have RF output, check yours to be sure)

#### **Cable:**

1. From Macintosh serial port to the disc player
2. From disc player to video monitor (video cable with BNC connector on disc player end)

### **Software**

Prototype 2.0 has been written and designed using HyperCard™ version 1.2.1 developed by Apple Computer, Inc. You will need to have HyperCard on your hard drive or accessible on floppy disk to open the prototype.

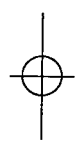
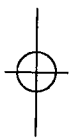
A good source for more information about HyperCard is The Complete HyperCard Handbook by Danny Goodman, published by Bantam Books.

#### **GDA Discs and AVI Image Bank #1 videodisc (8")**

Prototype 2.0 is delivered to you on two floppy disks, the **GDA Start** disc and the **Ref. & Cards** disc, and consists of three HyperCard stacks: the GDA Start stack, the Reference stack and the Cards stack. The easiest way to use the prototype is to copy the three stacks on to your hard drive. The Archive videodisc is available through the American Video Institute at the address below.

#### **For more information contact:**

**Mark Collien**  
GDA Project Coordinator  
American Video Institute  
Rochester Institute of Technology  
One Lomb Memorial Drive  
Post Office Box 9887  
Rochester, New York 14623-0887  
**716-475-6625**



## **GDA Stacks**

To get the most out of Prototype 2.0 you'll need to become familiar with the different types of information included in the database and where in the three stacks that information is stored.

### **GDA Start**

**You should always open the prototype through the GDA Start stack.**

The GDA Start stack contains a brief introduction that describes the focus of the prototype, lists the names of the individuals that worked on its development and lists the academic departments and outside organizations that financially supported it.

The GDA Start stack also includes the **Stack Builder** card that delivers to you the database accessing tools. There is one simple rule for using the software: **you must choose what it is that you would like to see**. The Stack Builder card will allow you to access all the work done by a certain designer, all the work created in a specific location or done in a specific medium. The Stack Builder card also includes a query tool that will allow you to search the database for more specific information. You will be, in effect, building a stack of items that you can save for later viewing.

The last component of the GDA Start stack is the **Classification Tree**. It is a structure developed by Professor Roger Remington that organizes the images in the prototype database according to the type of work that they represent. There are two major branches to this tree. **Design Work** includes all the design applications and **Designers Archive** includes things like photographs of the designers, their surroundings and their correspondences.


### **Data Cards**

Each video image is linked to a text frame, called a **Data Card**, that lists the name of the designer, the country where the work was created, the medium used, the date of the piece, its title and, occasionally, a comment. There are 806 Data Cards that document the 806 video images in the Prototype 2.0 database.

### **Reference**

The **Reference** stack includes biographies of the designers included in the database, definitions of terms used in those biographies and bibliographic information. It also has extensive cross-referencing notations which can direct the user to related material and topics within or outside the database.



 **Graphic Design Archive**

**Prototype 2.0**

<b>Stack Builder</b>	Click the <b>Stack Builder</b> button to begin.
<b>Introduction</b>	Click the <b>Introduction</b> button to find out more about this prototype.
<b>Set Player</b>	If you are using a videodisc player, click the <b>Set Player</b> button to set to the correct player type.

The Graphic Design Archive is an electronic desktop museum of the history of graphic design.

The purpose of the project is to provide an expansive database of text and image frames that can be accessed interactively for educational and informational purposes.

Graphic Design Archive ©1989  
All Rights Reserved

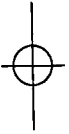
**quit**




## ***Opening Prototype 2.0***

Double click on the GDA Start stack icon to open the prototype.  
For a moment you will see on the screen a card composed  
of some of the noteworthy symbols and icons from the  
history of graphic design.

The icon and symbol card will open up to reveal  
the first interactive card of the prototype.





 **Graphic Design Archive**

**Prototype 2.0**

**Current Player:**  
None

**Set Player**  
None  
Pioneer 4200  
Pioneer 6000A  
Hitachi 9550  
Sony 2000  
Sony 1500  
Sony 1000  
Cancel

Click the **Stack Builder** button to begin.

Click the **Introduction** button to find out more about this prototype.


If you are using a videodisc player, click the **Set Player** button to set to the correct player type.

The **Graphic Design Archive** is an electronic desktop museum of the history of graphic design.

The purpose of the project is to provide an expansive database of text and image frames that can be accessed interactively for educational and informational purposes.

Graphic Design Archive © 1989  
All Rights Reserved

**QUIT**

 **Graphic Design Archive**

**Prototype 2.0**

**Setting Disc Player To:**  
Sony 1000

**Set Player**

Click the **Stack Builder** button to begin.

Click the **Introduction** button to find out more about this prototype.

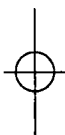
If you are using a videodisc player, click the **Set Player** button to set to the correct player type.

The **Graphic Design Archive** is an electronic desktop museum of the history of graphic design.

The purpose of the project is to provide an expansive database of text and image frames that can be accessed interactively for educational and informational purposes.

Graphic Design Archive © 1989  
All Rights Reserved

**QUIT**





## ***Set Player***

If you will be using a videodisc player with the prototype set the correct player type by following the steps below.

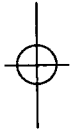
If you will not be using a videodisc player make sure that the player type is set to **None**.

1. Click on the **Set Player** button.
2. Select your player type by clicking on its name in the menu that appears under the Set Player button.

**Notice** that the last menu option is **Cancel**. If you click on the Set Player button unintentionally, click on **Cancel** to restore this card to its original state.

Once you have selected a player type the menu will disappear, and the field above the Set Player button will indicate that your choice is being set.

Hereafter when you open the prototype you need not repeat these steps unless you have changed videodisc players.







**Graphic Design Archive**

**Prototype 2.0**

**Stack Builder** | Click the Stack Builder button to begin.

**Introduction** | Click the Introduction button to find out more about this prototype.

**Set Player** | If you are using a videodisc player, click the Set Player button to set to the correct player type.

The Graphic Design Archive is an electronic desktop museum of the history of graphic design.

The purpose of the project is to provide an expansive database of text and image frames that can be accessed interactively for educational and informational purposes.

Graphic Design Archive © 1989  
All Rights Reserved

**QUIT**

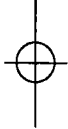
**Graphic Design Archive**

**Introduction**

The focus of the prototype is the following group of designers who worked between 1930 and 1955:

Saul Bass	E. McKnight Kauffer
Herbert Bayer	Oyergy Kepes
Lester Beall	Leo Lionni
Joseph Binder	Alvin Lustig
Alexey Brodovitch	Herbert Matter
Will Burtin	Paul Rand
A. M. Cassandre	Ladislav Sutnar
Charles Eames	Bradbury Thompson
William Golden	Jan Tschichold

**Return** **QUIT**



**Graphic Design Archive**

**Introduction**

The Graphic Design Archive is being developed by:  
Rochester Institute of Technology  
One Lomb Memorial Drive  
Rochester, New York U.S.A.

It is a cooperative effort of:  
Department of Graphic Design  
College of Fine & Applied Arts  
American Video Institute  
Department of Applied Computer Studies  
College of Liberal Arts

**Return** **QUIT**



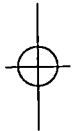
## **Introduction**

Click on the **Introduction** button and the screen will open to a card that describes the focus of the prototype and lists the names of the twenty designers included in the database.

There are more cards in the Introduction section following this first one.

1. Click on the arrow buttons in the lower right corner of each card to browse through them. A black arrow indicates that there are more cards in the indicated direction; a gray arrow indicates that you're at the beginning or the end of the Introduction section.

2. The **Return** button will return you to the first interactive card of the prototype. Return there to proceed with this tutorial or to start using the prototype on your own.





**Graphic Design Archive**

**Prototype 2.0**

**Stack Builder** Click the Stack Builder button to begin.

**Introduction** Click the Introduction button to find out more about this prototype.

**Set Player** If you are using a videodisc player, click the Set Player button to set to the correct player type.

The Graphic Design Archive is an electronic desktop museum of the history of graphic design.

The purpose of the project is to provide an expansive database of text and image frames that can be accessed interactively for educational and informational purposes.

Graphic Design Archive © 1989  
All Rights Reserved

**QUIT**

**Stack Builder**

**The Stack** **Images**

**Designer**

**Location**

**Medium**

**File**

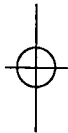
**Tree**

**Query**

**Search** **Save** **Remove**

**Ref.** **Cards**

**START** **QUIT**



**Stack Builder**

**The Stack** **Images**

**Designer** Bass, Saul  
Bayer, Herbert  
**Location** Beall, Lester  
Binder, Joseph  
**Medium** Brodevitch, Alexey  
**File** Burtin, Will  
**Tree** Cassandre, A. Mouron  
Eames, Charles  
Golden, William  
Gutsti, George  
Kauffer, E. McKnight  
Kepes, Gyergy

**Search** **Save** **Remove**

**Ref.** **Cards**

**START** **QUIT**



## **Stack Builder**

Click the **Stack Builder** button on the first interactive card to go to the Stack Builder card.

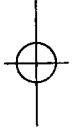
On the left of the Stack Builder card is a menu of the data accessing tools. On the right is a rectangle labelled **The Stack**. All of the information gathered by the accessing tools is put into **The Stack** in the form of an itemized list. At first, **The Stack** and the buttons below it are outlined in gray indicating that **The Stack** is empty.

The first three tools on the menu are the simplest for accessing the database. They represent three different categories of information: the names of the designers included in the database, the locations in which the designers worked and the media they used to create their work.

1. Click on the **Designer** button. A field will appear in the center of the Stack Builder card. This field contains a list of the names of the twenty designers included in the database. You can scroll through this list by using the arrow buttons at the right side of the field.

2. Select one of the names by clicking on it.

You can use this same process to select items by clicking on the **Location** or **Medium** buttons.





**Stack Builder**

The Stack Images

Designer  
Location  
Medium  
File  
Tree  
Query

Bayer, Herbert 42

Scan <> Save Remove

Ref. Cards

START QUIT

**Stack Builder**

The Stack Images

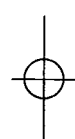
Designer  
Location  
Medium  
File  
Tree  
Query

Bayer, Herbert 42

Scan <> Save Remove

Ref. Cards

START QUIT



**Stack Builder**

The Stack Images

Designer  
Location  
Medium  
File  
Tree  
Query

Bayer, Herbert 42

Scan <> Save Remove

Ref. Cards

START QUIT



## **Stack Builder**

You should now have at least one item in The Stack. The number that is to the right of each item in The Stack indicates the number of images on the Archive videodisc that correspond to that item.

Notice that The Stack and the buttons below it are outlined in black indicating that The Stack is no longer empty.

**3.** Click once on an item in The Stack to select it. The item will become highlighted.

Double click anywhere in The Stack to select all the items.

Once you have selected an item in The Stack and it is highlighted, you have four options:

**Scan:** If you are using a videodisc player, you can click on the Scan button and the images that are associated with the selected item will be shown to you, at two second intervals, on the video screen.

Click again anywhere on the Stack Builder card to stop the scan.

The arrow buttons to the right of the Scan button will allow you to proceed one at a time through the video images.


**Save:** Information about saving a file can be found on page 15 of this guide.

**Remove:** Click on the Remove button to remove items from The Stack.

**Cards:** The Cards button will take you to the Data Cards that describe the video images that are associated with the selected item.

**4.** With an item in The Stack highlighted, click on the **Cards** button.



 **Data Card**


◀ 1 ▶

Designer | **Bayer, Herbert**  
 Date | **1924**  
 Location | **Germany**  
 Medium | **Dimensional form**

Title Sketches for dimensional designs used for promotional application  
 Comment

---

Ref. Return  
 START QUIT

 **Data Card**

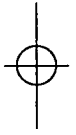
◀ 42 ▶

Designer | **Bayer, Herbert**  
 Date | **1984**  
 Location | **USA**  
 Medium | **Printed image, lithography**

Title Page from Bauhaus book devoted to Bayer  
 Comment From BAUHAUS by Vingler

---

Ref. Return  
 START QUIT





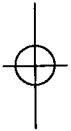
## **Data Cards**

Once you have arrived on the first Data Card, notice the **scrollbar**, the black bar that runs horizontally across the top of the card. There are **arrows** at both ends of the scrollbar, and inside it there is a white box that contains a **number**. The number refers to the position of this card and corresponding video image in relation to total number of cards and images in the selected item. The card to the left, for instance, represents the first image of the 42 images that were done by Herbert Bayer.

**1.** Click on the scrollbar **arrows** to proceed one at a time through the cards and the corresponding video images.

**2.** Click on the **white numbered box**, hold the mouse down and drag the box across the bar. Notice that the number changes. When you let the mouse up, the data card and the video image will also change.

**3.** Click on the **Ref.** button in the lower right corner of the Data Card to find out more about the designer that is listed on the card.







**Reference**

**Bayer, Herbert** **See Also**

Herbert Bayer(b.1900-d.1985) was born in Austria and apprenticed there in an architectural studio before becoming a student at the Bauhaus in Weimar in 1921. He taught advertising and typography at the Bauhaus in Dessau between 1925 to 1928. He discouraged traditional typographic ornament and layout, instead advocating the use of sans serif type and all lower-case letters. He worked to integrate typography and photography. He immigrated to the United States in 1938. Although Bayer wanted to be remembered as a painter, he is known primarily for his publication design, photography, and exhibition design for American clients such as Container Corporation of America, ARCO, Fortune magazine and others.

**Bauhaus**  
**Container Corp. Of America**  
**Constructivism**  
**International Style**  
**Functionalism**

**Index** **Notes** **Return**

**START** **QUIT**

**Reference**

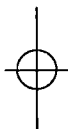
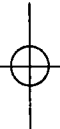
**Bayer, Herbert** **See Also**

Herbert Bayer(b.1900-d.1985) was born in Austria and apprenticed there in an architectural studio before becoming a student at the Bauhaus in Weimar in 1921. He taught advertising and typography at the Bauhaus in Dessau between 1925 to 1928. He discouraged traditional typographic ornament and layout, instead advocating the use of sans serif type and all lower-case letters. He worked to integrate typography and photography. He immigrated to the United States in 1938. Although Bayer wanted to be remembered as a painter, he is known primarily for his publication design, photography, and exhibition design for American clients such as Container Corporation of America, ARCO, Fortune magazine and others.

American School  
 Art Nouveau  
 Arts and Crafts Movement  
 Bass, Saul  
 Bauhaus  
 Bayer, Herbert  
 Beall, Lester  
 Binder, Joseph  
 Bradley, William  
 Bradevitch, Alexey  
 Burtin, Will  
 Cassandre, A. Meuron  
 Constructivism  
 Container Corp. of America  
 Cubism  
 de Stijl

**Index** **Notes** **Return**

**START** **QUIT**



**Reference**

**Bayer, Herbert** **See Also**

Herbert Bayer(b.1900-d.1985) was born in Austria and apprenticed there in an architectural studio before becoming a student at the Bauhaus in Weimar in 1921. He taught advertising and typography at the Bauhaus in Dessau between 1925 to 1928. He discouraged traditional typographic ornament and layout, instead advocating the use of sans serif type and all lower-case letters. He worked to integrate typography and photography. He immigrated to the United States in 1938. Although Bayer wanted to be remembered as a painter, he is known primarily for his publication design, photography, and exhibition design for American clients such as Container Corporation of America, ARCO, Fortune magazine and others.

Ades, Dawn,  
 "The Twentieth Century Poster:  
 Design of the Avant Garde"  
 (New York, Minneapolis: Walker Art  
 Center and Abbeville Press, 1984).

Cohen, Arthur,  
 "Herbert Bayer: Painter Designer  
 Architect"  
 (New York: Reinhold, 1967).

**Index** **Notes** **Return**

**START** **QUIT**



## **Reference Stack**

Most of the cards in the **Reference** stack contain biographical information about the designers in the database. There are several additional cards that contain definitions of some of the terms and movements mentioned in the biographies.

On the right side of almost every card, under the **See Also** heading, in bold type, there is a list of terms for cross-referencing.

1. Click on one of these **See Also** terms to get to a Reference card by that name.
2. Click on the **Index** button in the lower right corner of the card to bring up a scrolling list of all the topics included in the Reference stack. Click on any one of those topics to get to a Reference card that describes it.
3. Click on the **Notes** button to bring up a field of bibliographic information.
4. Click on the **Return** button in the lower right corner of the Reference card, to get back to the Data Cards.
5. Click on the **Return** button on the Data Cards to get back to the Stack Builder card.





**Stack Builder**

The Stack Images

Designer	Bayer, Herbert	42
----------	----------------	----

Location  
Medium  
File  
Tree

Query

Scan Save Remove

Ref. Cards

START QUIT

**Stack Builder**

Query Constraints

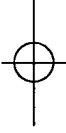
Designer	
Location	
Medium	
File	
Tree	

Cancel Query Search Remove

Scan Save Remove

Ref. Cards

START QUIT





## **Query**

The Query tool allows you to be more specific in your search for information. It was designed to combine the unique accessing power of the Classification Tree with the tools that access information categorically (Designer, Location and Medium.) The Query tool is the last item in the menu of database accessing tools on the left of the Stack Builder card.

1. Click on the **Query** button. Five boxes will appear to the right of the menu under the heading **Query Constraints**.

Notice that the Stack and the buttons below it are now gray. While you are involved in the Query function you will not have access to The Stack items or buttons.

Notice the **Cancel** button that has replaced the Query button. You can cancel the Query function at any time and the Stack Builder card will be restored to its previous state.





**Stock Builder**

Query Constraints

Designer	Golden, William
Location	Gristi, George
Medium	Kaufer, E. McKnight
File	Kepes, Gyergy
Tree	Lionni, Leo
Cancel	Lustig, Alvin
	<b>Matter, Herbert</b>
	Nitsche, Eric
	Rand, Paul
	Sutnar, Ladislav
	Thompson, Bradbury
	Tschichold, Jan

Search Save Remove

Ref. Cards

START QUIT

**Stock Builder**

Query Constraints

Designer	Canada
Location	Czechoslovakia
Medium	England
File	France
Tree	Germany
Cancel	<b>Switzerland</b>
	USA

Search Save Remove

Ref. Cards

START QUIT

**Stock Builder**

Query Constraints

Designer	Matter, Herbert	50
Location	Switzerland	17
Medium		
File		
Tree		
Cancel	Query Search	Remove

Search Save Remove

Ref. Cards

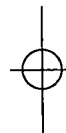
START QUIT



## Query


The series of steps on the next three pages will guide you through an example of a database search using the Query tool. The goal of the search is to find all the posters that were designed by Herbert Matter to promote tourism in Switzerland.

1. Click on the **Designer** button. The scrolling field of designers' names will appear in the center of the card. Select "Matter, Herbert." The scrolling field will disappear and "Matter, Herbert" will fill the rectangle directly to the right of the designer button.
2. Click on the **Location** button and select "Switzerland" from the scrolling field.
3. Click on the **Tree** button to go to the first card of the Classification Tree. The next page will direct you in navigating to the branch of the tree that represents tourism posters. You will be returning to the Stack Builder card with information gathered from the Classification Tree.





**Classification Tree**

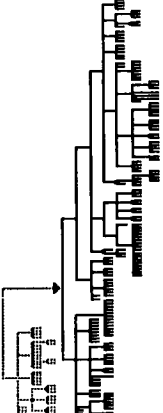


**Classification Tree**  
804 Images

**Design Work**  
**Designers Archive**

Scan <> Return  
START QUIT

**Classification Tree**

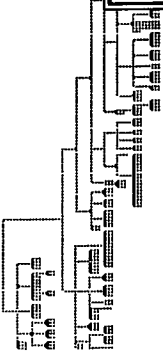


**Classification Tree**  
732 Images

**Design Work** | **Applied Arts**  
**Fine Arts**

Scan <> Return  
START QUIT

**Classification Tree**



**Use 30 Images for a query constraint?**

**Yes** **No** **Cancel**

**Tourism**  
30 Images

Draw/Print Impressions 2D  
Information Graphics  
Publication Design  
Poster

Scan <> Return  
START QUIT



## **Classification Tree**

The Classification Tree is a structure that organizes the images in the database according to the type of work that they represent.

The map on the left side the card illustrates the tree structure and indicates, with a flashing arrow, your current position within the tree.

The black bar in the center of the card states the name of the current tree branch. Underneath the black bar you will find the number of images that are classified under that branch of the tree.

To the right of the black bar, in bold type, are the names of the tree branches that you can access from your current position.

Notice that the map illustrates this list of accessible branches.

1. Click on the **Design Work** branch to the right of the black bar.

Notice the map on the Design Work card.

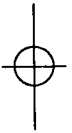
The gray area indicates the branches of the tree that can not be accessed from the Design Work branch. The black areas represent the branches that are currently accessible.

Notice that the name of the previous branch (Classification Tree) is now located above the black bar in the center of the card.

As you navigate through the tree your path will be recorded in a list above this bar. Click on the branch names in this list to move backwards through the tree.

Notice the **Scan** button in the lower right corner of the card.

At any time during your navigation through the tree you can stop and scan the images that are included in the current branch.



2. To get to the branch of the tree that represents tourism posters click on the branches indicated below:

**Design Work**  
**Applied Arts**  
**Draw/Print Impressions 2D**  
**Information Graphics**  
**Publication Design**  
**Poster**  
**Tourism**

The **Tourism** branch includes 30 images. Notice that the flashing map marker indicates that you are now at the end this particular branch of the tree.

3. Click the **Return** button in the lower right corner of the Tourism card to return to the Stack Builder card.

A dialogue box will appear and ask you if you want to use the 30 Tourism images as a Query Constraint.  
 Click **Yes**.





**Stack Builder**

**Query Constraints**

Designer	Matter, Herbert	50
Location	Switzerland	17
Medium		
File		
Tree	Tourism	30

Cancel Query Search Remove

Scan Save Remove

Ref. Cards

START QUIT

**Stack Builder**

**Query Constraints**

Designer	The result is 7 images!
Location	
Medium	
File	
Tree	

Cancel Add To Stack

Scan Save Remove

Ref. Cards

START QUIT

**Stack Builder**

**Name the result.**

Matter Swiss Posters

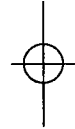
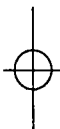
OK Cancel

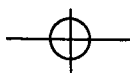
Cancel Add To Stack

Scan Save Remove

Ref. Cards

START QUIT





## **Query**

Three of the five query constraint boxes on the Stack Builder card should now contain information pertinent to your search for the Swiss tourism posters designed by Herbert Matter. You are now ready to instruct the computer to search the database.

1. Click on the **Query Search** Button under the Query Constraint boxes. The computer will need a moment to conduct the search, then a field will appear on the screen with the search results.

**"The result is 7 Images!"**

To gain access to these 7 images and their Data Cards you will have to add them to The Stack.

3. Click on the **Add to Stack** button that has replaced the Query Search button. A HyperCard dialogue box will appear.

**"Name the result."**

Type in a name that is less than 22 characters, then click **OK**.





**Stack Builder**

The Stack Images

Designer  
Location  
Medium  
File  
Tree  
Query

Bayer, Herbert	42
Matter Swiss Posters	7

Scan <> Save Remove

Ref. Cards

START QUIT

**Stack Builder**

The Stack Images

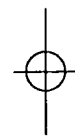
Designer  
Location  
Medium  
File  
Tree  
Query

Bayer, Herbert	42
Matter Swiss Posters	7

Scan <> Save Remove

Ref. Cards

START QUIT



**Stack Builder**

The Stack Images

Designer  
Location  
Medium  
File  
Tree  
Query

Bayer, Herbert	42
Matter Swiss Posters	7

Scan <> Save Remove

Ref. Cards

START QUIT



## **Stack Builder**

Once you have added the result of a Query Search to The Stack you can treat it like any other item in The Stack.

Click on the new item to select it.

1. Click on the **Scan** button to quickly view the images.  
Click again anywhere on the Stack Builder card to stop the scan.
2. Click on the **Save** button to save the item as a file on a floppy disk. See page 15 for more information about saving a file.
3. Click on the **Remove** button to remove the item from The Stack.
4. Click on the **Cards** button to go to the Data Cards that describe the images.





**Stack Builder**

The Stack Images

Designer  
Location  
Medium  
File  
Tree  
Query

Bayer, Herbert	42
Matter Swiss Posters	7

Scan <> Save Remove

Ref. Cards

START QUIT

**Stack Builder**

The Stack Images

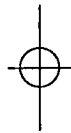
Designer  
Location  
Medium  
File  
Tree  
Query

Bayer, Herbert	42
Matter Swiss Posters	7

Scan <> Save Remove

Ref. Cards

START QUIT



**Stack Builder**

The Stack Images

Designer  
Location  
Medium  
File  
Tree  
Query

Bayer, Herbert	42
Matter Swiss Posters	7

Save Stack as unified list (no dups), or as is...

Union As is cancel

Scan <> Save Remove

Ref. Cards

START QUIT



## **Saving A File**

The items that you select and put into The Stack can be saved as a file on a floppy disk so that you can access them again at a later date. You can save the items one at a time as separate files or (if you have collected more than one item) you can save the entire Stack as a file.

### **To save a single item in The Stack:**

1. Click on the item to select it.
2. Click the **Save** button below The Stack.
3. The traditional Macintosh dialogue box will appear and allow you to type in the name of the file and indicate the name of the disk that you will be using to store the file.

### **To save all the items in The Stack:**

1. Double click anywhere in The Stack to select all the items.
2. Click the **Save** button below The Stack.
3. A HyperCard dialogue box presents you with three choices:

**Save the file as a unified list.** The Stack is in the form of an itemized list. Every item has a unique name and can be accessed separately. However, there is a possibility that a particular image may be included in more than one item.

Saving the file as a unified list would compile the separate stack items into one large list and remove any duplicate images.

Click on **Union**. The HyperCard dialogue box will disappear and the traditional Macintosh dialogue box will replace it on the screen. Type in a name for the file and click **Save**.

**Save the file as is.** This action would save The Stack as a file without changing the itemized format.

Click on **As Is** in the HyperCard dialogue box. The traditional Macintosh dialogue box will appear. Type in a name for the file and click **Save**.

**Cancel** will allow you to cancel the save function and restore the Stack Builder card to its previous state.



## ***Appendix B: Sorting and Programming Documentation***



on mouseUp

sortByDate  
sortByMedium  
sortByLocation  
sortByDesigner  
sortByTaxonomy  
TaxonomyList

sortByDate  
sortByTaxonomy  
sortByLocation  
sortByDesigner  
sortByMedium  
MediumList

sortByDate  
sortByTaxonomy  
sortByMedium  
sortByDesigner  
sortByLocation  
LocationList

--sortByTaxonomy  
--sortByMedium  
--sortByLocation  
--sortByDesigner  
--sortByDate  
--DateList

sortByDate  
sortByTaxonomy  
sortByMedium  
sortByLocation  
sortByDesigner  
DesignerList

sortByDate  
sortByDesigner  
end mouseUp

```
--*****
```

```
on sortByDate
```

```
  go card 1 of stack Cards
```

```
  sort numeric by first word of field "date"
```

```
end sortByDate
```

```
on sortByTaxonomy
```

```
  go card 1 of stack Cards
```

```
  sort by word 1 of field "taxonomy"
```

```
end sortByTaxonomy
```

```
on sortByMedium
```

```
  go card 1 of stack Cards
```

```
  sort by field "medium"
```

```
end sortByMedium
```

```
on sortByLocation
```

```
  go card 1 of stack Cards
```

```
  sort by field "location"
```

```
end sortByLocation
```

```
on sortByDesigner
```

```
  go card 1 of stack Cards
```

```
  sort by field "designer"
```

```
end sortByDesigner
```

```
on DesignerList
```

```
  Global theIDlist
```

```
  Global designlist
```

```
  set cursor to busy
```

```
  set lockscreen to true
```

```
  go card 1 of stack Cards
```

```
  makedesignerslist
```

```
  go to card "designer" of stack newlists
```

```
  put empty into bg field "contentList"
```

```
  put empty into bg field "IDlist"
```

```
  put theIDlist into bg field "IDlist"
```

```
  put designlist into bg field "contentList"
```

```
end DesignerList
```

```
on LocationList
```

```
  Global theIDlist
```

```
  Global locationlist
```

```
  set cursor to busy
```

```
  set lockscreen to true
```

```
  go card 1 of stack Cards
```

```
  makelocationslist
```

```
  go to card "location" of stack newlists
```

```
  put empty into bg field "contentList"
```

```
  put empty into bg field "IDlist"
```

```
  put theIDlist into bg field "IDlist"
```

```
  put locationlist into bg field "contentList"
```

```
end LocationList
```

```
on MediumList
```

```
  Global theIDlist
```

```
  Global mediumlist
```

```
  set cursor to busy
```

```
  set lockscreen to true
```

```
  go card 1 of stack Cards
```

```
  makemediumslist
```

```
  go to card "medium" of stack newlists
```

```
  put empty into bg field "contentList"
```

```
  put empty into bg field "IDlist"
```

```

    put theIDlist into bg field "IDlist"
    put mediumlist into bg field "contentList"
end MediumList

```

```

on TaxonomyList
  Global theIDlist
  Global taxonomylist
  set cursor to busy
  set lockscreen to true
  go card 1 of stack Cards
  maketaxonomylist
  go to card "taxonomy" of stack newlists
  put empty into bg field "contentList"
  put empty into bg field "IDlist"
  put theIDlist into bg field "IDlist"
  put taxonomylist into bg field "contentList"
end TaxonomyList

```

```

on DateList
  Global theIDlist
  Global datelist
  set cursor to busy
  set lockscreen to true
  makedateslist
  go to card "date" of stack newlists
  put empty into bg field "contentList"
  put empty into bg field "IDlist"
  put theIDlist into bg field "IDlist"
  put datelist into bg field "contentList"
end DateList

```

--ruth's script to make card id lists for GDA

```

on makeDesignersList
  Global theIDlist
  Global designlist
  put "$=@" into current
  go to first card
  put empty into designlist
  put empty into theIDlist
  Repeat with count = 1 to number of cards
    if field designer is not current then
      put field designer into current
      put current & return after designlist
    if count is not 1 then
      put "," after theIDlist
    End if
  End if
  Put last word of the ID of this card into line count of theIDlist
  go to next card
  End repeat
End makeDesignersList

```

```

on makeTaxonomyList
  Global theIDlist, taxonomylist
  put "$=@" into current
  go to first card
  put empty into taxonomylist
  put empty into theIDlist
  Repeat with count = 1 to number of cards
    if word 1 of field taxonomy is not current then
      put word 1 of field taxonomy into current
      put current & return after taxonomylist
    if count is not 1 then
      put "," after theIDlist

```

```
    End if
  End if
  Put last word of the ID of this card into line count of theIDlist
  go to next card
End repeat
End makeTaxonomyList

on makeLocationsList
  Global theIDlist
  Global locationlist
  put "$=@" into current
  go to first card
  put empty into locationlist
  put empty into theIDlist
  Repeat with count = 1 to number of cards
    if field location is not current then
      put field location into current
      put current & return after locationlist
      if count is not 1 then
        put "," after theIDlist
      End if
    End if
    Put last word of the ID of this card into line count of theIDlist
    go to next card
  End repeat
End makeLocationsList

on makeMediumsList
  Global theIDlist
  Global mediumlist
  put "$=@" into current
  go to first card
  put empty into mediumlist
  put empty into theIDlist
  Repeat with count = 1 to number of cards
    if field medium is not current then
      put field medium into current
      put current & return after mediumlist
      if count is not 1 then
        put "," after theIDlist
      End if
    End if
    Put last word of the ID of this card into line count of theIDlist
    go to next card
  End repeat
End makemediumslist

on makedateslist
  Global theIDlist
  Global datelist
  put "$=@" into current
  go to first card of stack cards
  put empty into datelist
  put empty into theIDlist
  Repeat with count = 1 to number of cards
    if first word of field date is not current then
      put first word of field date into current
      put current & return after datelist
      if count is not 1 then
        put "," after theIDlist
      End if
    End if
    Put last word of the ID of this card into line count of theIDlist
    go to next card
```

3/28/89 9:08 PM

Script of stack Tools/Lists:gda tools

Page 4

End repeat  
End makedateslist

```

on openStack
  hide menuBar
  global initialized, place
  if initialized is true then
    exit openStack
  else
    --Initialized directs hypercard to this stack (from the card stack)
    --(or from the reference stack) on first opening.
    --This insures that all necessary global variables are declared.
    --The user should always start from this stack.

    --Install these handlers into each interactive video stack.
    --These globals must be initially declared or video won't work!

    global SPortGlobals --serial port stuff
    global typeOfVideo,lastVideoFrame,blankNextVideo,videoSpeed,videoMode
    global playerType
    put line 2 of card field "current player"--
    of card "frontEnd" into playerType
    setVideoPlayer playerType --set the correct video drivers
    put true into initialized
    put "stackBuilder" into place
    wait 140 ticks
    visual iris open slow
    go to card frontEnd
  end if
end openStack

```

```

function ClickLine
  -- Script written by Luc Perron in the spring of 1988 at RIT
  -- This function returns the line number of where the mouse has been
  -- clicked in a field (Scrolling or not).
  -- Ex: put ClickLine() into LineNumber
  put item 2 of the rect of the Target into Top
  put item 2 of the clickLoc into Y
  if the style of the Target is "scrolling" then
    return (Y - Top + scroll of the Target) div--
    (textHeight of the Target) + 1
  else
    return (Y - Top) div (textHeight of the Target) + 1
  end if
end ClickLine

```

```

--This function returns the union of the two lists.
--This script is the result of the work of the spring 1988
--MicroComputer Control class taught by Steve Kurtz at RIT
function PlusList listOne, listTwo
  repeat with counter = 1 to number of lines in listOne
    if line counter of listOne is not in listTwo then
      put return & line counter of listOne after listTwo
    end if
  end repeat
  return listTwo
end PlusList

```

```

--This function finds the intersection of the 2 lists
--This script is the result of the work of the spring 1988
--MicroComputer Control class taught by Steve Kurtz at RIT
function InterList listOne, listTwo
  put empty into difference
  repeat with counter = 1 to number of lines in listOne
    if ((line counter of listOne)&return) is in listTwo then
      put line counter of listOne & return after difference
    end if
  end if

```

```

end repeat
if last character of difference is return then
  delete last character of difference
end if
return difference
end InterList

on sort
-- All cards in the three GDA stacks are in a specific order!
answer "It would not be wise to sort this stack" with "OK"
exit sort
end sort

on doMenu command
if command = "quit hyperCard" then
  set lockscreen to true

  -- Clean up the "Stack Builder" card to insure a fresh start!
  go cd stackBuilder
  put empty into card field theStack
  put empty into card field images
  hide cd field theStack
  hide cd field images
  put empty into card field "itemHolder"
  hide bg button "cancel"
  hide bg button "Query Search"
  hide bg button "Remove"

  -- Reset & hide the query fields and buttons
  repeat with z = 1 to 10
    if (z = 1) or (z = 2) or (z = 3) or (z = 4) or (z = 5) then
      put "no entry" into cd field ("list" & z)
      hide cd button z
      put empty into bg field z
      hide bg field z
    else
      put empty into bg field z
      hide bg field z
    end if
  end repeat

  --Hide these buttons because on first opening the Stack is empty.
  repeat with b = 7 to 12
    hide card button b
  end repeat
  show cd button "ref."

  -- These buttons provide visual feedback
  -- when selecting items from the Stack.
  repeat with x = 17 to 28
    set hilite of cd button x to false
  end repeat

else
  if command = "Delete Card" then
    answer "Please don't delete any cards!" with "OK"
    exit doMenu
  end if
end if
pass doMenu
end doMenu

```

```
on mouseUp
  global cardName, theNum, place
  -- cardName directs hyperCard to the correct card holding
  -- the contentList and the idList (card ids and frame numbers.)

  -- theNum represents both the line in field "theStack"
  -- that has been selected and the number of the corresponding item
  -- in hidden card field "itemHolder."

  if theNum is not empty then send mouseUp to cd stackBuilder
  -- This unselects anything in the stack that is hilited
  -- and puts empty into theNum.

  if hilite of target is true then
    -- Provides a way out of this procedure
    hide cd field theSelection
    hide cd button cover
    set hilite of target to false
    exit mouseUp
  else
    put "designer" into cardName
    set cursor to 4
    repeat with x = 1 to 6 -- hilite the correct button
      if the short name of bg button x is "designer" then
        set hilite of bg button x to true
      else
        set hilite of bg button x to false
      end if
    end repeat

    get bg field contentList of card cardName
    put it into card field "theSelection"
    set scroll of card field "theSelection" to 0
    show card button cover
    show card field "theSelection"
  end if
end mouseUp
```



4/4/89 6:28

Script of bkgnd button id 95 = "Location"

```
on mouseUp
  global cardName, theNum
  -- cardName directs hyperCard to the correct card holding
  -- the contentList and the idList (card ids and frame numbers.)

  -- theNum represents both the line in field "theStack"
  -- that has been selected and the number of the corresponding item
  -- in hidden card field "itemHolder."

  if theNum is not empty then send mouseUp to cd stackBuilder
  -- This unselects anything in the stack that is hilited
  -- and puts empty into theNum.

  if hilite of target is true then
    -- Provides a way out of this procedure
    hide cd field theSelection
    hide cd button cover
    set hilite of target to false
    exit mouseUp
  else
    put "location" into cardName
    repeat with x = 1 to 6
      if the short name of bg button x is "location" then
        set hilite of bg button x to true
      else
        set hilite of bg button x to false
      end if
    end repeat

    get bg field contentList of card cardName
    put it into card field "theSelection"
    set scroll of card field "theSelection" to 0
    show card button cover
    show card field "theSelection"
  end if
end mouseUp
```

4/4/89 6:29

Script of bkgnd button id 94 = "Medium"

```
on mouseUp
  global cardName, theNum
  -- cardName directs hyperCard to the correct card holding
  -- the contentList and the idList (card ids and frame numbers.)

  -- theNum represents both the line in field "theStack"
  -- that has been selected and the number of the corresponding item
  -- in hidden card field "itemHolder."

  if theNum is not empty then send mouseUp to cd stackBuilder
  -- This unselects anything in the stack that is hilited
  -- and puts empty into theNum.

  if hilite of target is true then
    -- Provides a way out of this procedure
    hide cd field theSelection
    hide cd button cover
    set hilite of target to false
    exit mouseUp
  else
    put "medium" into cardName
    repeat with x = 1 to 6
      if the short name of bg button x is "medium" then
        set hilite of bg button x to true
      else
        set hilite of bg button x to false
      end if
    end repeat

    get bg field contentList of card cardName
    put it into card field "theSelection"
    set scroll of card field "theSelection" to 0
    show card button cover
    show card field "theSelection"
  end if
end mouseUp
```

```

on mouseUp
  global Place, theNum

  -- Place is either "stackBuilder" or "query"
  -- This makes sure that the info.
  -- gets put into the correct containers (hidden fields).

  -- theNum represents both the line in field "theStack"
  -- that has been selected and the number of the corresponding item
  -- in hidden card field "itemHolder."

  if theNum is not empty then send mouseUp to cd stackBuilder
  -- This unselects anything in the stack that is hilited
  -- and puts empty into theNum.

  -- Hilite the correct button
  repeat with x = 1 to 6
    if (the short name of bg button x is "file") then
      set hilite of bg button x to true
    else
      set hilite of bg button x to false
    end if
  end repeat

  --PathName is an xfcn written by Andrew Gilmartin at Brown University
  --It brings up the traditional Apple window for file saving.
  get PathName("TEXT")
  if it is empty then --if operation is cancelled....
    set hilite of target to false
    choose browse tool
    exit mouseUp
  else
    --See script of card button "save" to find out how files are saved.
    put it into fileName
    open file fileName
    read from file fileName until "#"
    delete last char of it
    put it into nameHolder
    read from file fileName until "#"
    delete last char of it
    put it into imageHolder
    read from file fileName until empty
    delete last char of it
    put it into savedItems
    if last line of savedItems is empty then
      delete last line of savedItems
    end if

    if Place is "stackbuilder" then
      put nameHolder & return after cd field "theStack" --name of file
      put imageHolder & return after cd field "images" --no. of images
      put savedItems & "," & return after card field itemHolder
      --card ids and frame numbers put after hidden field "itemHolder"
      --Make sure all buttons and fields are showing...
      show cd field theStack
      show cd field images
      repeat with x = 7 to 12
        show cd button x
      end repeat
    else
      if Place is "query" then
        put fileName into FieldEntry
        --FileName includes the pathName

```

```
--This script gets the short name of the file
repeat with x = 1 to number of characters in FieldEntry
  if ":" is in FieldEntry then
    delete first character of FieldEntry
  else
    exit repeat
  end if
end repeat

put FieldEntry into bg field "file" --Name of the file
put imageHolder into bg field 9 --total number of images
put savedItems into cd field "list4" --card ids & frame numbers
--The query function uses 5 hidden fields
--to hold card id and frame numbers.
end if
end if

close file fileName
set hilite of target to false
choose browse tool
end if
end mouseUp
```

```
on mouseUp
  global Place, theNum
  put "query" into place

  -- Place is either "stackBuilder" or "query"
  -- This makes sure that the info.
  -- gets put into the correct containers (hidden fields).

  -- theNum represents both the line in field "theStack"
  -- that has been selected and the number of the corresponding item
  -- in hidden card field "itemHolder."

  if theNum is not empty then send mouseUp to cd stackBuilder
  -- This unselects anything in the stack that is hilited
  -- and puts empty into theNum.

  repeat with x = 1 to 6 --hilite the correct button...
    if (the short name of bg button x is "query") then
      set hilite of bg button x to true
    else
      set hilite of bg button x to false
    end if
  end repeat

  -- Hide all options other than those necessary for the query
  -- Show query fields and buttons

  hide card field theSelection
  hide card button cover
  hide card button queryCover
  repeat with n = 1 to 5
    show bg field n
    put n into temp
    add 5 to temp
    show bg field temp
    show cd button n
  end repeat
  show bg button "cancel"
  show bg button "query search"
  show bg button "remove"
  show card button stackCover
  hide cd field theStack
  hide cd field images
  repeat with z = 7 to 13
    hide cd button z
  end repeat
  set hilite of target to false
end mouseUp
```

4/4/89 6:31

Script of bkgnd button id 99 = "Cancel"

```
on mouseUp
  global place
  put "stackBuilder" into place

  -- Place is either "stackBuilder" or "query"
  -- This makes sure that the info.
  -- gets put into the correct containers (hidden fields).

  set cursor to 4
  set hilite of target to true
  repeat with x = 1 to 6 --Hilite the correct button
    if (the short name of bg button x = "cancel") then
      set hilite of bg button x to true
    else
      set hilite of bg button x to false
    end if
  end repeat

  -- Reset card buttons and fields to "Stack Builder" state
  hide card button cover
  hide cd field theSelection
  show cd button queryCover

  repeat with n = 1 to 5
    put n into temp
    add 5 to temp
    hide bg field temp
    hide bg field n
    hide cd button n
    set hilite of cd button n to false
  end repeat
  hide bg button cancel
  hide bg button "query Search"
  hide bg button "remove"
  hide card button "add to stack"
  put empty into card field result
  hide card field result
  hide cd button stackCover
  repeat with x = 1 to 6
    show bg button x
  end repeat
  if cd field theStack is not empty then
    show cd field theStack
    show cd field images
    repeat with z = 7 to 13
      show cd button z
    end repeat
  else
    show cd button 13
  end if
  set hilite of target to false
end mouseUp
```

```
-- getFinalResult was written by Steve Kurtz
-- It determines which of the 5 hidden lists will be used for the query
-- and calls the interList function (in the script of this stack)
```

```
on getFinalResult
  set cursor to 4
  global finalResult, nextList
  put empty into finalResult
  put empty into lists

  --check to see which of the lists contain card ids and frame numbers
  repeat with N = 1 to 5
    if cd field ("list"&N) is not "no entry" then
      put (cd field ("list"&N))&"," after lists
    end if
  end repeat

  put "dummy item" after lists
  put item 1 of lists into finalResult
  delete item 1 of lists

  --if there are more than 2 constraints
  --intersect the lists, 2 at a time until there are no more
  --put the result of the intersection into finalResult
  repeat while number of items of lists > 1
    put item 1 of lists into nextList
    delete item 1 of lists
    put interList(finalResult, nextList) into finalResult
    --interList is a fcn in stack script
  end repeat
end getFinalResult
```

```
on mouseUp
  global finalResult
  --finalResult is the end product of getFinalResult
  --It holds the result of the constrained search of the database
  set hilite of target to true
  hide bg button "remove"
  set cursor to 4
  put empty into temp
  repeat with x = 1 to 5
    set hilite of bg button x to false
    hide bg button x
    if bg field x is not empty then put x after temp
  end repeat
```

```
--Can't conduct a search if there are no constraints!
if temp = empty then
  answer "Make some selections first!" with "OK"
  set hilite of target to false
  repeat with x = 1 to 5
    show bg button x
  end repeat
  show bg button remove
  exit mouseUp
else
  getfinalResult --See above
  put number of lines in finalResult into lineTotal
  --lineTotal is the no. of images that match all constraints

  --Let the user know what the result of the search is...
  if lineTotal = 0 then
    answer "There are no matches for this query." with "OK"
```

```
    if it is "OK" then set hilite of target to false
    repeat with x = 1 to 5
        show bg button x
    end repeat
    show bg button "Remove"
    exit mouseUp
else
    if lineTotal = 1 then
        put "The result is "&lineTotal&" image!" ↵
        into card field result
    else
        put "The result is "&lineTotal&" images!" ↵
        into card field result
    end if
end if

show card field result
show card button "add to stack"
set hilite of target to false
end if
end mouseUp
```



```
on mouseUp
  set hilite of target to true
  -- Check to see if any buttons are hilited (fields selected)...
  if (hilite of card button 1 is false) and ¬
    (hilite of card button 2 is false) and ¬
    (hilite of card button 3 is false) and ¬
    (hilite of card button 4 is false) and ¬
    (hilite of card button 5 is false) then
    --If nothing has been selected...
    answer "Click on a field to select it, then clear." with "OK"
    set hilite of target to false
    exit mouseUp

    --Clear the selected field
    --and put "no entry" into the correct hidden field
  else
    repeat with x = 1 to 5
      if hilite of card button x is true then
        put empty into bg field x
        put x into temp
        add 5 to temp
        put empty into bg field temp
        put "no entry" into cd field ("list"&x)
        put "no entry" into cd field ("fNum"&x)
        set hilite of card button x to false
        set hilite of target to false
        exit repeat
      end if
    end repeat
  end if
end mouseUp
```

4/4/89 6:33

Script of card button id 81 = "Add To Stack"

```
on mouseUp
  set cursor to 4
  global finalResult, Place
  put "stackBuilder" into Place

  --finalResult is the end product of getFinalResult
  --It holds the result of the constrained search of the database.
  --getFinalResult can be found in the script
  --of cd button "query search"

  --Place is either "stackBuilder" or "query"
  --This makes sure that the info.
  --gets put into the correct containers (hidden and visible fields).

  set hilite of target to true
  ask "Name the result."
  if it is empty then
    resetCard
  else
    put it into queryName --if it is not empty
    put queryName & return after card field theStack
    put number of lines in finalResult & return after card field images
    if last character of finalResult = return then
      delete last character of finalResult
    end if
    put finalResult & "," & return after cd field "itemHolder"
    --"itemHolder" is the hidden field that holds the card ids
    --and frame numbers of the items in the Stack.
    resetCard
  end if
end mouseUp

on resetCard
  repeat with a = 1 to 5
    put a into temp
    add 5 to temp
    hide bg field temp
    hide bg field a
    hide cd button a
    set hilite of cd button a to false
  end repeat

  repeat with z = 1 to 6
    show bg button z
  end repeat
  show cd button queryCover

  hide bg button "Query Search"
  hide bg button "remove"
  hide card button "add to stack"
  hide card field result
  hide bg button Cancel

  hide cd button stackCover
  if cd field theStack is not empty then
    show cd field theStack
    show cd field images
    repeat with b = 7 to 13
      show card button b
    end repeat
  end if
  set hilite of target to false
end resetCard
```

on mouseUp

global cardName, place

--Lists of card ids and frame numbers are stored on cards.

--Each list is sorted a different way.

--DesignerList is sorted by Date, Taxonomy, Medium, Location, Designer

--CardName is the name of the card and list from which the item came.

--ItemHolder is a hidden field that contains the card ids

--and frame numbers of everything in the Stack.

--The first word of each line is a card id.

--The second word of each line is its corresponding frame number.

--Place is either "stackBuilder" or "query"

--This makes sure that the info.

--gets put into the correct containers (hidden fields).

put clickLine () into lineNum

select line lineNum of card field theSelection

put (item lineNum of bg field idList of cd cardName) into tempitem

--tempitem is a temporary container.

if first line of tempitem is empty then delete first line of tempitem

put number of lines in tempitem into imageHolder

--imageHolder is a temporary container.

if place = "stackBuilder" then

put tempitem & "," & return after card field "itemHolder"

put (line lineNum of card field theSelection)→

& return after card field theStack

put imageHolder & return after card field images

repeat with x = 7 to 12

show card button x

end repeat

show cd field theStack

show cd field images

hide card field "theSelection"

hide card button cover

set hilite of bg button cardName to false

-- the lines of the stack will correspond

-- to the correct items in card field "itemHolder"

-- this allows access to these items separately

-- even after they have been put into the Stack

else

if place = "query" then

hide card field "theSelection"

hide card button cover

set hilite of bg button cardName to false

put (line lineNum of card field theSelection)→

into bg field cardName

if cardName is "designer" then

put tempitem into cd field "list1"

put imageHolder into bg field 6

else if cardName is "medium" then

put tempitem into cd field "list3"

put imageHolder into bg field 8

else if cardName is "location" then

put tempitem into cd field "list2"

put imageHolder into bg field 7

4/4/89 6:34

Script of card field id 75 = "theSelection"

Page 2

```
    end if  
  end if  
end if  
end mouseUp
```

```

on mouseUp
  global indexHolder, counter, theNum, source
  put empty into source
  put empty into indexHolder
  put empty into counter
  put clickLine () into theNum
  if visible of cd button cover is true then
    repeat with x = 1 to 5
      set hilite of bg button x to false
    end repeat
    hide cd field theSelection
    hide cd button cover
  end if

  -- IndexHolder keeps track of the number of the video image
  -- that is on the screen when the user has stopped scanning.
  -- It allows the user to scan images, stop on a particular one
  -- and go to the correct data card for that image.
  -- It is important that the IndexHolder is reset
  -- every time a new line is selected
  -- so that the new item can be viewed in the same way.

  -- Counter keeps track of the line number
  -- of the item that is being scanned. The forward and backward
  -- scan buttons add and subtract from counter.

  -- theNum represents both the line in field "theStack"
  -- that has been selected and the number of the corresponding item
  -- in hidden card field "itemHolder."

  set cursor to 4

  put theTicks into originalTicks
  -- this script differentiates the single click operations
  -- from the double click operations. The user can double click
  -- anywhere in card field "theStack" to select the whole thing.
  wait 15 ticks
  if the mouseClick is true then --if double clicked...
    put 0 into temp
    repeat with x = 1 to number of lines in cd field theStack
      add 1 to temp
      put x into buttonNum
      add 16 to buttonNum
      set hilite of cd button buttonNum to true
      -- Hilite all the lines that are not empty
      -- visual feedback for "select all."
    end repeat
    if temp = 1 then
      put 1 into theNum
    else
      put "selectAll" into theNum
    end if
  else
    if line theNum of cd field theStack is empty then
      --if the user clicks once on an empty line turn off all buttons.
      put empty into theNum
      repeat with x = 1 to number of lines in cd field theStack
        put x into buttonNum
        add 16 to buttonNum
        set hilite of cd button buttonNum to false
      end repeat
    exit mouseUp
  end if

```

```
else
  put (theNum + 16) into oneButton
  repeat with x = 1 to number of lines in cd field theStack
    -- If the user clicks once on a full line
    -- hilite the correct button and turn the others off.
    put x into buttonNum
    add 16 to buttonNum
    set hilite of cd button buttonNum to false
  end repeat
  set hilite of cd button oneButton to true
end if
end mouseUp
```

```

--Scan uses the SearchVideo xcmd installed in this stack
--Hidden card field "itemHolder" contains a list of card ids
--and frame numbers. The second word of each line in the list
--is a frame number.

on mouseUp
  global theNum, counter, scanHolder, indexHolder, source

  -- theNum represents both the line in field "theStack"
  -- that has been selected and the number of the corresponding item
  -- in hidden card field "itemHolder."

  -- Counter keeps track of the line number
  -- of the item that is being scanned. The forward and backward
  -- scan buttons add and subtract from counter.

  -- ScanHolder contains the items that are being scanned.
  -- The first word of each line is a card id number.
  -- The second word of each line is a frame number.

  -- IndexHolder keeps track of the number of the video image
  -- that is on the screen when the user has stopped scanning.
  -- It allows the user to scan images, stop on a particular one
  -- and go to the correct data card for that image.

  --Source is either "scan" or "reference".
  --The script of the stack "cards" checks to see what source is
  --to determine whether or not to initialize the scroll box
  --and where to place it.
  --If source is scan then the scroll bar is initialized
  --with a new list and indexHolder determines
  --the placement of the scroll box.

  if visible of cd button cover is true then
    repeat with x = 1 to 5
      set hilite of bg button x to false
    end repeat
    hide cd field theSelection
    hide cd button cover
  end if

  put "scan" into source
  put empty into counter
  set hilite of target to true
  set cursor to 4

  if theNum is empty then --if nothing has been selected. .
    answer "Click or double click, select items to view" with "OK"
    set hilite of target to false
    --these buttons hilite each selected lines of the Stack
    repeat with x = 17 to 28
      set hilite of cd button x to false
    end repeat
    exit mouseUp
  end if

  if theNum = "selectAll" then
    put cd field itemHolder into scanHolder
  else
    if theNum is not "selectAll" then
      put item theNum of cd field itemHolder into scanHolder
    end if
  end if
end if

```

```
if line 1 of scanHolder = empty then delete line 1 of scanHolder
if number of lines in scanHolder = 1 then
  searchVideo (second word of line 1 of scanHolder)
  --the second word of each line is a frame number
  set hilite of card button scan to false
  put 1 into indexHolder
  beep
  exit mouseUp
else
  repeat with x = 1 to number of lines in scanHolder
    if the mouseClick is true then
      put x into counter
      set hilite of card button scan to false
      exit repeat
    else
      searchVideo (second word of line x of scanHolder)
      --the second word of each line is a frame number
      put x into counter
      wait 130 ticks
    end if
  end repeat
  if (counter > 1) and¬
    (counter is not number of lines in scanHolder) then
    subtract 1 from counter
  end if
  put counter into indexHolder
  set hilite of card button scan to false
end if
beep
end mouseUp
```



```

--See script of card button "scan" for comments
on mouseUp
  global Counter, theNum, scanHolder, indexHolder, source
  put "scan" into source
  set cursor to 4

  if visible of cd button cover is true then
    repeat with x = 1 to 5
      set hilite of bg button x to false
    end repeat
    hide cd field theSelection
    hide cd button cover
  end if

  if theNum is empty then
    answer "Click or double click, select items to view" with "OK"
    set hilite of target to false
    exit mouseUp
  end if

  if counter is empty then
    if theNum = "selectAll" then
      put cd field itemHolder into scanHolder
    else
      if theNum is not "selectAll" then
        put item theNum of cd field itemHolder into scanHolder
      end if
    end if

    if line 1 of scanHolder = empty then delete line 1 of scanHolder
    put number of lines in scanHolder into counter
    searchVideo (second word of line counter of scanHolder)
    put counter into indexHolder

  else
    if Counter = 1 then
      put number of lines in scanHolder into counter
      searchVideo (second word of line counter of scanHolder)
      put counter into indexHolder
      set hilite of target to false
      exit mouseUp
    else
      if counter is not 1 then subtract 1 from Counter
      searchVideo (second word of line counter of scanHolder)
      put counter into indexHolder
    end if
  end if
end mouseUp

```

--See script of card button "scan" for comments.

on mouseUp

```
global Counter, theNum, scanHolder, indexHolder, source
put "scan" into source
set cursor to 4
```

```
if visible of cd button cover is true then
  repeat with x = 1 to 5
    set hilite of bg button x to false
  end repeat
  hide cd field theSelection
  hide cd button cover
end if
```

```
if theNum is empty then
  answer "Click or double click, select items to view" with "OK"
  set hilite of target to false
  exit mouseUp
end if
```

```
if counter is empty then
  if theNum = "selectAll" then
    put cd field itemHolder into scanHolder
  else
    put item theNum of cd field itemHolder into scanHolder
  end if
```

```
  if line 1 of scanHolder = empty then delete line 1 of scanHolder
  put 1 into counter
  searchVideo (second word of line counter of scanHolder)
  put counter into indexHolder
  set hilite of target to false
else
  if Counter = number of lines in scanHolder then
    put 1 into counter
    searchVideo (second word of line counter of scanHolder)
    put counter into indexHolder
    set hilite of target to false
    exit mouseUp
  else
    add 1 to Counter
    searchVideo (second word of line counter of scanHolder)
    put counter into indexHolder
  end if
end if
end mouseUp
```

```
--There are 2 ways to save a file: as a unified list or an itemized list
--A unified list has no duplicates
--and is represented with a name given to the union by the user.
--When reading a unified file back into the stack it appears as 1 item.
--An itemized file will read into the stack
--exactly as it was originally - an itemized list.
```

```
on mouseUp
```

```
global theNum, stackEntry, imageTotal, savedItems
global unionResult, fileName
```

```
if visible of cd button cover is true then
  repeat with x = 1 to 5
    set hilite of bg button x to false
  end repeat
  hide cd field theSelection
  hide cd button cover
end if
```

```
set hilite of target to true
```

```
-- theNum represents both the line in field "theStack"
-- that has been selected and the number of the corresponding item
-- in hidden card field "itemHolder."
```

```
--The PathName xfcn written by Andrew Gilmartin at Brown University.
--PathName returns a full pathname for the file name
--StackEntry is the short version of the file name
```

```
--imageTotal is the number of images being saved.
--In a unified file it is 1 number.
--Otherwise it is an itemized list.
```

```
--savedItems is a list of the card ids and frame numbers being saved.
```

```
--UnionResult is returned
--by the function PlusList found in the script of this stack
```

```
if theNum is empty then
  answer "Click on an item in the Stack to save" with "OK"
  set hilite of target to false
  exit mouseUp
end if
```

```
--The user can save 1 item
--or double click to select all items and save the whole Stack.
```

```
if theNum is "selectAll" then
  answer "Save Stack as unified list (no dups), or as is..." with -
  "Union" or "As is" or "cancel"
  if it is "cancel" then
    choose browse tool
    put empty into theNum
    repeat with x = 17 to 28
      set hilite of cd button x to false
    end repeat
    set hilite of target to false
    exit mouseUp
```

```
else
```

```
  if it is "union" then
    set cursor to 4
```

```
  getUnion
```

```
if line 1 of unionResult is empty then
  delete line 1 of unionResult
end if
put number of lines in unionResult into imageTotal
put unionResult into savedItems

get newPathName("Save as:",empty)
if it is empty then
  choose browse tool
  put empty into theNum
  set hilite of target to false
  exit mouseUp
else
  put it into fileName
end if
put fileName into stackEntry
repeat with x = 1 to number of characters in stackEntry
  --to get the short name of the file
  if ":" is in stackEntry then
    delete first character of stackEntry
  else
    exit repeat
  end if
end repeat

else
  if it is "as is" then
    get newPathName("Save as:",empty)
    if it is empty then
      choose browse tool
      put empty into theNum
      set hilite of target to false
      exit mouseUp
    else
      put it into fileName
    end if
    put cd field "itemHolder" into savedItems
    put cd field "images" into imageTotal
    put cd field theStack into stackEntry
  end if
end if
end if
saveFile

else
  if theNum is not "selectAll" then
    put item theNum of cd field "itemHolder" into savedItems
    if line 1 of savedItems is empty then
      delete line 1 of savedItems
    end if
    put the number of lines in savedItems into imageTotal
    put line theNum of cd field theStack into stackEntry
    get newPathName("Save as:",stackEntry)
    if it is empty then
      choose browse tool
      put empty into theNum
      repeat with x = 17 to 28
        set hilite of cd button x to false
      end repeat
      set hilite of target to false
      exit mouseUp
    else
      put it into fileName
```

```
    end if
    saveFile
  end if
end mouseUp
```

```
on saveFile
```

```
--This script uses the same globals as above
--and actually writes the file to a disk.
```

```
global stackEntry, imageTotal, savedItems, fileName
```

```
if last character of savedItems = return then
  delete last character of savedItems
end if
```

```
if last character of imageTotal = return then
  delete last character of imageTotal
end if
```

```
if last character of stackEntry = return then
  delete last character of stackEntry
end if
```

```
open file fileName
write stackEntry to file fileName
write "#" to file fileName
write imageTotal to file fileName
write "#" to file fileName
write savedItems to file fileName
close file fileName
```

```
--Reset the card buttons to false
choose browse tool
put empty into theNum
set hilite of target to false
repeat with x = 17 to 28
  set hilite of cd button x to false
end repeat
end saveFile
```

```
on getUnion
```

```
--uses the function plusList found in the script of this stack.
```

```
global unionResult, nextList
put empty into unionResult
put cd field "itemHolder" into lists
put "dummy item" after lists
put item 1 of lists into unionResult
delete item 1 of lists
if line 1 of lists is empty then delete line 1 of lists
repeat while number of items of lists > 1
  put item 1 of lists into nextList
  delete item 1 of lists
  put plusList (unionResult, nextList) into unionResult
end repeat
end getUnion
```

```
on mouseUp
  global theNum

  --The user must select a line with 1 click
  --or double click in field theStack to select all
  --before it can be removed.

  --ItemHolder is a hidden field with all of the card ids
  --and frame numbers in items - separated by commas.

  --theNum represents both the line in field "theStack"
  --that has been selected and the number of the corresponding item
  --in hidden card field "itemHolder."

  set hilite of target to true
  set cursor to 4

  if theNum is empty then
    answer "Click or double click, select items to remove" with "OK"
    set hilite of target to false
    exit mouseUp
  else
    if theNum is "selectAll" then
      put empty into cd field theStack
      put empty into cd field images
      put empty into cd field "itemHolder"
    else
      set lockScreen to true
      delete line theNum of card field theStack
      delete line theNum of card field images
      delete item theNum of card field "itemHolder"
      repeat while (line 1 of cd field itemHolder is empty) and ¬
        (cd field "itemHolder" is not empty)
        delete line 1 of cd field "itemHolder"
      end repeat
    end if
  end if

  repeat with x = 17 to 28
    set hilite of cd button x to false
  end repeat
  repeat with x = 1 to 6
    set hilite of bg button x to false
  end repeat

  if card field theStack is empty then
    repeat with x = 7 to 12
      hide card button x
    end repeat
    hide cd field theStack
    hide cd field images
    put empty into card field "itemHolder"
  end if
  set hilite of target to false
  put empty into theNum
end mouseUp
```

4/4/89 6:38

Script of card button id 238 = "Cards"

```
on mouseUp
  global theNum, theList

  -- theNum represents both the line in field "theStack"
  -- that has been selected and the number of the corresponding item
  -- in hidden card field "itemHolder."

  --theList is card ids of the selected item or items from theStack
  --theList is used to initialize and place the scrollBox (on data card)
  --and is accessed on the goToCard script in the data card stack.

  set hilite of target to true
  set cursor to 4

  if theNum is empty then
    answer "Click or double click to select items first" with "OK"
    set hilite of target to false
    exit mouseUp
  else

    if theNum = "selectAll" then
      --put the whole field into theList
      put cd field "itemHolder" into theList
    else
      --put just the selected item into theList
      put item theNum of cd field "itemHolder" into theList
    end if
  end if
  if line 1 of theList = empty then delete line 1 of theList

  visual effect iris open
  set hilite of target to false
  --The first word of each line of theList is a card id number.
  go card id (first word of line 1 of theList) of stack cards
  put empty into theNum
end mouseUp
```

4/4/89 6:39

Script of card button id 239 = "Ref."

```
on mouseUp
  global theNum
  if theNum is not empty then send mouseUp to cd stackBuilder

  --theNum represents both a line in field "theStack"
  --that has been selected and the number of the corresponding item
  --in hidden card field "itemHolder."

  set hilite of target to true
  push card --to remember where we came from
  visual effect iris open
  go card 1 of stack "reference" --go to intro. card of reference
  set hilite of target to false
end mouseUp
```



```
on openCard
  --reset the card to its most neutral state
  hide card field theSelection
  hide card button cover
  repeat with x = 17 to 28
    set hilite of cd button x to false
  end repeat
end openCard

on mouseUp
  global theNum, counter
  -- theNum represents both a line in field "theStack"
  -- that has been selected and the number of the corresponding item
  -- in hidden card field "itemHolder."

  -- Counter keeps track of the line number
  -- of the item that is being scanned. The forward and backward
  -- scan buttons add and subtract from counter.

  -- This script resets these variables and any hilited buttons
  -- when a user clicks anywhere outside of card field "theStack"

  repeat with x = 17 to 28
    if (x is 17) or (x is 18) or (x is 19) or
    or (x is 20) or (x is 21) then
      put x into temp
      subtract 16 from temp
      set hilite of cd button temp to false
      set hilite of cd button x to false
    else
      set hilite of cd button x to false
    end if
  end repeat
  put empty into theNum
  put empty into counter
end mouseUp
```

on openCard

```
--Hide the scan buttons if there is nothing to scan.
if bg field scanList contains "no matches" then
  hide bg button scan
  hide bg button back
  hide bg button forward
else
  show bg button scan
  show bg button back
  show bg button forward
end if
end openCard
```

on idle

```
--Makes the Marker flash when the computer is not doing anything else.
wait 10 ticks
set hilite of card button "Marker" to
not (hilite of card button "Marker")
end idle
```

on mouseUp

```
--The Classification Tree was originally designed for Hypercard
--and script written by Luc Perron in the spring 1988.
--It has been rewritten and redesigned to solve some problems,
--but the basic concepts remain the same.
```

```
--Since some of the lower branches of the tree have the same name,
--it was necessary to find a way to name these cards uniquely.
--Each card name now contains the first character of the first word
--of the branch directly above it and a number 1 or 2.
--1 signifies the "Design Work" branch.
--2 signifies the "Designers Archive" branch.
```

```
--This script controls the navigation through the Classification Tree.
--Background Fields must have a name and be locked for this to work.
```

set cursor to 4

```
if the Target contains "Bkgnd Field" then --is it a background field?
  put the short name of the target into theField --get its name
  put ClickLine() into lineNumber --xfcn in script of this stack
  put line lineNumber of field theField into theName --get the line
```

```
--These first three cards have nothing after their name...
```

```
if (theName is "Classification Tree") or
(theName is "design work")
or (theName is "designers Archive") then
  go cd theName
  exit mouseUp
end if
```

```
if theField is "to" then
```

```
  put " " &
  (first character of first word of bg field "currentLocation") &
  (first character of bg field refNum) after theName
```

```
else
```

```
  if (theField is "from") then
    put " " &
    (first character of first word of line (lineNumber - 1)
of field "from") &
    (first character of bg field refNum) after theName
```

```
  end if
end if
```

4/4/89 6:40

Script of background id 3711 = "TaxonomyMap"

Page 2

```
    if theName is not empty then go to card theName
end if
end mouseUp
```

```
--Scan uses the SearchVideo xcmd installed in this stack
--Each branch in the tree is represented by a card.
--Hidden background field "scanList" (on every card)
--contains a list of frame numbers for all the images in the database
--that are in the branch represented by that card.
```

```
on mouseUp
```

```
  global imageCounter
  put empty into imageCounter
```

```
  --ImageCounter keeps track of the line number
  --of the item that is being scanned. The forward and backward
  --scan buttons add 1 and subtract 1 from imageCounter.
  --ImageCounter needs to be reset (with empty) with each new scan.
```

```
  set hilite of target to true
  set cursor to 4
```

```
  --if there are no images in that branch...
```

```
  if bg field scanList contains "no matches" then
    put bg field currentLocation into holder
    answer "There are no " & holder & " images" with "OK"
    set hilite of target to false
    exit mouseUp
```

```
  else
```

```
    --Keep scanning until the user clicks the mouse...
```

```
    repeat with imageCounter = 1 to number of lines in bg field scanList
```

```
      if the mouseClicked is true then
        set hilite of target to false
        beep
        subtract 1 from imageCounter
        exit mouseUp
```

```
      else
```

```
        searchVideo (second word of line imageCounter of bg field scanList)
        wait 130 ticks
```

```
      end if
```

```
    end repeat
```

```
    set hilite of target to false
```

```
    beep
```

```
  end if
```

```
end mouseUp
```

5/2/89 11:03 PM Script of bkgnd button id 50 = "back"

```
on mouseUp
  global imageCounter

  --ImageCounter keeps track of the line number
  --of the item that is being scanned. The forward and backward
  --scan buttons add 1 and subtract 1 from imageCounter.

  set cursor to 4
  if (imageCounter = 1) or (imageCounter = empty) then
    put (number of lines in bg field scanList) into imageCounter
    searchVideo (second word of line imageCounter of bg field scanList)
  else
    subtract 1 from imageCounter
    searchVideo (second word of line imageCounter of bg field scanList)
  end if
end mouseUp
```

5/2/89 11:04 PM Script of bkgnd button id 51 = "forward"

```
on mouseUp
  global imageCounter

  --ImageCounter keeps track of the line number
  --of the item that is being scanned. The forward and backward
  --scan buttons add 1 and subtract 1 from imageCounter.

  set cursor to 4
  set hilite of target to true

  if (imageCounter = number of lines in bg field scanList) or
  (imageCounter = empty) then
    put 1 into imageCounter
    searchVideo (second word of line imageCounter of bg field scanList)
  else
    add 1 to imageCounter
    searchVideo (second word of line imageCounter of bg field scanList)
  end if
  set hilite of target to false
end mouseUp
```

```
--This script makes a list of card ids and frame numbers
--of all the images that match a particular branch of the tree.
--Then it prompts the user for directions on what to do with the list.
```

```
on mouseUp
```

```
  global treePlace, theItemHolder, ImageHolder
```

```
  --Treeplace is the name of the branch of the tree being returned from.
  --TheItemHolder - list of card ids and frame numbers for that branch.
  --ImageHolder is the total number of images from that branch.
```

```
  set hilite of target to true
```

```
  set lockScreen to true
```

```
  set cursor to 4
```

```
  put bg field currentLocation into treePlace
```

```
  if bg field scanList contains "no matches" then
```

```
    answer "No matches were found, return anyway?" with "Yes" or "No"
```

```
    if it is "yes" then
```

```
      set lockScreen to false
```

```
      visual effect iris close
```

```
      go card stackBuilder
```

```
      set hilite of target to false
```

```
      exit mouseUp
```

```
    else
```

```
      set hilite of target to false
```

```
      exit mouseUp
```

```
  end if
```

```
  else
```

```
    put bg field scanList into theItemHolder
```

```
    put number of lines in theItemHolder into ImageHolder
```

```
  end if
```

```
  putInfoInPlace
```

```
end mouseUp
```

```
on putInfoInPlace
```

```
  global theItemHolder, imageHolder, TreePlace, Place
```

```
  -- Place is either "stackBuilder" or "query"
```

```
  -- This makes sure that the info.
```

```
  -- gets put into the correct containers (hidden & visible fields)
```

```
  -- when returning to the StackBuilder card.
```

```
  if place is "query" then
```

```
    answer "Use " & imageHolder & " images, for a query constraint?"
```

```
    with "Yes" or "No" or "Cancel"
```

```
    if it is "cancel" then --cancel return, stay on the present card
```

```
      set hilite of target to false
```

```
      exit putInfoInPlace
```

```
    end if
```

```
    if it is "yes" then --return with info for query constraint
```

```
      set lockScreen to false
```

```
      visual effect iris close
```

```
      go card stackBuilder
```

```
      set hilite of target to false
```

```
      --put the info in the right place and reset the buttons
```

```
      put treePlace into bg field tree
```

```
      put imageHolder into bg field 10
```

```
      put theItemHolder into cd field list5
```

```
      set hilite of bg button tree to false
```

```
  else
```

```
    if it is "no" then --return, but don't use the info for a query...
```

```
      set lockScreen to false
```

```
      visual effect iris close
```

```
      go card stackBuilder
```

```
    set hilite of target to false
    set hilite of bg button tree to false
  end if
end if
end if

if place is "stackBuilder" then
  answer "Add " & imageHolder & " images, to stack?" with-
  "Yes" or "No" or "Cancel"
  if it is "cancel" then --cancel return, stay on present card...
    set hilite of target to false
    exit putInfoInPlace
  end if
  if it is "yes" then --add the list to the Stack...
    set lockScreen to false
    visual effect iris close
    go card stackBuilder
    set hilite of target to false
    --make sure there are no empty lines in the list
    if first line of theItemHolder is empty then
      delete first line of theItemHolder
    end if
    put theItemHolder & "," & return after cd field "itemHolder"
    --Keep the list as an item (between commas)
    put treePlace & return after card field theStack
    put imageHolder & return after card field images
    set hilite of bg button tree to false
    --reset the buttons on the stackBuilder card...
    if cd field theStack is not empty then
      show cd field theStack
      show cd field images
      repeat with n = 7 to 12
        show cd button n
      end repeat
    end if
  else
    --if it is "No" then return, but don't add the list to the stack.
    set lockScreen to false
    visual effect iris close
    go card stackBuilder
    set hilite of target to false
    set hilite of bg button tree to false
  end if
end if
end putInfoInPlace
```



```
--Graphic Design Archive Stackware™
--Copyright 1989 Rochester Institute of Technology
--Portions copyright 1988/1987 Apple Computer, Inc.
--Unless otherwise noted all scripts written by Cathleen Britt at RIT.
```

```
on openStack
```

```
  global initialized, index, indexHolder, source
```

```
  --Initialized is a boolean expression (true or false)
  --It insures that the user always opens the GDA Archive
  --through the "GDA Start" stack
  --and that the necessary globals (xcmds) are declared.
  --If it is true that means that the globals in stack "GDA Start"
  --have been declared and that the video drivers are working.
  --If it is not true then go to stack "GDA Start" and openStack there.
```

```
  --Index is the number of the card that is open
  --(number of the line in theList)
  --The index number is shown on the card in the scroll box
```

```
  --IndexHolder keeps track of the number of the video image
  --that is on the screen when the user has stopped scanning.
  --It allows the user to scan images, stop on a particular one
  --and go to the correct data card for that image.
```

```
  --Source is either "scan" or "reference".
  --This script checks to see what source is
  --to determine whether or not to initialize the scroll box
  --and where to place it.
```

```
  if initialized is not true then --to declare globals...
```

```
    go to stack "GDA Start"
```

```
  else
```

```
    --if returning to card stack from reference stack...
```

```
    if source is "reference" then --return to the correct card...
```

```
      put indexHolder into index --show the correct index number...
```

```
      put empty into source
```

```
    else
```

```
      --If source is scan then the scroll bar is initialized
```

```
      --with a new list and indexHolder determines
```

```
      --the placement of the scroll box.
```

```
      if source is "scan" then
```

```
        initializeScrollBar
```

```
        put indexHolder into index --get correct index number
```

```
        goToCard --go to correct card
```

```
        put empty into source
```

```
      else
```

```
        --go to first card id on theList
```

```
        hide menuBar
```

```
        initializeScrollBar
```

```
        goToCard
```

```
      end if
```

```
    end if
```

```
  end if
```

```
end openStack
```

```
on openCard
```

```
  global playerType
```

```
  --PlayerType -the type of videoDisc player that is being used.
```

```
PlaceScrollBar
```

```
--If no videoDisc player is being used...
```

```
if playerType = "none" then --disables searchVideo command
```

```
  exit openCard
```

```

else
  --each card has a hidden field that holds a frame number...
  put first word of field "frame Number" into FNumber
  if field "Frame Number" is not empty then
    searchVideo FNumber --xcmd to find that frame on disc
  end if
end if
end openCard

on closeCard
  put empty into field "scrollBox"
end closeCard

on goToCard
  global Index, theList
  if theList is EMPTY then
    go to card Index
  else
    go to card ID (first word of line index of theList)
  end if
end goToCard

on mouseStillDown
  --This is used only by the scroll bar to allow for continuous scroll
  send mouseDown to the Target
end mouseStillDown

on InitializeScrollBox
  --Written by Luc Perron, 1988, RIT
  --This part of the code is to define the global variables that
  --are going to be used for the scroll bar. It should be used every
  --time there is a change in the total number of items in theList
  --The scroll box contains 3 buttons and one field.
  global maxLeft, maxRight, deltaX, V
  global theList, index, indexMax, theStep
  put 1 into index
  if theStep is empty then put 10 into theStep
  if theList is empty then
    put the number of cards into indexMax
  else
    put the number of lines in theList into indexMax
  end if
  get rect of field "scrollBox"
  put (third item of it) - (first item of it) into width
  --save the y coordinate (V) so that the scrollbar
  --always move at the same height across the screen...
  put second item of the location of field "scrollBox" into V
  put first item of rect of bkgnd button "pageScroll" + (width div 2) -
  into maxLeft
  get rect of bkgnd button "Next"
  put third item of rect of bkgnd button "pageScroll" - (width div 2) -
  into maxRight
  put (maxRight - maxLeft) / indexMax into DeltaX
end InitializeScrollBox

on PlaceScrollBox
  --Written by Luc Perron, 1988, RIT
  --This part will place the scroll box at its proper place each time
  --you open a new card. (More calculations...)
  global MaxLeft, MaxRight, DeltaX, V, Index, indexMax, theList
  put Index into field "scrollBox"
  if Index = 1 then
    put MaxLeft into H    -- This is the place for the first card
  else if Index = indexMax then

```

```

    put MaxRight into H    -- This is the place for the last card
else
    put trunc (MaxLeft + (Index - 1) * DeltaX + DeltaX / 2) into H
end if
show field "scrollBox" at H, V  -- Show the box at its place
end PlaceScrollBox

```

on DragBox

```

--Originally written by Luc Perron
--Modified to work with large stacks 6/10/88 MAC/SHK
--This code is to allow the user to drag the scroll box around while
--displaying the proper card number, and then go to the selected
--card as soon as the user releases the mouse button.

```

```

global MaxLeft, MaxRight, DeltaX, V, Index, theList, IndexMax

```

```

repeat until the mouse is up

```

```

    put the mouseH into H

```

```

    if H < MaxLeft then put MaxLeft into H

```

```

    else if H > MaxRight then put MaxRight into H

```

```

    show field "scrollBox" at H, V

```

```

    add 0.49 to H

```

```

    put trunc ((H - MaxLeft - 0.5) / DeltaX) + 1 into temp

```

```

    if temp < 1 then

```

```

        put 1 into Index

```

```

    else

```

```

        put temp into index

```

```

    end if

```

```

    put index into field "scrollBox"

```

```

end repeat

```

```

goToCard

```

```

end DragBox

```

on ScrollPage

```

--Written by Luc Perron

```

```

--This code controls the action when the user clicks in the black
--area of the scroll bar.  If user clicks on the right of the scroll
--box, go forward, if user clicks on the left, go backward.

```

```

--At the moment, it does the same as the arrow buttons, but the code
--could easily be modified to move by more than one step.

```

```

--Put the increment desired into "theStep" in initializeScrollBox.

```

```

global Index, theStep, indexMax

```

```

if the mouseH > first item of location of field "scrollBox" then

```

```

    add theStep to Index

```

```

    visual effect scroll left --slow

```

```

else

```

```

    visual effect scroll right --slow

```

```

    subtract theStep from Index

```

```

end if

```

```

if Index > indexMax then put indexMax into Index

```

```

else if Index < 1 then put 1 into Index

```

```

goToCard

```

```

end ScrollPage

```

on ScrollLeft

```

--Written by Luc Perron

```

```

-- Go to the previous card in the list.

```

```

global Index, indexMax

```

```

visual effect scroll right

```

```

if Index > 1 then

```

```

    subtract 1 from Index

```

```

    goToCard

```

```

end if

```

```

wait 90 ticks

```

```

end ScrollLeft

```

```
on ScrollRight
--Written by Luc Perron
--Go to the Next card in the list.
global Index, indexMax
visual effect scroll left
if Index < indexMax then
    add 1 to Index
    goToCard
end if
wait 90 ticks
end ScrollRight

--on sort
--The cards have been sorted to put them in specific order...
--answer "It would not be wise to sort this stack" with "OK"
--end sort

on doMenu command
    if command = "Delete Card" then
        beep
        --answer "Please don't delete any cards!" with "OK"
    else
        if command = "Quit HyperCard" then
            --To insure a fresh start...
            set lockscreen to true
            go cd stackBuilder of stack "GDA Start"
            put empty into card field theStack
            put empty into card field images
            put empty into card field "itemHolder"

            repeat with z = 1 to 10
                put empty into bg field z
                hide bg field z
            end repeat

            repeat with a = 1 to 5
                put "no entry" into cd field ("list" & a)
            end repeat
            hide bg button "cancel"
            hide bg button "Query Search"
            hide bg button "Remove"

            repeat with b = 7 to 12
                hide card button b
            end repeat
            hide cd field theStack
            hide cd field images
            repeat with x = 17 to 28
                set hilite of cd button x to false
            end repeat
            pass doMenu
        end if
    end if
    pass doMenu
end doMenu
```

4/4/89 6:44

Script of bkgnd button id 148 = "Ref."

```
on mouseUp
  --This script takes the user to the corresponding card
  --in the reference stack (according to designer name)
  --Refer to openStack in the script of this stack
  --for definition of global variables.
  global index, indexHolder, source
  set hilite of target to true
  set cursor to 4
  put index into indexHolder
  put "reference" into source
  push card
  put field "designer" into referenceName
  set hilite of target to false
  visual effect iris open
  go cd referenceName of stack "Reference"
end mouseUp
```

4/4/89 6:45

Script of bkgnd button id 141 = "Return"

```
on mouseUp
  set hilite of target to true
  visual effect iris close
  set hilite of target to false
  go cd stackBuilder of stack "GDA Start"
end mouseUp
```

```
on openStack
  global initialized

  --Initialized is a boolean expression (true or false)
  --It insures that the user always opens the GDA Archive
  --through the "GDA Start" stack
  --and that the necessary globals (xcmds) are declared.
  --If it is true that means that the globals in stack "GDA Start"
  --have been declared and that the video drivers are working.
  --If it is not true then go to stack "GDA Start" and openStack there.

  if initialized is not true then
    go to stack "GDA Start"
  else
    hide menuBar
  end if
end openStack

function ClickLine
  -- Script written by Luc Perron in the spring of 1988 at RIT
  -- This function returns the line number of where the mouse has been
  -- clicked in a field (Scrolling or not).
  -- Ex: put ClickLine() into LineNumber
  put item 2 of the rect of the Target into Top
  put item 2 of the clickLoc into Y
  if the style of the Target is "scrolling" then
    return (Y - Top + scroll of the Target) div-
      (textHeight of the Target) + 1
  else
    return (Y - Top) div (textHeight of the Target) + 1
  end if
end ClickLine

on doMenu command
  if command = "quit hyperCard" then
    set lockscreen to true
    set cursor to 4
    --To insure a fresh start, clean up the "stackBuilder" card
    --Reset buttons and clean out fields....
    go cd stackBuilder of stack "GDA Start"
    put empty into card field theStack
    put empty into card field images
    put empty into card field "itemHolder"

    repeat with z = 1 to 10
      put empty into bg field z
      hide bg field z
    end repeat

    repeat with a = 1 to 5
      put "no entry" into cd field ("list" & a)
    end repeat
    hide bg button "cancel"
    hide bg button "Query Search"
    hide bg button "Remove"

    repeat with b = 7 to 12
      hide card button b
    end repeat
    hide cd field theStack
    hide cd field images

    repeat with x = 17 to 28
      set hilite of cd button x to false
```

4/4/89 6:46

Script of stack Reference:Reference

Page 2

```
    end repeat  
  end if  
  pass doMenu  
end doMenu
```



4/4/89 6:46

Script of bkgnd button id 18 = "Index"

on mouseUp

```
--This script hides and empties field "indexHolder" if it is visible
--If it is hidden, it gets the contents
--of field "indexHolder" on the introduction card
--and puts it into the field "indexHolder" on this card and shows it.
--The contents of field "indexHolder" is stored one time only.
```

```
if visible of bg field indexHolder is true then
  set visible of bg field indexHolder to false
  put empty into bg field "indexHolder"
  hide bg button cover
  set hilite of target to false
else
  set hilite of target to true
  hide bg field notes
  set hilite of bg button notes to false
  get bg field "indexHolder" of cd "introduction"
  put it into bg field "indexholder" of this cd
  set scroll of bg field "indexHolder" to 0
  show bg button cover
  show bg field "indexHolder"
end if
end mouseUp
```

4/4/89 6:47

Script of bkgnd field id 27 = "indexHolder"

on mouseUp

--Refer to script of this stack for more info on "clickLine ()"

put ClickLine() into lineNum

put line lineNum of bg field indexHolder into cardName

select line lineNum of bg field indexHolder

--show that it has been selected by hiliting it.

set lockscreen to true

hide bg field "indexHolder"

hide bg button cover

set hilite of bg button "index" to false

set lockscreen to false

visual effect iris open

go cd cardName

end mouseUp

4/4/89 6:47

Script of bkgnd button id 45 = "Notes"

```
on mouseUp
  if visible of bg field "notes" is true then
    set visible of bg field "notes" to false
    set hilite of target to false
  else
    hide bg field indexHolder
    hide bg button cover
    set hilite of bg button index to false
    set visible of bg field Notes to true
    set hilite of target to true
  end if
end mouseUp
```

4/4/89 6:48

Script of bkgnd field id 3 = "seeAlso"

on mouseUp

--Refer to script of this stack for more info about "clickLine ()"

put ClickLine() into lineNum

put line lineNum of bg field seeAlso into cardName

visual effect iris open

go to card cardName

end mouseUp

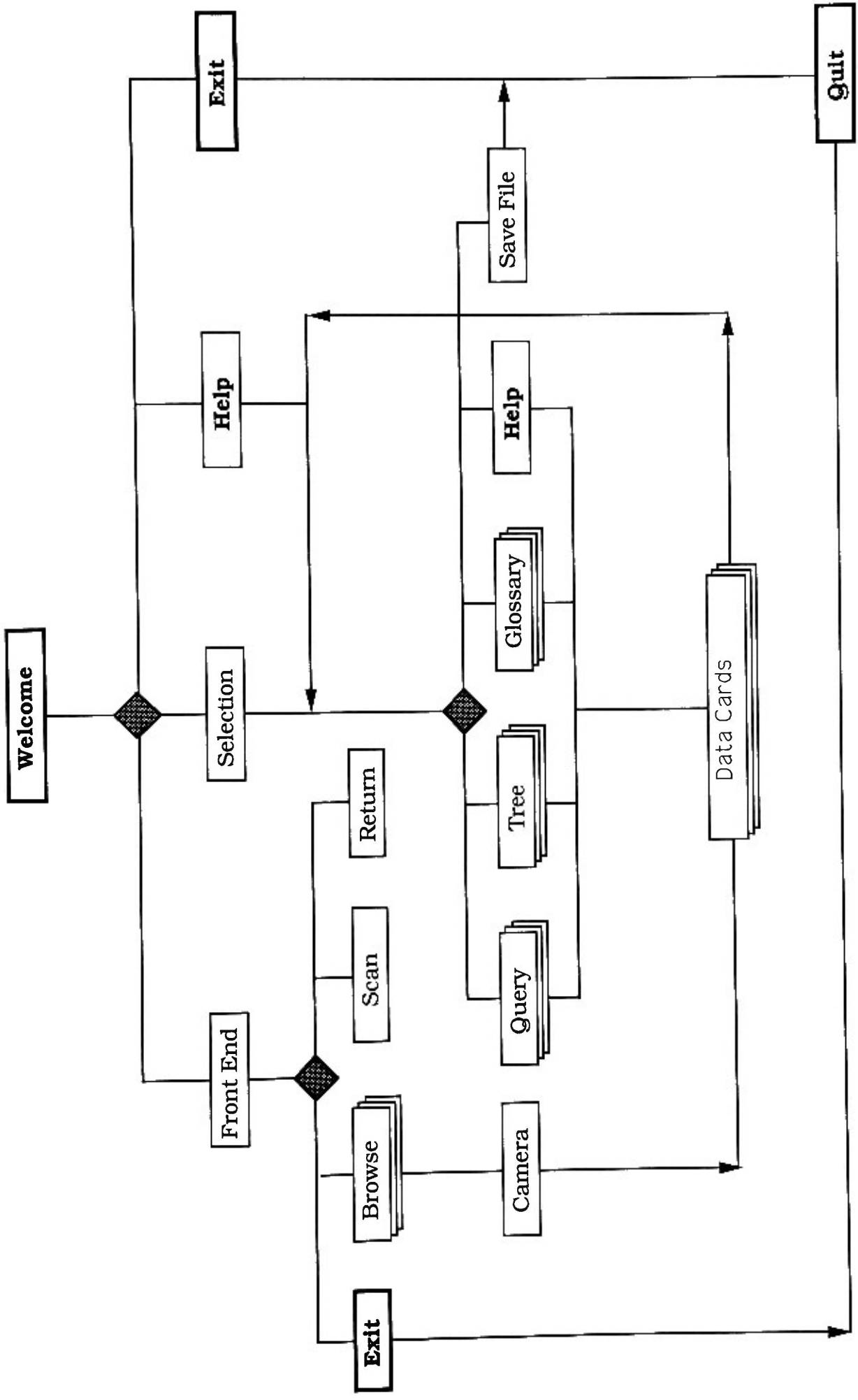
4/4/89 6:48

Script of bkgnd button id 38 = "Return"

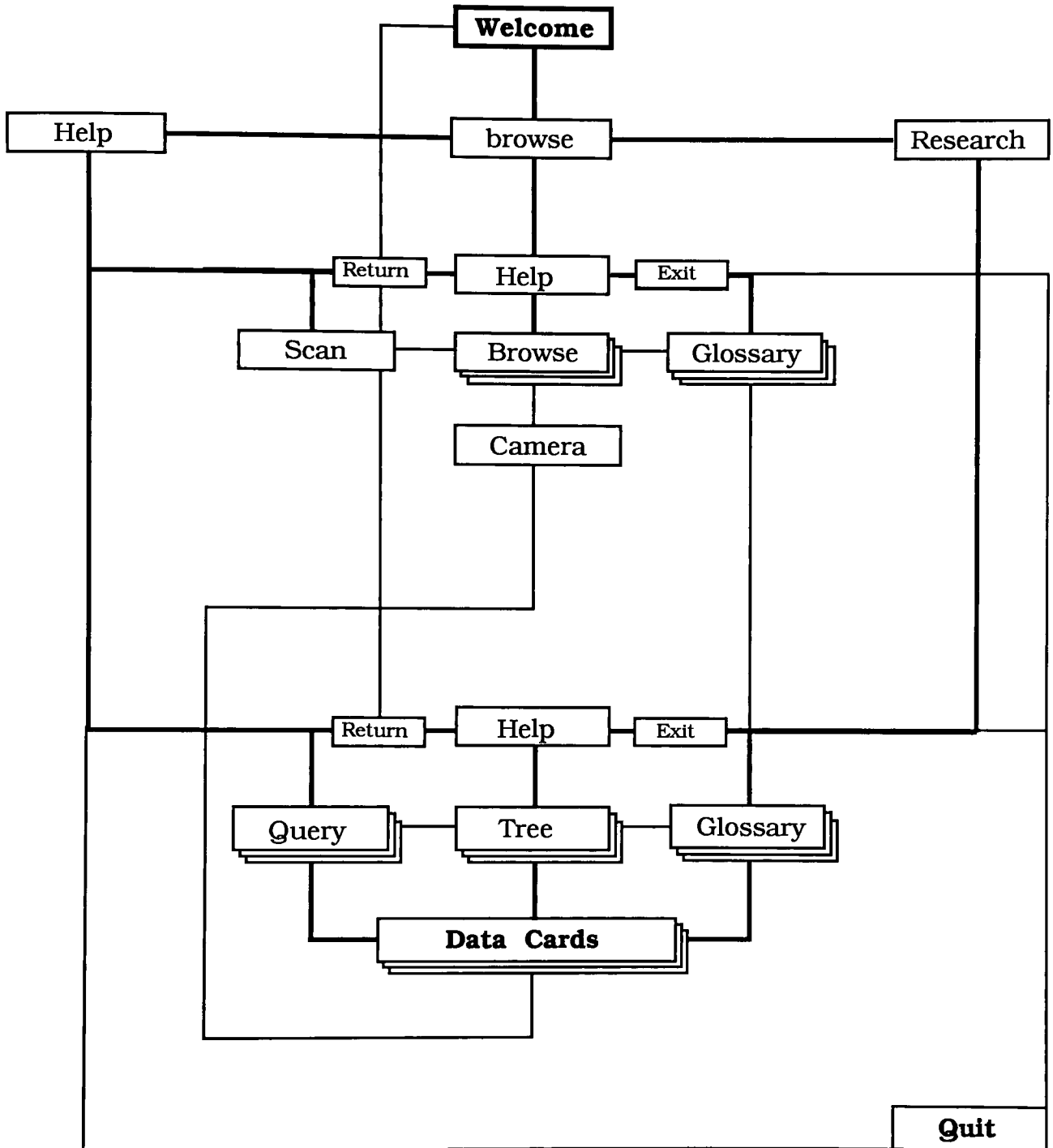
```
on mouseUp
  --From Reference
  set hilite of target to true
  visual effect iris close
  set hilite of target to false
  pop card
end mouseUp
```



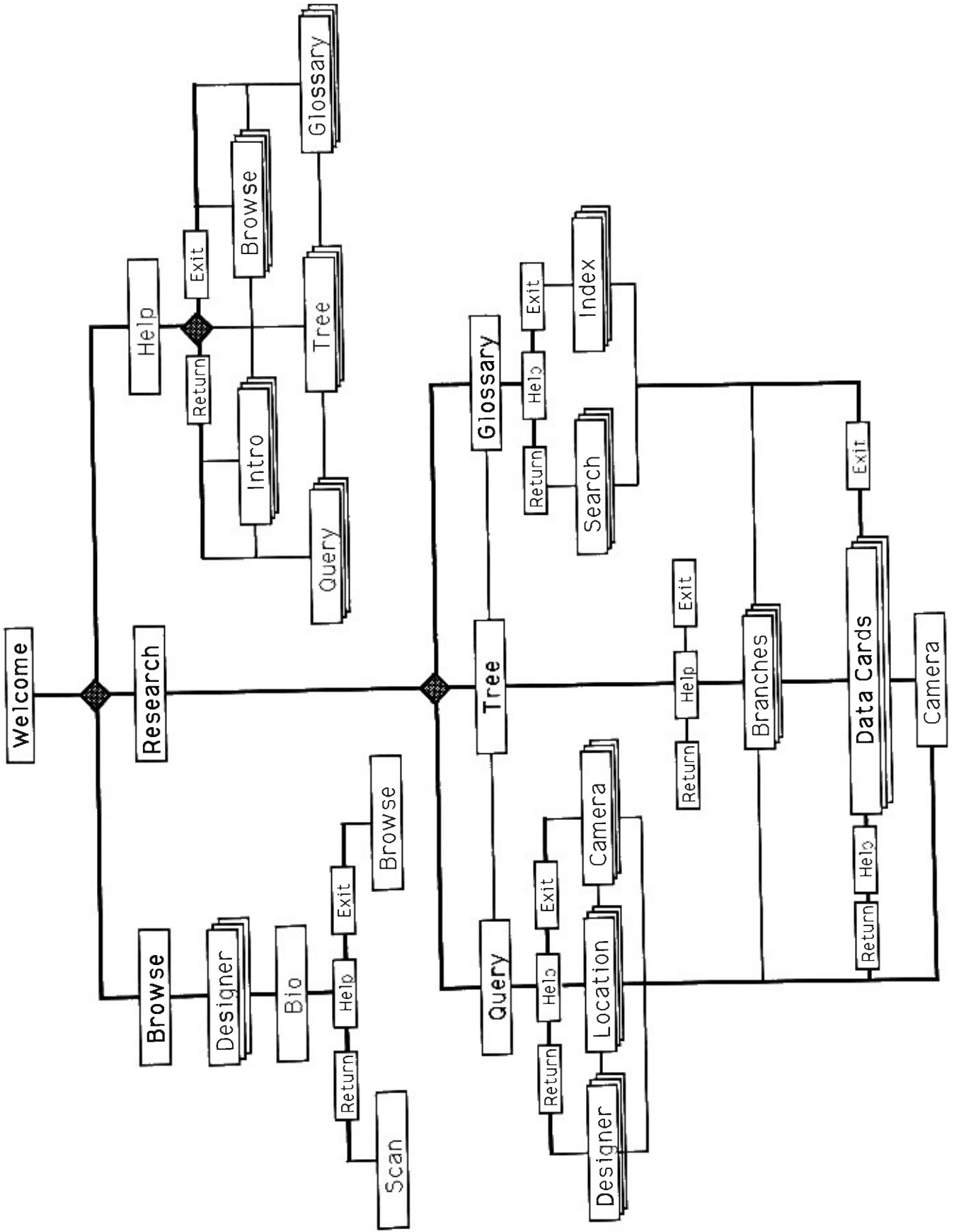
***Appendix C: Flow Charts***





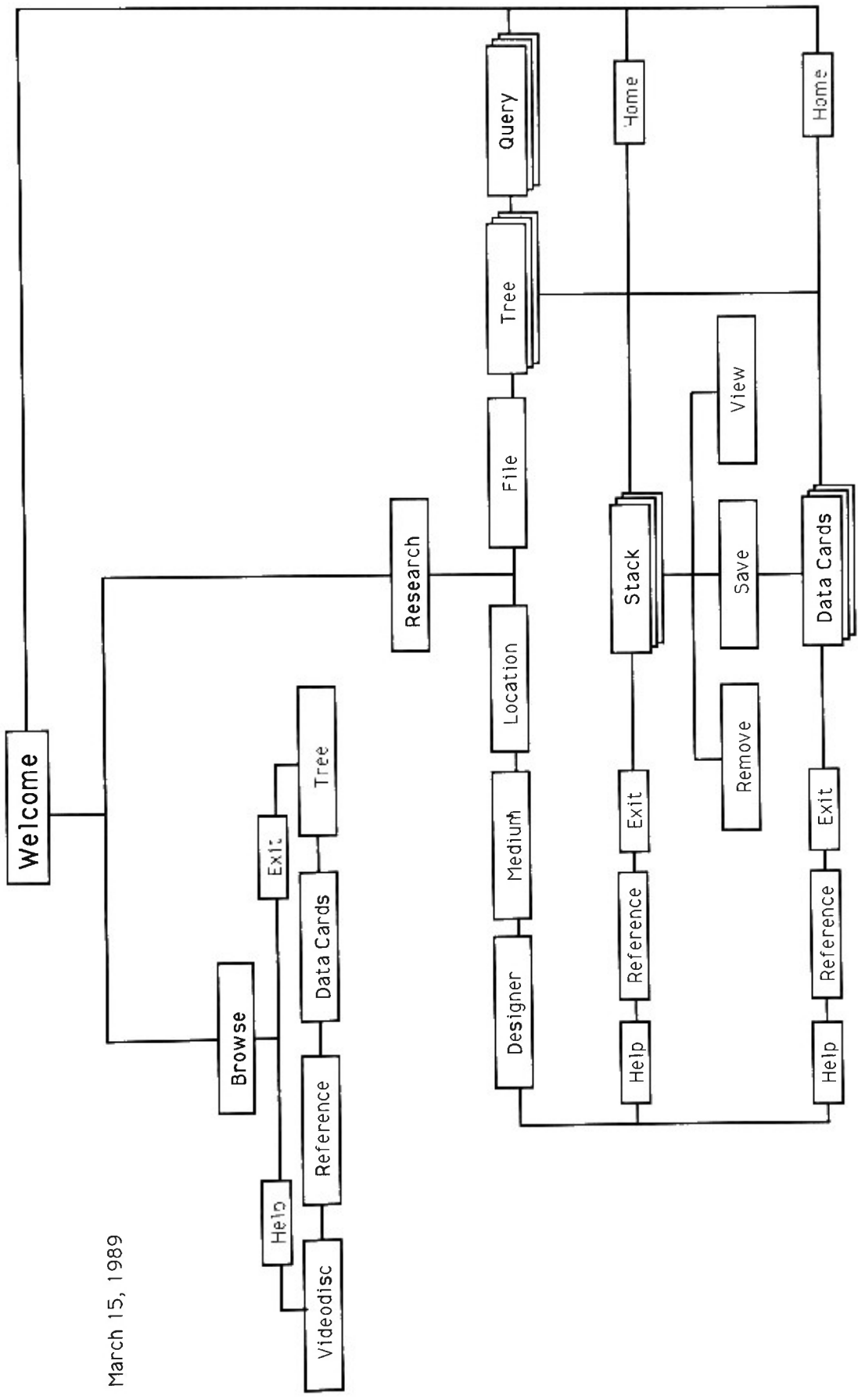


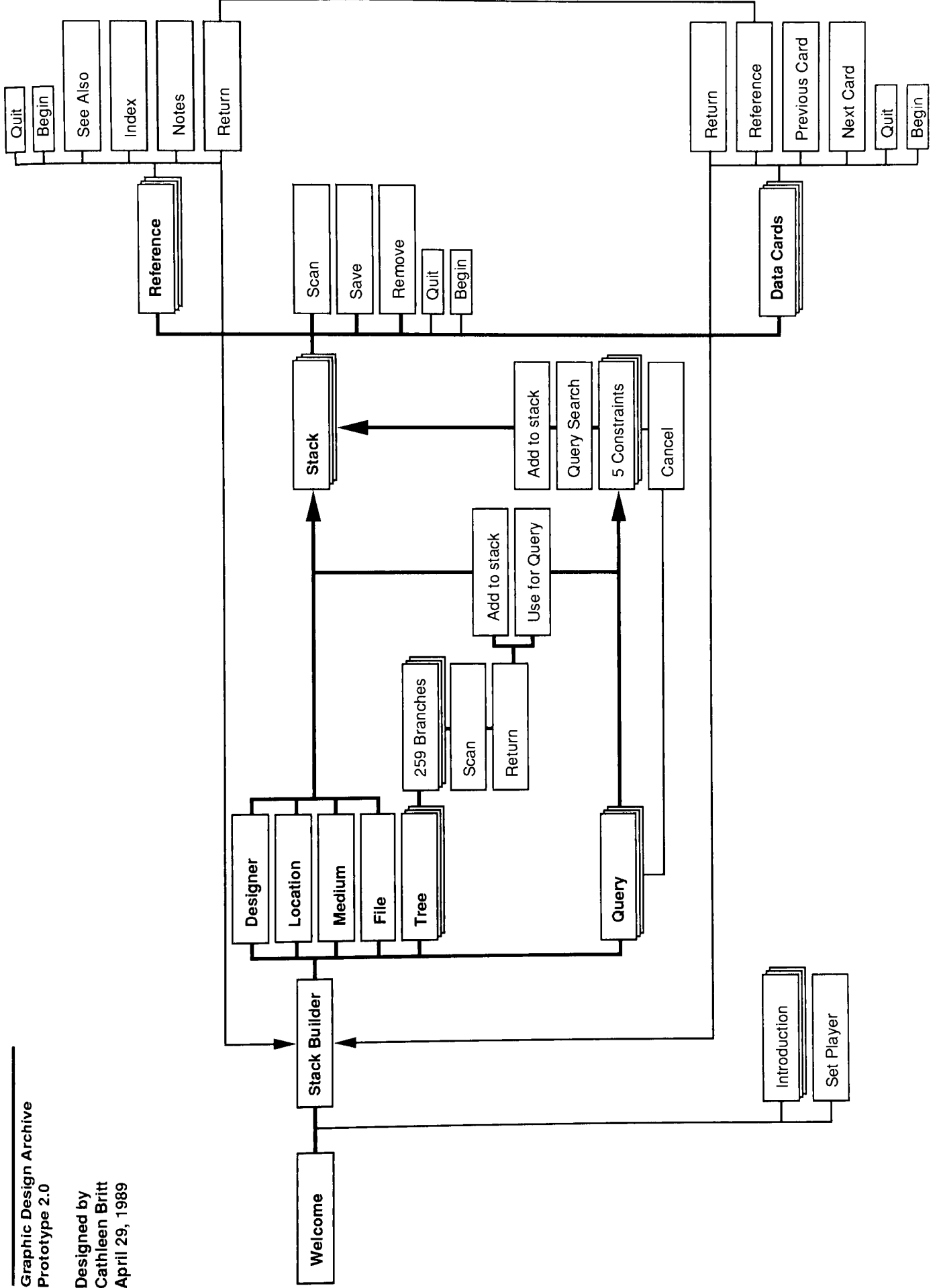
December 18, 1988





March 15, 1989

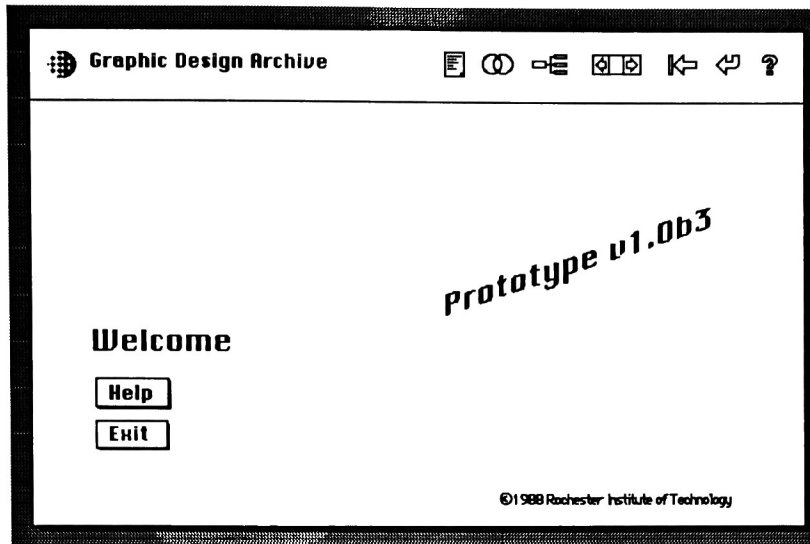




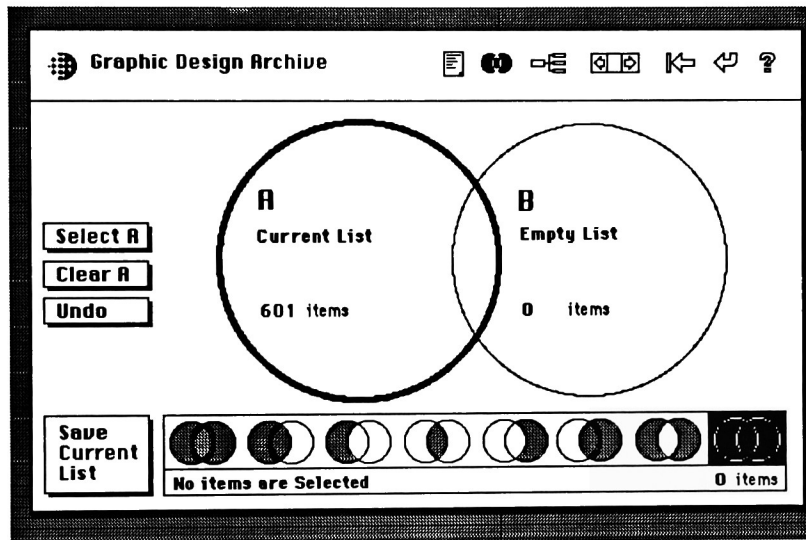


## ***Appendix D: Prototype 1.0 Screen Designs***

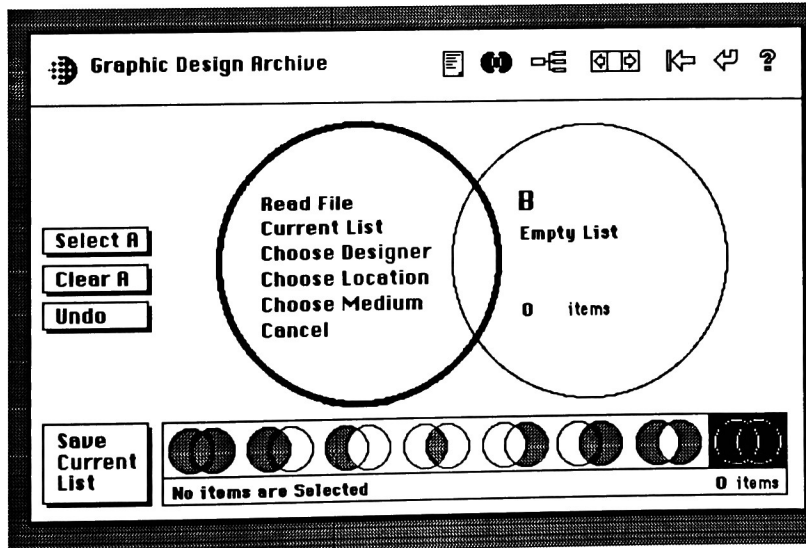
# Prototype 1.0



Welcome card



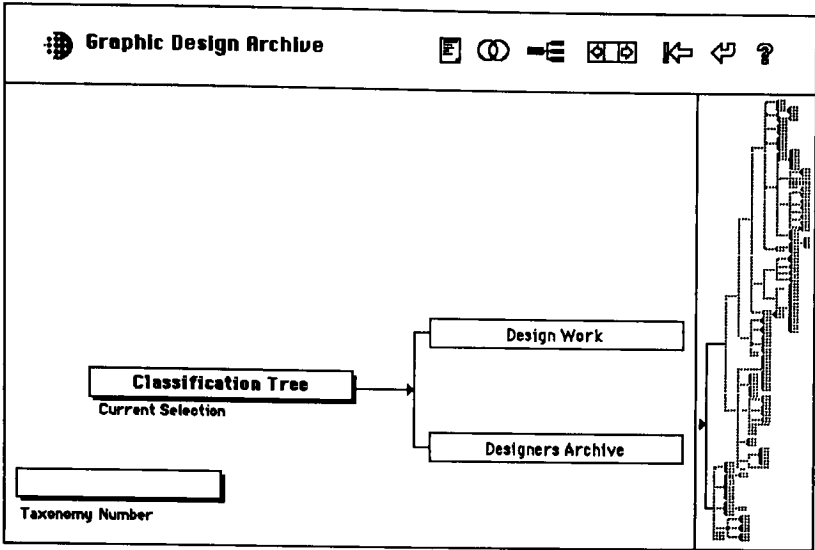
Query card



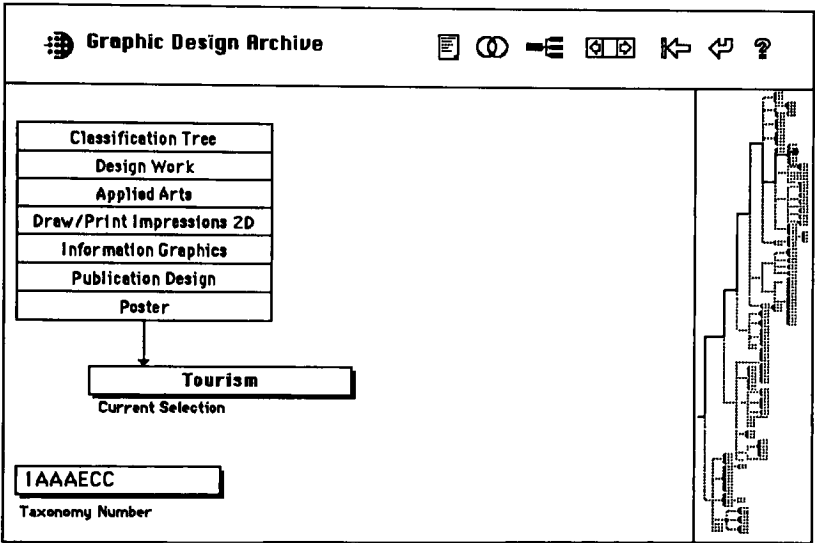
Query card



Prototype 1.0



Classification Tree



Classification Tree

Graphic Design Archive

Designer Bass, Saul Title Magazine page showing elements of identity program for Celanese

Date 1960 Location USA

Medium Printed image, lithography Comment \_\_\_\_\_

1

Data Card

Glossary Card

**Graphic Design Archive Glossary**

---

**Bayer, Herbert (b.1900)**

Bayer was born in Austria; he apprenticed there in an architectural studio before becoming a student at the Bauhaus in Weimar, 1921-1923. He taught advertising and typography at the Bauhaus in Dessau, 1925 to 1928, where he discouraged traditional typographic ornament and layout, advocated the use of sans serif type and all lower-case letters, and worked to integrate typography and photography. He emigrated to the United States in 1938. Bayer is known for his photography, publication design, and exhibition design for for the Container Corporation and other clients.

See also

- International Style
- Functionalism
- Constructivism

Notes    Search    Index

Glossary Card with See Also

**Graphic Design Archive Glossary**

---

**Bayer, Herbert (b.1900)**

Bayer was born in Austria; he apprenticed there in an architectural studio before becoming a student at the Bauhaus in Weimar, 1921-1923. He taught advertising and typography at the Bauhaus in Dessau, 1925 to 1928, where he discouraged traditional typographic ornament and layout, advocated the use of sans serif type and all lower-case letters, and worked to integrate typography and photography. He emigrated to the United States in 1938. Bayer is known for his photography, publication design, and exhibition design for for the Container Corporation and other clients.

Constructivism

Constructivism formed in Russia during the 1910s and 20s and became an internationally influential movement in art and design. Led by El Lissitzky, who travelled widely through Europe, the Constructivists advocated the use of geometry, photography, and mass-production to create an aesthetically and politically revolutionary mode of communication. The movement was defeated in the Soviet Union with the rise of Stalin. After WWII, Constructivism inspired Swiss modernism's interest in "programmatic" or systems-oriented painting and design.

Notes    Search    Index

Glossary Card with Notes

**Graphic Design Archive Glossary**

---

**Bayer, Herbert (b.1900)**

Bayer was born in Austria; he apprenticed there in an architectural studio before becoming a student at the Bauhaus in Weimar, 1921-1923. He taught advertising and typography at the Bauhaus in Dessau, 1925 to 1928, where he discouraged traditional typographic ornament and layout, advocated the use of sans serif type and all lower-case letters, and worked to integrate typography and photography. He emigrated to the United States in 1938. Bayer is known for his photography, publication design, and exhibition design for for the Container Corporation and other clients.

Constructivism

Constructivism formed in Russia during the 1910s and 20s and became an internationally influential movement in art and design. Led by El Lissitzky, who travelled widely through Europe, the Constructivists advocated the use of geometry, photography, and mass-production to create an aesthetically and politically revolutionary mode of communication. The movement was defeated in the Soviet Union with the rise of Stalin. After WWII, Constructivism inspired Swiss modernism's interest in "programmatic" or systems-oriented painting and design.

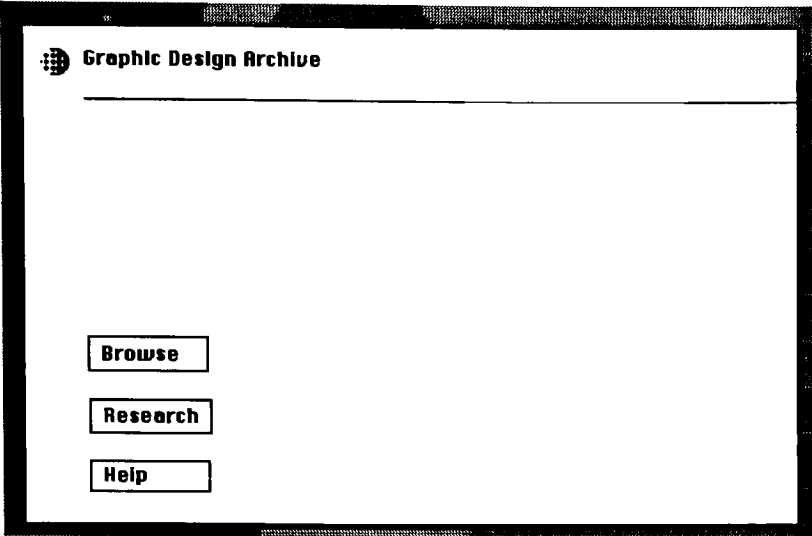
Source: Ades, Dawn. *The Twentieth Century Poster: Design of the Avant Garde*. NY, Minneapolis: Walker Art Center and Abbeville Press, 1984.

For more information, see Cohen, Arthur, *Herbert Bayer: Painter Designer Architect* (NY: Reinhold, 1967).

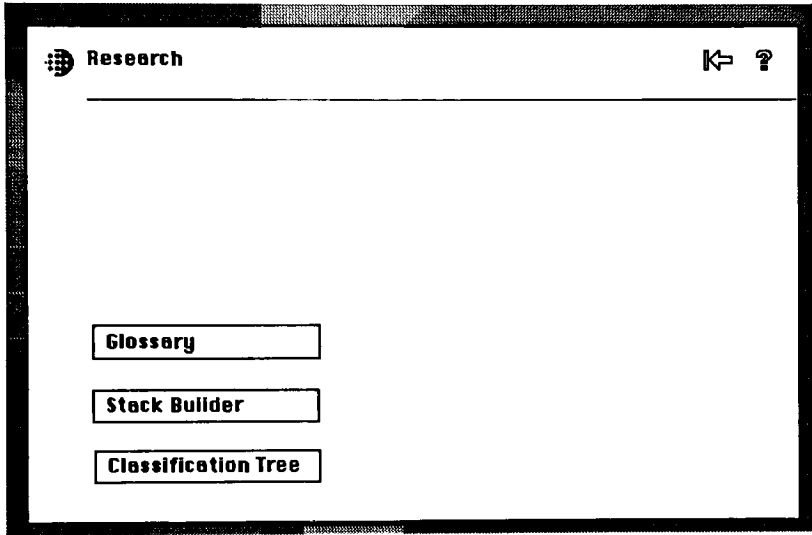
Notes    Search    Index



## ***Appendix E: February-March Development, Screen Designs***



Welcome Card



Research Card

Stack Builder Card

**Stack Builder Four**

The Stack Images

Add  
Remove  
Intersect

Designer  
Location  
Medium  
File  
Camera  
Tree  
Cancel

Clear Save Scan

Stack Builder Card

**Stack Builder Four**

The Stack Images

Add  
Remove  
Intersect

Designer  
Location  
Medium  
File  
Camera  
Tree  
Cancel

Bass, Saul  
Bayer, Herbert  
Beall, Lester  
Binder, Joseph  
Brodevitch, Alexey  
Burtin, Will  
Cassandre, A. Mouron  
Eames, Charles  
Golden, William  
Guisti, George  
Kauffer, E. McKnight  
Kepes, Gyorgy  
Lionni, Leo

Clear Save Scan

Stack Builder Card with Query function

**Stack Builder Four**

The Stack Images

Add  
Remove  
Intersect

Designer  
Location  
Medium  
File  
Camera  
Tree  
Cancel

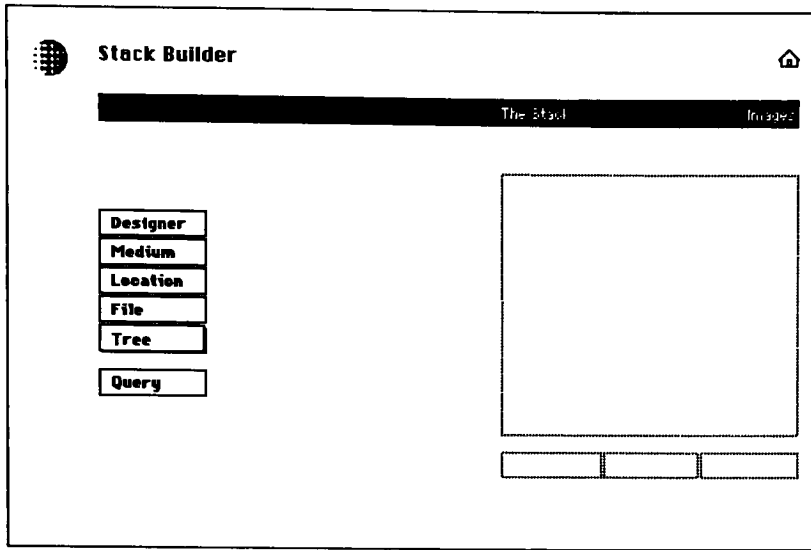
One

Brodevitch, Alexey 64

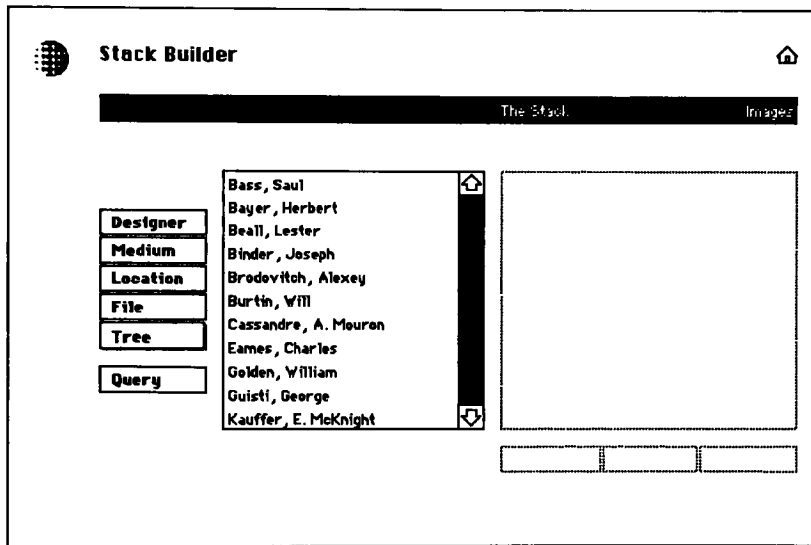
OK

Two

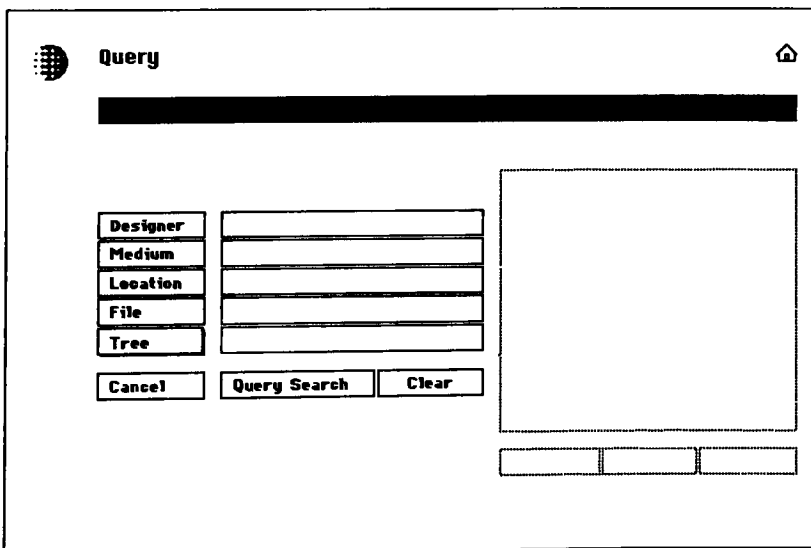
Clear Save Scan



Stack Builder Card

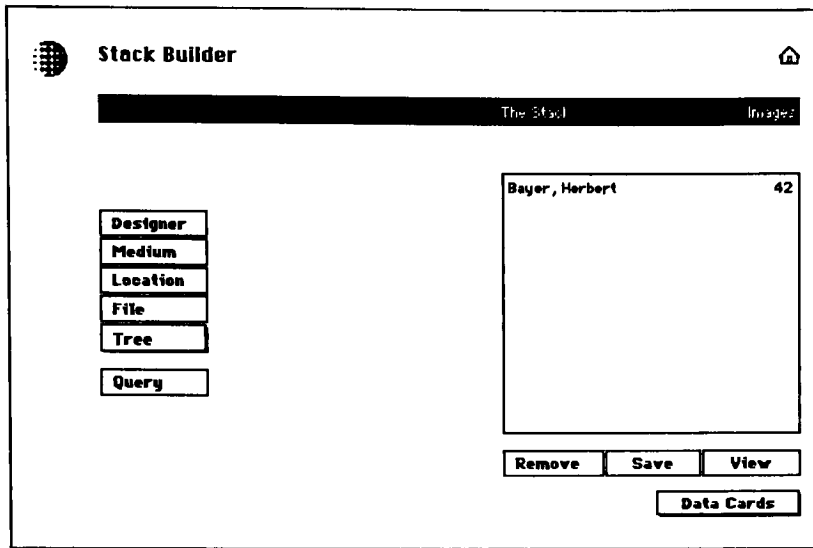


Stack Builder Card

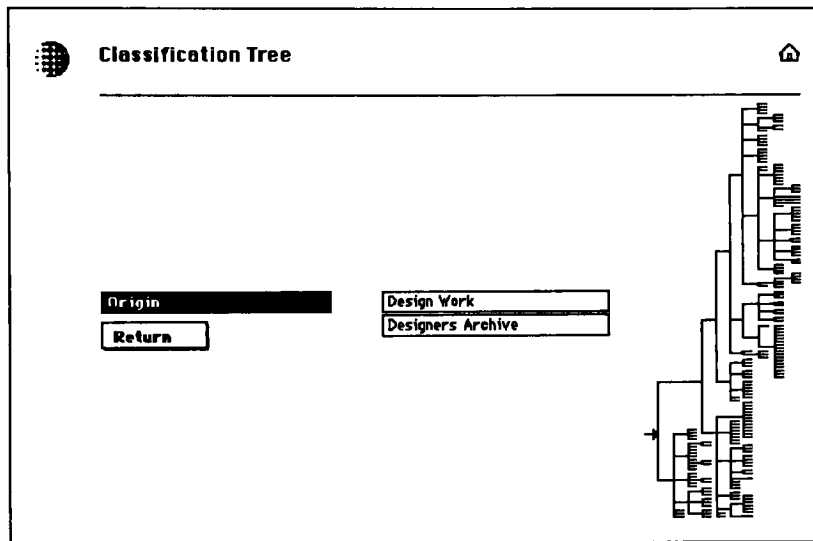


Stack Builder Card with Query Function

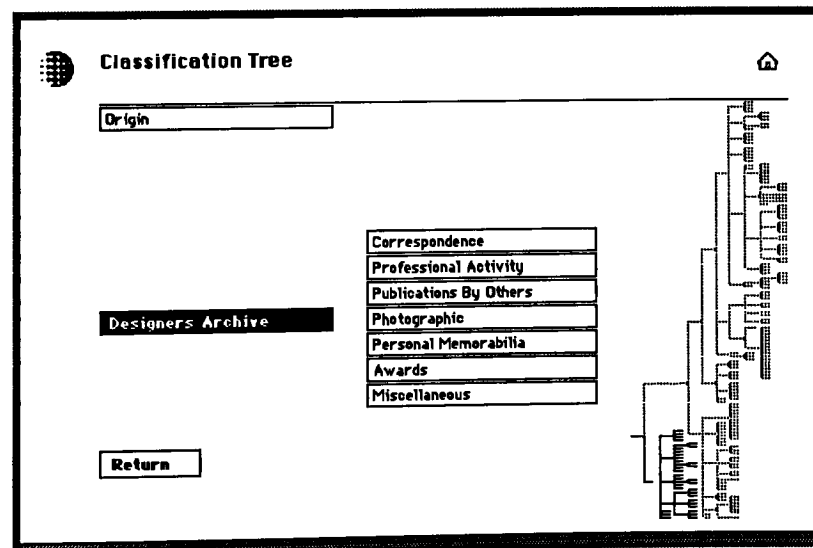
Stack Builder Card



Classification Tree



Classification Tree





# Early March Demo

**Stack Builder** 🏠 🏠 🏠 🏠 🏠 ?

The Stack Images

Designer  
Location  
Medium

Tree  
File  
Query

Remove Save Video

Stack Builder Card

**Query** 🏠 🏠 🏠 🏠 ?

Query Expression

Designer  
Location  
Medium

Tree  
File  
Stack

and or

and or

and or

Find Result Clear

Stack Builder Card with Query Function




## ***Appendix F: Prototype 2.0 Screen Designs***

Welcome Card



First Interactive Card

 **Graphic Design Archive**

**Prototype 2.0**

**Stack Builder** Click the **Stack Builder** button to begin.

**Introduction** Click the **Introduction** button to find out more about this prototype.

**Set Player** If you are using a videodisc player, click the **Set Player** button to set to the correct player type.


The Graphic Design Archive is an electronic desktop museum of the history of graphic design.

The purpose of the project is to provide an expansive database of text and image frames that can be accessed interactively for educational and informational purposes.

Graphic Design Archive © 1989  
All Rights Reserved

**QUIT**


Introduction

 **Graphic Design Archive**

**Introduction**

The focus of the prototype is the following group of designers who worked between 1930 and 1955:

Saul Bass	E. McKnight Kauffer
Herbert Bayer	Gyorgy Kepes
Lester Beall	Leo Lionni
Joseph Binder	Alvin Lustig
Alexey Brodovitch	Herbert Matter
Will Burtin	Paul Rand
A. M. Cassandre	Ladislav Sutnar
Charles Eames	Bradbury Thompson
William Golden	Jan Tschichold

 **Return**

**QUIT**

Stack Builder Card

**Stack Builder**

The Stack Images

Designer  
Location  
Medium  
File  
Tree  
Query

Search Save Remove

Ref. Cards

START QUIT

Stack Builder Card

**Stack Builder**

The Stack Images

Designer Bass, Saul  
Location Bayer, Herbert  
Medium Beall, Lester  
File Binder, Joseph  
Tree Brodovitch, Alexey  
Query Burtin, Will  
Cassandre, A. Mouron  
Eames, Charles  
Golden, William  
Guisti, George  
Kauffer, E. McKnight  
Kepes, Gyorgy

Search Save Remove

Ref. Cards

START QUIT

Stack Builder Card with Query Function

**Stack Builder**

Query Constraints

Designer  
Location  
Medium  
File  
Tree

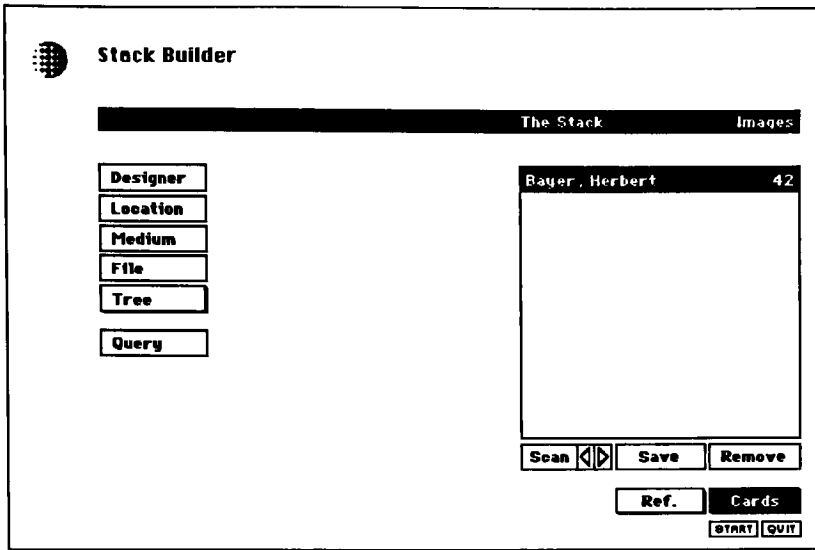
Cancel Query Search Remove

Search Save Remove

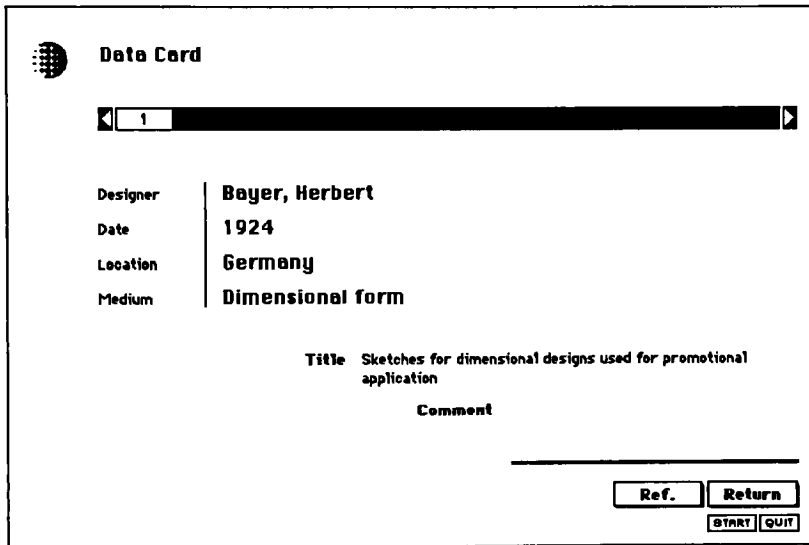
Ref. Cards

START QUIT

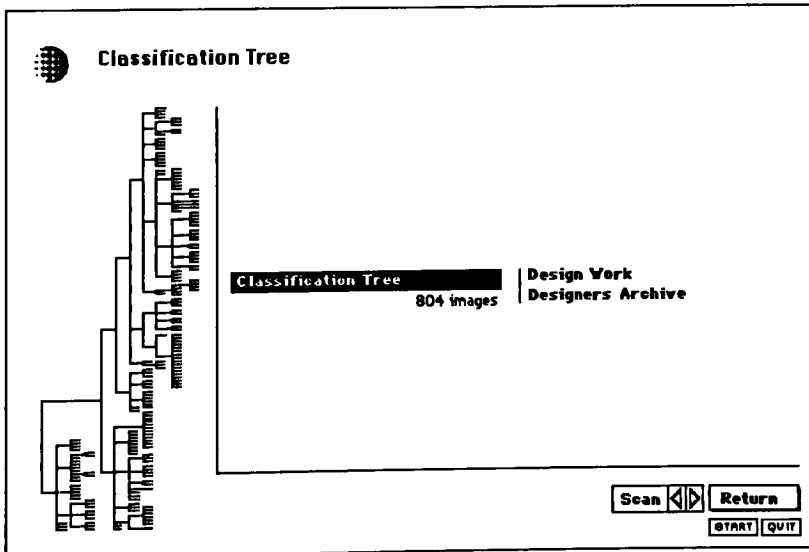
Stack Builder Card




Data Card



Classification Tree



Reference Card


 **Reference**

**Bayer, Herbert** **See Also**

Herbert Bayer (b.1900-d.1985) was born in Austria and apprenticed there in an architectural studio before becoming a student at the Bauhaus in Weimar in 1921. He taught advertising and typography at the Bauhaus in Dessau between 1925 to 1928. He discouraged traditional typographic ornament and layout, instead advocating the use of sans serif type and all lower-case letters. He worked to integrate typography and photography. He immigrated to the United States in 1938. Although Bayer wanted to be remembered as a painter, he is known primarily for his publication design, photography, and exhibition design for American clients such as Container Corporation of America, ARCO, Fortune magazine and others.

**Bauhaus**  
**Container Corp. Of America**  
**Constructivism**  
**International Style**  
**Functionalism**

Reference Card with Index


 **Reference**

**Bayer, Herbert** **See Also**

Herbert Bayer (b.1900-d.1985) was born in Austria and apprenticed there in an architectural studio before becoming a student at the Bauhaus in Weimar in 1921. He taught advertising and typography at the Bauhaus in Dessau between 1925 to 1928. He discouraged traditional typographic ornament and layout, instead advocating the use of sans serif type and all lower-case letters. He worked to integrate typography and photography. He immigrated to the United States in 1938. Although Bayer wanted to be remembered as a painter, he is known primarily for his publication design, photography, and exhibition design for American clients such as Container Corporation of America, ARCO, Fortune magazine and others.

American School  
 Art Nouveau  
 Arts and Crafts Movement  
 Bass, Saul  
 Bauhaus  
 Bayer, Herbert  
 Beall, Lester  
 Binder, Joseph  
 Bradley, William  
 Brodovitch, Alexey  
 Burtin, Will  
 Cassandre, A. Mouron  
 Constructivism  
 Container Corp. of America  
 Cubism  
 de Stijl

Reference Card with Notes

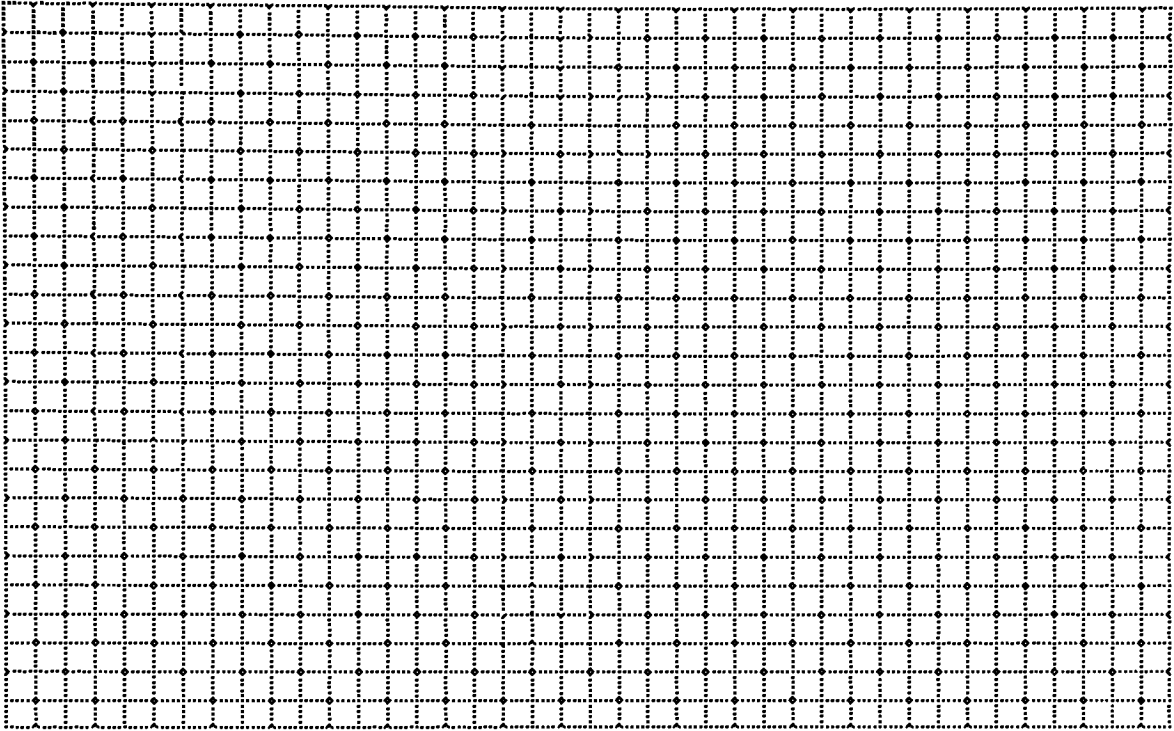
 **Reference**

**Bayer, Herbert** **See Also**

Herbert Bayer (b.1900-d.1985) was born in Austria and apprenticed there in an architectural studio before becoming a student at the Bauhaus in Weimar in 1921. He taught advertising and typography at the Bauhaus in Dessau between 1925 to 1928. He discouraged traditional typographic ornament and layout, instead advocating the use of sans serif type and all lower-case letters. He worked to integrate typography and photography. He immigrated to the United States in 1938. Although Bayer wanted to be remembered as a painter, he is known primarily for his publication design, photography, and exhibition design for American clients such as Container Corporation of America, ARCO, Fortune magazine and others.

Ades, Dawn,  
 "The Twentieth Century Poster:  
 Design of the Avant Garde"  
 (New York, Minneapolis: Walker Art  
 Center and Abbeville Press, 1984).

Cohen, Arthur,  
 "Herbert Bayer: Painter Designer  
 Architect"  
 (New York: Reinhold, 1967).







## ***Appendix G: Prototype 2.0 Evaluation Documents***

## ***Script For Pre-Evaluation Software Demonstration 4/6/89***

The Graphic Design Archive consists of approximately 4000 slides from Roger Remington's collection, that are concerned with the history of graphic design. The slides were photographed on 35mm motion picture film. The film was then transferred to one inch videotape and sent to a production house in NY to be pressed onto the videodisc.

What you'll be seeing today is just a small sampling from those 4000 images. In the hyperCard prototype that I have built there exists data for 806 images done by 20 different designers that worked in New York between 1930 and 1955. My intention in having you test the system is simply to begin the process of evaluating how well the software works as an accessing tool.

There is really only one simple rule in using the software and that is that you must choose what it is that you would like to see. On the computer screen in front of you is what I call the Stack Builder card. On the left is a menu of the options available and on the right, in gray, is the area where you will be storing your selections. I think of the menu as a sort of a card catalogue and the stack as my desktop where I can lay out and browse through my selections.

The different menu options are what hyperCard refers to as buttons.

1. Click once now on the Designer button. A Field will appear with a list of the names of the 20 designers that are represented by images in the database. The arrows to the right of the field allow you to move up or down the list as needed.

2. Click once on any one of these names and your selection will appear in the newly hilited stack along with the number of images that are in the database by that designer. You can use the same process to select images by location or by medium. If you have clicked on buttons Designer, Location, or Medium by mistake, click once more on the same button and the scrolling field will disappear.

3. The File button can be used to read in a previously saved file. However, we won't be testing the file or save buttons so you can ignore them for now.

4. Notice the shadow surrounding the Tree button. The shadow indicates that clicking on that button will take you to another application that has a function different from that of the "stack builder" card. Click now on the Tree button.

The Classification Tree is a system, developed by Roger Remington, of organizing all the images in the database according to the type of work that these images represent. On the left is a map of the entire Tree. Notice the flashing arrow; it indicates your current location in the tree. The name of the branch is in the reversed bar in the center of the card.

5. The lines to the right of the bar are the branches you have to choose from. Click on one of them to get to the next lower branch. Notice now that the map has a gray area and a hilited area. The hilitate indicates branches that are still accessible, the gray indicates branches that are no longer accessible in that part of the tree.

6. Notice also that your previous location is recorded in plain type above the current location bar. Click on these to backup and begin again. When you get to a branch that interests you, there are 2 choices:

- a. Scan the images with a click of the scan button in the lower left corner. Click anywhere to stop the scan. Proceed one image at a time with the arrow buttons.

**b.** Return to the stack builder card by a click of the Return button in the lower right corner. After a moment a box will appear and ask you whether or not you would like to add this information to the stack. Click "yes" and you will return with the images to the stack builder card, "No" to return without the images, and "Cancel" to stay put in the tree.

Once you have returned to the stack builder card and have some selections in the stack, you have 5 choices. Click once on an item in the stack to select it. Double clicking on the stack will select everything. Once you have something selected:

1. Click the Scan button to quickly view the selected images or use the arrow buttons to view them one at a time.
2. Use the Save button to save a file (not necessary for this evaluation.)
3. Click on the Remove button to delete selected items from the Stack.
4. Click on the Cards button (notice the shadow) and you will go to the data cards that describe the selected images.

The data cards have a black bar at the top with a numbered box inside, and arrows at either end. This is the scroll bar. The number refers to the the number of the image you are currently viewing on the video monitor. Click on the box and drag it, with the mouse down to the right. Notice that the number changes. When you let the mouse up that number image will appear on the video monitor and the data card that describes that image will appear on the Macintosh screen. The arrows at either end of the scrollbar allow you to proceed one at a time through the images.

5. The Reference button is available on both the stack builder card and the data cards. Click on the reference button from the data cards and you will bring up a card that contains biographical information about the designer whose work is currently showing.

The index button on the reference card brings up a scrolling list of all the terms in that we now have information about in the database. Click on one of these terms to get to a card that contains information about it. Bold terms to the right can be clicked on to get to other information. Notes brings up a box of miscellaneous information. Click again on the notes button to hide the note box.

Click on the Return button to go back to the data cards.

Click Return on the data cards to get back to the stack builder card.

The Query button is now the only menu item we have not investigated. This is the tool to use to search the database for more specific information. For instance, I would like to find all the magazine ads designed by Paul Rand.

1. Click on the Query button, five boxes appear to the right of the menu. Notice the Cancel button and the Remove button (functions the same as the Remove button for the Stack)
2. Click Designer and choose Rand, Paul. (Notice your selection appears in the box to the right of designer.)
3. Then click on Tree button to get to the Classification Tree and follow this path:  
Design Work  
Applied Arts  
Draw/Print Impressions 2D  
Marketing Graphics

Advertisement Magazine is an end branch of the tree, there are 107 images in this branch. Click Return and "Yes" to using 107 images for a query constraint.

**4.** Back on the Stack builder card (Notice Magazine in the box to the right of the Tree button) click the Query Search button.

**5.** Click on Add To Stack to add the result of the query into the stack. You will have to type in a name (less than 25 characters) for the query result and click OK. You have just completed a successful query function.

You are now ready to do the assignment!

## **Comments on Evaluative Results - Number of Correct Answers**

1. Overall, first time users had an easy time with the software, even with the more advanced functions involved in the Query procedure. One test subject, in particular, impressed me because of her lack of Macintosh experience. With some help from Susan in getting started, she answered all the questions correctly and finished before the others in her group.

2. As a result of the evaluation, a problem with the programming in the scan button became apparent and eventually pointed to another problem with the data on some of the cards. There were some missing frame numbers. These problems have since been resolved.

I noticed that some of the test subjects did things in an order that was unanticipated (leaving more than 1 button at a time highlighted.) I will have to make some additions to the program to catch for these variables.

3. It became obvious during the testing that some of the terms used in the list of mediums and the Classification Tree are somewhat ambiguous. Many of the test subjects looked for the "Dimensional Designs" medium in the Classification Tree, rather than in the list of mediums on the Stack Builder card. Others had a hard time finding information in the tree. It is not obvious, for instance, where to find "alaphabet designs" or "tourism" posters. Most of them did find the information (some had help others used trial and error.)

### Organizational Hints...

**Classification Tree** classifies images according to type of work.

**Data Cards** contain factual info. about video images (designer, location, date, title, comments)

**Reference** contains biographical information about designers, general information about movements, companies ect...

---

### Graphic Design Archive Evaluative Questions (number of correct answers out of 13)

1. Name the **Designer** whose last name begins with "N." \_\_\_\_\_ 13/13
  - a.) How many images in the database were done by this designer?\_ 13/13
  - b.) The data cards indicate that these images include a series of posters designed for one client. Name the client. \_\_\_\_\_ 13/13
  - c.) What is the name of this series of work? \_\_\_\_\_ 13/13
  
2. How many images in the database are listed as having been done in the **Dimensional Designs** medium? \_\_\_\_\_ Only after giving further instructions 12/13
  - a.) Who designed the last one? \_\_\_\_\_ 12/13
  - b.) For what client? \_\_\_\_\_ 12/13
  - c.) What year was this designer born? \_\_\_\_\_ 11/13
  - d.) What year did he move to New York? \_\_\_\_\_ 11/13
  
3. How many images are included in the **Designers Archive** branch of the **Classification Tree**? \_\_\_\_\_ 12/13
  - a.) How many of these are photographs of designers? \_\_\_\_\_ 8/13
  - b.) Who is the designer who is smoking a pipe?  
Only after further instructions \_\_\_\_\_ 13/13
  - c.) Who took this photo? \_\_\_\_\_ 13/13
  
4. Use the **Query** function on the **Stack Builder** card and the **Tree** to find...
  - a.) How many **Alphabet Designs** were done by Herbert Bayer? \_\_ 12/13
  - b.) The name of one of these **Alphabet Designs**? \_\_\_\_\_ 12/13
  - c.) The date of this design? \_\_\_\_\_ 12/13
  
5. How many **Tourism** posters were done by **Herbert Matter**? \_\_\_\_\_ Optional question 7/13
  - a.) Name one of the countries that these posters promoted? \_\_\_\_\_ 7/13

46 Warrington Dr.  
Rochester, New York 14618  
716 (442-8140)

Roger Remington  
Department of Graphic Design  
College of Fine and Applied Arts

Re: Graphic Design Archive on Videodisc

Dear Roger:

Enclosed is a summary of the prototype evaluation of the Graphic Design Archive. I enjoyed working with the many fine individuals that comprise the team. The evaluation was very successful and a step in the right direction for the continued refinement on the prototype.

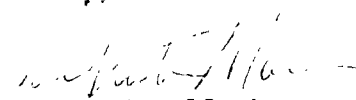
From the standpoint of computer-human interaction the success of the prototype is documented by the ability of novice and casual users to correctly obtain the information asked for in the worksheet. The positive response on the questionnaire concerning subjects perceptions of the prototype, also indicates that the interface is successfully interactive. When the interface is working the medium is not the message, but rather becomes a desktop metaphor. As more involved data is asked for it is important that the interface remains as invisible as possible.

The next step in the evaluation procedure is to get more involved in how well individuals are able to problem solve using the prototype with interactive data. Delving into the individual's ability to process and utilize all



the interactive data that is presented. As we discussed this would be a good focus for the summer evaluation project.

Sincerely,



Susan Preston-Mauks

On April 6, 1989, R.I.T Graphic design students took part in an evaluative session designed to study the human-machine interface with a *Prototype* in the *Graphic Design Archive*. The interface was evaluated for both presentation and educational application. It is important here to note that, the design of the user interface should take into consideration the ability of the human mind to process and store information. Information, was defined in this study as a collection of data contained in the software and on the videodisc. The data includes 806 images on video disc that are accessed via a customized Hypercard program. This collection came from 4,000 images contained on the first **Graphic Design Archive Videodisc**. The Archive is designed as an interactive, multifunctional resource for students, teachers, researchers, professional designers and other interested individuals. As an educational tool the interface plays an important communication role, connecting the user to the information, and assisting the user in generating data for their own needs.

Although several informal evaluation sessions had been conducted previously, a formal evaluation session was needed for further refinement on the **prototype 2**. Graphic Design students were selected from current design classes. These subjects filled out a pre testing questionnaire concerning their age, sex, computer experience and grade point in college., then randomly divided into groups of three and assigned a time slot for evaluating the prototype.

The experiment proved very successful in several areas. First, information concerning problems moving within the program were discovered and corrected. Other refinements based on feedback from the experiment are also being implemented to improve clarity. Second, the ability to process the information contained in a multimedia context produced few errors, especially in the first few questions that involved simple interaction with the computer( 99%).In the next several questions that involved sequential processing the correct response was also high (98%) . Only the last question labeled optional had a low correct response rate (53% correct). Of the 47% incorrect, (26%) of the answers on the last were left blank due to time restraints. Thirdly, subjects reported an overall positive perception of the software in all seven areas. Perceptions of frustration utilizing the interface between fluent users and occasional users was very low. Results seemed to indicate that the interface was simple enough for any ability user and served as a facilitator, rather than a inhibitor in accessing information. Subjects also indicated a high level of understanding in working with the prototype. This perceived understanding indicated a well designed prototype for use as an educational tool.

This evaluation was the first of a series to study the parameters of the program. Other evaluations planned include delving further into the interaction between the user and the interface when searching for data as well as, the ability of the prototype 2 to help develop relationships among items of information..

Three workstations were set up for the test. Utilizing a MacIntosh II and two MacIntosh SE computers; a video-disc player and the prototype video disc, color video monitor and the prototype 2 software.

Two individuals, one the designer of the software and the project evaluation consultant served as facilitators in the experiment. Each group received a hands on guided tour. These subjects then received a worksheet containing questions that would only be answerable using the software, and videodisc. Completion of the worksheet was allowed in any order, however questions were designed to move from easy to more complicated cognitive processing.

The questions were designed to evaluate five functions of the interface and software. These included the Browse, Stack, Classification Tree, Query Function, and Database. A data collection procedure was used to record response time, stops, and questions asked. It was intended that feedback received from problems encountered with the interface would allow for additional refinement of the interface.

Along with analyzing how accurate the interface was in leading students to information, it also seemed appropriate to query the students in regard to their perceptions of the interface and software. This was done by giving them a questionnaire to complete after they had finished the worksheet. The questionnaire was designed to survey subjects perceptions of their experience with the prototype. Seven perceptions were queried including, frustration, attention, understanding, graphics knowledge, communication, and future usage. Respondents were requested to indicate their degree of agreement or disagreement with items on a five-point scale from agree strongly to disagree strongly

Name \_\_\_\_\_

Please circle the letter that is nearest to an accurate statement for you!

1. My age as of January 1, 1989 was

- (a) less than 18 (b) 18 (c) 19 (d) 20 (e) older than 20

2. I am a

- (a) male (b) female

3. How do you rate your computer experience?

- (a) casual user (b) fluent (c) very fluent

4. What kind of computer do you use?

- (a) none (b) Macintosh (c) IBM compatible (d) VAX system (e) other

5. My overall grade point average is

- (a) 2.0 or below (b) 2.1-2.5 (c) 2.6-3.0 (d) 3.1-4.0 (e) don't know

## Hints...

Tree classifies images according to type of work.

**Data Cards** contain factual info. about video images (designer, location, date, title, comments)

**Reference** contains biographical information about designers, general information about movements, companies ect...

## Graphic Design Archive Evaluative Questions

1. Name the **Designer** whose last name begins with "N." \_\_\_\_\_

a.) How many images in the database were done by this designer? \_\_\_\_\_

b.) The data cards indicate that these images include a series of posters designed for one client. Name the client. \_\_\_\_\_

c.) What is the name of this series of work? \_\_\_\_\_

*look under medium on stack builder card*

2. How many images in the database are listed as having been done in the **Dimensional Designs** medium? \_\_\_\_\_

a.) Who designed the last one? \_\_\_\_\_

b.) For what client? \_\_\_\_\_

c.) What year was this designer born? \_\_\_\_\_

d.) What year did he move to New York? \_\_\_\_\_

3. How many images are included in the **Designers Archive** branch of the **Classification Tree**? \_\_\_\_\_

a.) How many of these are photographs of designers? \_\_\_\_\_

b.) Who is the designer who is smoking a pipe? \_\_\_\_\_

c.) Who took this photo? \_\_\_\_\_

4. Use the **Query** function on the **Stack Builder** card and the **Tree** to find...

a.) How many **Alphabet Designs** were done by Herbert Bayer? \_\_\_\_\_

b.) The name of one of these **Alphabet Designs**? \_\_\_\_\_

c.) The date of this design? \_\_\_\_\_

---

5. How many **Tourism** posters were done by Herbert Matter? \_\_\_\_\_

a.) Name one of the countries that these posters promoted? \_\_\_\_\_

## QUESTIONNAIRE;

### GRAPHIC DESIGN ARCHIVE PROTOTYPE ÉVALUATION: APRIL 1989

BELOW IS A LIST OF 10 QUESTIONS CONCERNING THE GRAPHIC DESIGN ARCHIVE PROGRAM THAT YOU JUST COMPLETED. PLEASE CIRCLE THE RESPONSE TO EACH QUESTION THAT APPLIES, BASED ON YOUR EXPERIENCE WITH THE ASSIGNMENT. THANK YOU FOR YOUR ASSISTANCE.

1. THIS PROGRAM WAS MORE INTERESTING THAN I THOUGHT IT WOULD BE .

1.strongly agree    2.agree    3. neither agree, nor disagree    4. disagree    5.disagree strongly

2. THIS PROTOTYPE IS AN EFFECTIVE WAY TO COMMUNICATE INFORMATION ABOUT GRAPHIC DESIGNERS.

1.strongly agree    2.agree    3. neither agree, nor disagree    4. disagree    5.disagree strongly

3. I GOT FRUSTRATED TRYING TO UNDERSTAND HOW TO USE THE SOFTWARE TO GET THE INFORMATION THE ASSIGNMENT ASKED FOR.

1.strongly agree    2.agree    3. neither agree, nor disagree    4. disagree    5.disagree strongly

4. THE GRAPHICS OF THE PROGRAM (ie desktop interface and stack design) WERE;

1. Too complex.    2. Too confusing    3. Understandable    4. didn't notice

5. I GAINED SOME KNOWLEDGE ABOUT WORKING ON A COMPUTER WITH A LAZER DISK BY DOING THIS ASSIGNMENT.

1.strongly agree    2.agree    3. neither agree, nor disagree    4. disagree    5.disagree strongly

6. THE MENUS AND BUTTONS WERE APPROPRIATE AND I UNDERSTOOD WHAT THEY MEANT.

1.strongly agree    2.agree    3. neither agree, nor disagree    4. disagree    5.disagree strongly

7.I HAD A HARD TIME MOVING THROUGH THE PROGRAM.

1.strongly agree    2.agree    3. neither agree,    4. disagree    5.disagree strongly  
nor disagree

8.THIS PROGRAM AIDED MY OVERALL UNDERSTANDING OF GRAPHIC DESIGNERS.

1.strongly agree    2.agree    3. neither agree,    4. disagree    5.disagree strongly  
nor disagree

9. THIS PROGRAM KEPT MY ATTENTION BETTER THAN IF I WAS LEARNING THIS INFORMATION THROUGH A TEXTBOOK.

1.strongly agree    2.agree    3. neither agree,    4. disagree    5.disagree strongly  
nor disagree

10. I WOULD USE THIS PROGRAM AGAIN TO DO RESEARCH, JUST FOR FUN, OR TO LEARN MORE ABOUT THE GRAPHIC DESIGNERS.

(CIRCLE THE APPROPRIATE ONE (S))

1.Research

2. Entertainment

3.Information

PLEASE FEEL FREE TO COMMENT ON THE PROGRAM BELOW. YOUR SUGGESTIONS ARE APPRECIATED.

COMMENTS



## SUSAN PRESTON MAUKS

46 WARRINGTON DR.  
ROCHESTER, NEW YORK  
14618  
(716) 442-8140

### EDUCATION

GRADUATE WORK: ROCHESTER INSTITUTE OF TECHNOLOGY: SCHOOL  
PSYCHOLOGY  
M.S. EDUCATIONAL PSYCHOLOGY; CASE WESTERN RESERVE UNIVERSITY,  
CLEVELAND, OHIO  
B.S. PHYSICAL EDUCATION, HEALTH, RECREATION BOWLING GREEN STATE  
UNIVERSITY

### WORK EXPERIENCE

**CORNELL UNIVERSITY** 1984-Present  
ASSITANT PROFESSOR

**MONROE COMMUNITY COLLEGE** 1984-Present  
ASSITANT PROFESSOR

**KEUKA COLLEGE** 1983-1984  
ASSISTANT PROFESSOR  
ASSISTANT DIRECTOR OF WEED PHYSICAL ARTS CENTER

**KENT STATE UNIVERSITY:** 1979-1981  
DIRECTOR OF AQUATICS AND ADAPTED RECREATION

### PUBLICATIONS

**Field Hockey is For Me:** 1982: Lerner Publications, Minneapolis Minnesota  
**Synchronized Swimming is For Me** 1981:Lerner Publications,  
Minneapolis Minnesota.



***Appendix H: Apple Computer, Inc. Human Interface Guidelines***

81  
Lax  
M.T.  
1

The Apple Desktop Interface is the result of a great deal of concern with the human part of human-computer interaction. It has been designed explicitly to enhance the effectiveness of people. This approach has frequently been labeled *user friendly*, though *user centered* is probably more appropriate. It has been thought of as the ideal interface for beginners, though it would be more useful to think of it as good for people in general. It has been labeled *simple*, though *direct* and *effective* make more sense. And it has been described as *easy to learn*, though *accessible* would be as true.

## A view of the user

Not very long ago, most users of personal computers were also programmers. In fact, many of them were computer builders as well, because personal computers were available only as kits. Today, most personal computers are seen as tools that magnify a person's ability to perform all kinds of tasks that were formerly done without computers. The Apple Desktop Interface provides a consistent and familiar computer environment in which people can perform their many tasks. People aren't trying to *use computers*—they're trying to get their jobs done.

Given this focus on people and their tasks, the Apple Desktop Interface has had to assume a model of people, in order to suit the interface to them. People, however, are delightfully complex and varied, which assures that a theory of human activity that would provide a complete framework for the design of human-computer interaction is a long way off. Such a theory would be oversimplified anyway, because computers themselves change the way we think, feel, and behave. Computer design and human activity must therefore evolve together. Apple believes that caring how people behave will help computer designers provide a consistent world that a person can enter with ease and effectiveness, even though many of the details of human activity are not understood.

The Apple Desktop Interface is based on the assumption that people are instinctively curious: they want to learn, and they learn best by active self-directed exploration of their environment. People strive to master their environment: they like to have a sense of control over what they are doing, to see and understand the results of their own actions. People are also skilled at manipulating symbolic representations: they love to communicate in verbal, visual, and gestural languages. Finally, people are both imaginative and artistic when they are provided with a comfortable context; they are most productive and effective when the environment in which they work and play is enjoyable and challenging.

Page 1  
Chapter 1: Philosophy  
A-1

The next section of this chapter sets forth ten human interface principles that explicitly emphasize the views expressed above. A subsequent section describes a programming strategy that takes these principles into account. Chapter 2 describes a specific set of elements for a Desktop Interface that can be used consistently across a range of very different applications. Chapter 3 provides standards and specifications for their implementation.

## General design principles

This section describes the ten fundamental principles of the Apple Desktop Interface.

### Metaphors from the real world

**Use concrete metaphors and make them plain, so that users have a set of expectations to apply to computer environments.**

**Whenever appropriate, use audio and visual effects that support the metaphor.**

Most people now using computers don't have years of experience with several different computer systems. What they do have is years of direct experience with their immediate world. To take advantage of this prior experience, computer designers frequently use metaphors for computer processes that correspond to the everyday world that people are comfortable with.

The desktop itself is the primary metaphor for the Apple Desktop Interface. It *appears* to be a surface on which users can keep tools and documents. Yet many of the elements of the Apple Desktop Interface don't have a clear physical counterpart. For example, scroll bars clearly belong to the computer domain; they only loosely resemble real scrolls. And pull-down menus aren't much like real restaurant menus, except in providing the opportunity to make choices from alternatives.

The desktop, then, is an inviting metaphor that provides easy access to the system. Other metaphors, especially when consistent with the desktop, can fit into the system. Once immersed in the desktop metaphor, users can adapt readily to loose connections with physical situations—the metaphor need not be taken to its logical extremes.

## Direct manipulation

**Users want to feel that they are in charge of the computer's activities.**

People expect their physical actions to have physical results, and they want their tools to provide feedback. For example, when character keys are pressed, users like to hear a click as they see the corresponding characters appear on the screen. When a drawing tool is moved, a line appears. This is true whether or not a computer is being used. Moving a document from one folder or disk to another, or into the trash, can seem to be a physical activity with physical feedback in the computer world as it is in the paper world. The physical activity of moving the mouse reinforces the sense of an action with a real result.

Users want topics of interest to be highlighted. They want to see what functions are available at any given moment. If grave consequences might follow from any of those functions, they want to know about them—before any damage is done. They want clues that tell them that a particular command is being carried out, or, if it cannot be carried out, they want to know why not and what they can do instead.

People appreciate visual effects, such as animation, that show that a requested action is being carried out. This is why, when a window is closed, it appears to shrink into a folder or icon. Visual effects can also add entertainment and excitement to programs that might otherwise seem dull. Why shouldn't using a computer be fun?

## See-and-point (instead of remember-and-type)

**Users select actions from alternatives presented on the screen.**

**The general form of user actions is noun-then-verb, or "Hey, you—do this."**

**Users rely on recognition, not recall; they shouldn't have to remember anything the computer already knows.**

**Most programmers have no trouble working with a command-line interface that requires memorization and Boolean logic. The average user is not a programmer.**

The Apple Desktop Interface is visually and spatially oriented. The way everything—text, applications, documents, lines, controls—appears on the screen is consistent and well thought out. The screen provides an environment in which people can work effectively, taking full advantage of the power of the computer while enjoying a sensible human environment.

Users interact directly with the screen, choosing objects and activities they are interested in by pointing at them. The mouse is currently the most common pointing device, but other effective pointing devices are available.

There are two fundamental paradigms for how the Apple Desktop Interface works. They share two basic assumptions: that users can see, on the screen, what they're doing; and that they can point at what they see. In one paradigm, users first select an object of interest (the noun) and then select an action (the verb) to be performed on the object. All actions available for the selected object are listed in the menus, so that users who are unsure of what to do next can quickly jog their memory by scanning through them. Users can choose, at any time, any available action, without having to remember any particular command or name. This paradigm requires only recognition, rather than recall, of the desired activities.

In the second paradigm, the user drags an object (the noun) onto some other object which has an action (the verb) associated with it. In the Finder, for example, the user can drag icons into the trash can, into folders, or into disks. No action is chosen from the menus, but it's obvious what happens to the object that is sent to another object. For example, an object sent to the trash can is discarded, and the document sent to a disk icon is copied to that disk. In this variant of the Desktop Interface, users do have to remember what an object such as the trash can is for, so it is especially important that objects look like what they do. If the trash can didn't look like the place to discard something, or we didn't know from daily experience that folders contain documents, such an interface wouldn't work. However, when this type of interface is well thought out, it can be easier to learn than menu commands.

Command-line interfaces, on the other hand, require the user to remember a command and type it into the computer. This kind of interface makes considerable demands on the user's memory—especially when the commands are complex or cryptic. Such an interface is especially galling to the new or infrequent user, but it distracts all users from their task and focuses attention instead on the computer's needs.

There are, however, some advantages to the *remember-and-type* approach. Sometimes, when the user is completely certain of what action is desired, a simple keystroke command may be the fastest way to achieve it. For this reason, some desktop applications include *keyboard equivalents* for some menu activities. Keyboard equivalents are a logical extension of the Apple Desktop Interface, fine-tuning it for particular situations. It is essential, however, that keyboard equivalents offer an *alternative* to the *see-and-point* approach—not a substitute for it. Users who are new to a particular application, or who are looking for potential actions in a confused moment, must always have the option of finding a desired object or action on the screen.

## Consistency

**Effective applications are both consistent within themselves and consistent with one another.**

Having learned, in one application, a general set of skills, the user can transfer those skills to other applications. By using the standard elements of the Apple Desktop Interface, you ensure consistency within your application and you benefit from consistency across applications.

Within an application, there should always be one coherent way for the user to implement actions. Though some shortcuts may be provided, users should always be able to rely on familiar and straightforward ways to get things done.

The standard elements of the Apple Desktop Interface ensure consistency, ease of learning, and familiarity across applications. This benefits the typical user, who usually divides working time among several applications, and it benefits every software developer because the user learning how to use a new application builds on prior experiences with the same elements in other applications. This sometimes means that a programmer's new solution that precisely matches a particular situation should be set aside in favor of a slightly less effective but more commonly used solution. In most cases, consistency should be valued above idiosyncratic cleverness.

## WYSIWYG (what you see is what you get)

**There should be no secrets from the user, no abstract commands that only promise future results.**

**There should be no significant difference between what the user sees on the screen and what eventually gets printed.**

A very important use of computers is the processing and printing of text and graphics. In some systems, the computer is an intermediary: the user manipulates a range of computer commands to indicate what is desired, and the computer passes these commands along to a printer. This kind of system keeps the user unnecessarily distant from the final document. The user should be in charge of both the content and the formatting (spatial layout as well as font choices) of the document. The computer should quickly and directly display the result of the user's choices, so the user doesn't have to wait for a printout or make mental calculations of how the screen version will be translated onto paper.

The principle behind this approach is known as *what you see is what you get*. (abbreviated WYSIWYG, pronounced *witzzy-wig*). This approach is highly consistent with the direct manipulation aspect of the Apple Desktop Interface. For example, when a user uses the Finder to copy a document from one disk to another, the user "sees" a copy of the document move to the new disk and can trust that the document is now found on both disks. WYSIWYG is also in the spirit of using a computer as a thinking tool *and* as a production tool.

## User control

**The user, not the computer, initiates and controls all actions.**

People learn best when they're actively engaged. Too often, however, the computer acts and the user merely reacts within a limited set of options. In other instances, the computer "takes care" of the user, offering only those alternatives that are judged "good" for the user or that "protect" the user from detailed deliberations.

On the surface, the concept of computer as protector may seem quite appealing, but this approach puts the computer, rather than the user, in the driving role—something quite at odds with the basic philosophy of the Apple Desktop Interface.

In the Apple Desktop Interface, if the user attempts something risky, the computer provides a warning, but allows the action to proceed if the user confirms that this is what he wants. This approach "protects" the beginner but allows the user to remain in control.

## Feedback and dialog

**Keep the user informed.**

**Provide immediate feedback.**

**User activities should be simple at any moment, though they may be complex taken together.**

To be in charge, the user must be informed. When, for example, the user initiates an operation, immediate feedback confirms that the operation is being carried out, and (eventually) that it's finished. When the application isn't responding to user input because it's processing a different task, the user must be informed of when to expect delays, why, and for how long.

The user must also be kept informed of the progress of an operation: for example, the reason an operation can't be completed at a certain time as well as the fact that it can't.

This communication should be brief, direct, and expressed in the user's vocabulary rather than the programmer's.

## Forgiveness

Users make mistakes; forgive them.

The user's actions are generally reversible—let users know about any that aren't.

Even though users like to have full documentation with their software, they don't like to read manuals (do you?). They would rather figure out how something works in the same way they learned to do things when they were children: by exploration, with lots of action and lots of feedback.

As a result, users sometimes make mistakes or explore a bit further than they really wanted to. Make your application tolerant and forgiving. Forgiveness means letting users do anything reasonable, letting them know they won't break anything, always warning them when they're entering risky territory, then allowing them either to back away gracefully or to plunge ahead, knowing exactly what the consequences are. Even actions that aren't particularly risky should be reversible. Tell the users about any exceptions to this rule.

When options are presented clearly and feedback is appropriate and timely, learning is relatively error-free. Alert messages should therefore be infrequent. If the user is subjected to a barrage of alert messages, something is wrong with the program design.

## Perceived stability

Users feel comfortable in a computer environment that remains understandable and familiar rather than changing randomly.

People use computers because computers are versatile and fast. Computers can calculate, revise, display, and record many kinds of information far faster than people can. If users are to cope with the complexity that the computer handles so easily, they need some stable reference points.

To provide a visual sense of stability, the Apple Desktop Interface provides a two-dimensional space on which objects are placed. It also defines a number of consistent graphic elements (menu bar, window border, and so on) to maintain the illusion of stability.

To provide a conceptual sense of stability, the interface provides a clear finite set of objects and a clear finite set of actions to perform on those objects. Even when particular actions are unavailable, they are not eliminated from a display, but are merely dimmed.

It is the *illusion* of stability that is important, not stability in any strict physical sense. The environment can, and should, change as users interact with it, but users should feel that they have a number of familiar "landmarks" to count on.

## Aesthetic integrity

Visually confusing or unattractive displays detract from the effectiveness of human-computer interactions.

Different "things" look different on the screen.

Users should be able to control the superficial appearance of their computer workplaces—to display their own style and individuality.

Messes are acceptable only if the user makes them—applications aren't allowed this freedom.

In traditional applications, the visual appearance of the screen has been a low priority and consequently somewhat arbitrary. In contrast, the Apple Desktop Interface depends on the visual appearance of the screen. People deserve and appreciate attractive surroundings. Consistent visual communication is very powerful in delivering complex messages and opportunities simply, subtly, and directly.

Users should have some control over the look of their workspaces. This allows individual expression and relieves the computer designer of having to devise one "look" that appeals to everyone.

The next section summarizes some basic principles of visual design.

## Principles of graphic communication

Good design must communicate, not just dazzle. It must inform, not just impress.

The services of a skilled graphic designer are worth the expense.

The real point of graphic design, which comprises both pictures and text, is clear communication. In the Apple Desktop Interface, everything the user sees and manipulates on the screen is graphic. As much as possible, all commands, features, and parameters of an application, and all the user's data, appear as graphic objects on the screen.

Graphics are not merely cosmetic. When they are clear and consistent, they contribute greatly to ease of learning, communication, and understanding. The success of graphic design is measured in terms of the user's satisfaction and success in understanding the interface.

If you design your icons and other graphics on the target screen, rather than on paper, you'll take advantage of whatever that screen has to offer and you'll have the best design possible. Not all screens are alike. For example, a Macintosh Plus has approximately square pixels that are either black or white. Apple II pixels aren't square, and can be any of many different colors.

## Visual consistency

The purpose of visual consistency is to construct a *believable environment* for users. Because such concepts as storing documents in folders and throwing things away in the trash can be the same both in the real world and in the Apple Desktop environment, users don't have to relearn them to begin working. This transfer of skills is one of the most important benefits of a consistent interface, especially for beginning users.

Photographic realism isn't essential; the important thing is that the user understands the intended meaning. A well-designed symbol or caricature can convey meaning better than a completely realistic picture.

If images don't efficiently convey meaning, the user is lost in an environment of random objects, and communication breaks down. Graphics—the icons, windows, dialog boxes, and so on—are the basis of effective human-computer dialog and must be designed with that in mind.

## Simplicity

Simple design is good design. Don't clutter the screen with too many windows, overload the user with complex icons, or put dozens of buttons in a dialog box. Because icons and dialog boxes must fit in a small space, the messages they convey must be simple and straightforward. Simple designs are easy to learn and to use, and they give the interface a consistent look.

The icons, menus, windows, and other graphic elements on the screen make up a basic language with which the user and computer communicate. The user selects an icon and chooses an action from a menu, effectively telling the computer to "Open MacPaint," for example. For this language to work well, the messages must be simple.

## Clarity

Good graphic design begins with an understanding of the situation the user is in or of the problem being solved. A picture isn't always the answer—sometimes words do the job better. Make graphics clear and readable. Try them out on real users, not just on your fellow artists or programmers. The most important part of the graphic should be recognized first, then the second most important part, and so on. Use visual cues such as arrows, movement, and the arrangement of elements to direct the eye to the correct place. The symbols used in different kinds of alerts tell the user if the alert is a note, caution, or warning.

Animation, *when used sparingly*, is one of the best ways to draw the user's attention to a particular place on the screen. For example, users soon learn that the quickest way to find a pointer on a busy screen is to move the mouse, making the pointer move on the screen. Animated pointers reassure the user, during a lengthy process such as saving a large document to disk, that the system is alive and well.

## A strategy for programming

The Apple Desktop Interface relies on some distinctive models for programming, some of which are unfamiliar even to experienced programmers.

To help the programmer make use of this interface, and to carry through in these models, some Apple hardware systems provide an abundance of tools in ROM. The developer derives two major advantages from using ROM-based tools and resources: compatibility and efficiency. The more a program bypasses or replaces these tools and resources, the more likely that sooner or later it will be incompatible with new products or features.

Although a developer might know a more direct way of getting information or performing an operation, using system-provided features ensures hardware independence. For example, always reference the proper data structures to determine the current size of a screen rather than using the constant values for current hardware.

The next sections deal with some important programming issues that are at the heart of the Apple Desktop Interface.



If you design your icons and other graphics on the target screen, rather than on paper, you'll take advantage of whatever that screen has to offer and you'll have the best design possible. Not all screens are alike. For example, a Macintosh Plus has approximately square pixels that are either black or white. Apple II pixels aren't square, and can be any of many different colors.

## Visual consistency

The purpose of visual consistency is to construct a *believable environment* for users. Because such concepts as storing documents in folders and throwing things away in the trash can be the same both in the real world and in the Apple Desktop environment, users don't have to relearn them to begin working. This transfer of skills is one of the most important benefits of a consistent interface, especially for beginning users.

Photographic realism isn't essential; the important thing is that the user understands the intended meaning. A well-designed symbol or caricature can convey meaning better than a completely realistic picture.

If images don't efficiently convey meaning, the user is lost in an environment of random objects, and communication breaks down. Graphics—the icons, windows, dialog boxes, and so on—are the basis of effective human-computer dialog and must be designed with that in mind.

## Simplicity

Simple design is good design. Don't clutter the screen with too many windows, overload the user with complex icons, or put dozens of buttons in a dialog box. Because icons and dialog boxes must fit in a small space, the messages they convey must be simple and straightforward. Simple designs are easy to learn and to use, and they give the interface a consistent look.

The icons, menus, windows, and other graphic elements on the screen make up a basic language with which the user and computer communicate. The user selects an icon and chooses an action from a menu, effectively telling the computer to "Open MacPaint," for example. For this language to work well, the messages must be simple.

## Clarity

Good graphic design begins with an understanding of the situation the user is in or of the problem being solved. A picture isn't always the answer—sometimes words do the job better. Make graphics clear and readable. Try them out on real users, not just on your fellow artists or programmers. The most important part of the graphic should be recognized first, then the second most important part, and so on. Use visual cues such as arrows, movement, and the arrangement of elements to direct the eye to the correct place. The symbols used in different kinds of alerts tell the user if the alert is a note, caution, or warning.

Animation, *when used sparingly*, is one of the best ways to draw the user's attention to a particular place on the screen. For example, users soon learn that the quickest way to find a pointer on a busy screen is to move the mouse, making the pointer move on the screen. Animated pointers reassure the user, during a lengthy process such as saving a large document to disk, that the system is alive and well.

## A strategy for programming

The Apple Desktop Interface relies on some distinctive models for programming, some of which are unfamiliar even to experienced programmers.

To help the programmer make use of this interface, and to carry through in these models, some Apple hardware systems provide an abundance of tools in ROM. The developer derives two major advantages from using ROM-based tools and resources: compatibility and efficiency. The more a program bypasses or replaces these tools and resources, the more likely that sooner or later it will be incompatible with new products or features.

Although a developer might know a more direct way of getting information or performing an operation, using system-provided features ensures hardware independence. For example, always reference the proper data structures to determine the current size of a screen rather than using the constant values for current hardware.

The next sections deal with some important programming issues that are at the heart of the Apple Desktop Interface.

## Modelessness

**With few exceptions, a given action on the user's part should always have the same result, irrespective of past activities.**

Modes are contexts in which a user action is interpreted differently than the same action would be interpreted in another context. In other words, the same action, when completed in two different modes, results in two different reactions. A mode typically restricts the operations that the user can perform while the mode is in effect.

Because people don't usually operate modally in real life, dealing with modes in computer environments gives the impression that computers are unnatural and unfriendly.

A mode is especially confusing when the user enters it unintentionally. When this happens, familiar objects and commands may take on unexpected meanings and the user's habitual actions cause unexpected results.

Most conventional software uses modes heavily. It's tempting to use modes because they sometimes make programming easier. But if you yield to the temptation too frequently, users will consider using your application a chore rather than a satisfying experience.

This is not to say that you should never use modes in applications. Sometimes a mode is the best way out of a particular problem. Most of these acceptable modes fall into one of the following categories:

- Long-term modes, such as doing word processing as opposed to graphics editing. In this sense, each application is a mode.
  - Short-term "spring-loaded" modes, in which the user must constantly do something to maintain the mode. Examples would be holding down the mouse button to scroll text or holding down the Shift key to extend a text selection.
  - Alert modes, in which the user must rectify an unusual situation before proceeding. Keep these modes to a minimum.
- Other modes are acceptable if they do one of the following:
- They emulate a familiar real-life situation that is itself modal. For example, choosing different tools in a graphics application resembles the real-life choice of physical drawing tools. MacPaint and other palette-based applications exemplify this use of modes.
  - They change only the attributes of something, but not its behavior. The boldface and underline modes of text entry are examples.
  - They block most other normal operations of the system to emphasize the modality, as in error conditions incurable through software (for example, a dialog box that disables all menu items except Close).

If an application uses modes, there must be a clear visual indication of the current mode, and the indicator should be near the object most affected by the mode. A good example is the changing pointer in MacPaint: it looks like a pencil, paintbrush, spray can, or eraser, depending on the function ("mode") the user has chosen. It should also be very easy to get into or out of the mode (such as by clicking on a different palette symbol).

No mode should ever prevent a user from saving a document or quitting the application.

## The event loop

**Applications are prepared for the user to do anything at any time.**

The event loop is central to programming for the Apple Desktop Interface. The event loop is the central routine of any application. An application doesn't have to expect a certain set of events in a particular order, but constantly looks for inputs (mouse actions, keystrokes, disk insertions) that can occur in any order and to which it must respond in specific ways.

This approach to programming contrasts with programs that systematically limit the alternatives available to the user, assuring that the user follows the "right" path to the "right" place. Instead, the emphasis is on responding to each local request the user makes, leaving the responsibility for the final destination with the user. In each context, the widest possible range of user activities should be allowed. For example, there's no reason not to let the user set printing options before there's anything to print.

## Reversible actions

**Always provide a way out.**

Because the Apple Desktop Interface encourages users to be active, they often request something they don't really want. To encourage such deliberate (though often unplanned) activities and to give users a sense of control over these activities, programmers should make actions reversible whenever possible. Users should, for example, be able to cancel activities easily, particularly those that are unexpectedly involved. They should also have a range of deliberate choices to confirm that they do want to do something particularly drastic, complex, or time-consuming.

## The screen

### The screen is the stage for human-computer interactions.

In many computer systems, most of the activity is invisible. Users make inputs, to which the computer returns elaborate responses after some amount of calculation. The screen then becomes the "mail slot" through which exchanges between the user and the computer system take place.

In the Apple Desktop Interface, the screen displays a representation of the "world" that the computer creates for the user. On this screen is played out the full range of human-computer interactions. Initially, it provides the alternatives; then it reflects the results of requested activities; then it again shows the alternatives; and so on. And it does this in an extremely well-defined way. The details are in Chapters 2 and 3.

Though the screen is itself not the interface—the functionality provided by the interface elements is the interface—the screen does play a central role, and managing it is one of the programmer's most important tasks.

## Plain language

### Communicate with the user in concise and simple terms.

The Apple Desktop Interface is approachable by the unsophisticated user. It requires no special "language." In fact, much of the user-computer interaction is graphic. The user points to objects on the screen and selects from available lists; the computer changes text and graphics at the user's request.

Occasionally, the computer must display textual messages, either to describe a particular situation or to ask the user for a specific decision. In these instances, the phrasing must be very direct and unambiguous. It should inform users directly of the options available.

Whenever words are involved, the design team should include a skilled writer.

## User testing

### The primary test of the user interface is its success with users.

Can users understand what to do and can they accomplish the task at hand easily and efficiently? The best way to answer these questions is to put them to the users.

## The design process

Users should be involved early in the design process so that changes in the basic concept of the product can still be made, if necessary. While there's a natural tendency to wait for a good working prototype before showing the product to anyone, this is too late for the user to have a significant impact on design. In the absence of working code, you can show test subjects alternate designs on paper or storyboards. There are many ways that early concepts can be tested on potential users of a product. Then, as the design progresses, the testing can become more refined and can focus on screen designs and specific features of the interface.

## Test subjects

There is no such thing as a "typical user." You should, however, be able to identify some people who are familiar with the *task* your application supports but are unfamiliar with the *specific technology* you are using. These "naive experts" make good subjects because they don't have to be taught what the application is for, they are probably already motivated to use it, and they know what they need to accomplish the task.

You don't need to test a lot of people. The best procedure for formative testing (testing during the design process) is to collect data from a few subjects, analyze the results, and apply them as appropriate. Then, identify new questions that arise and questions that still need answers, and begin all over again—it is an iterative process.

## Procedures

Planning and carrying out a true experimental test takes time and expert training. But many of the questions you may have about your design do not require such a rigid approach. Furthermore, the computer and application already provide a controlled setting from which objective data can be gathered quite reliably. The major requirements are

- to make *objective* observations
- to record the data *during the user-product interaction*

Objective observations include measurements of time, frequencies, error rates, and so forth. The simple and direct recording of what someone does and says while working is also an objective observation, however, and is often very useful to designers. Test subjects can be encouraged to talk as they work, describing what they are doing or trying to do, what they expect to happen, and so on. This record of a person's "thinking aloud" is called a *protocol* by researchers in the fields of cognition and problem-solving, and is a major source of their data.

The process of testing described here involves the application designer and the test subjects in a regular cycle of feedback and revision. Although the test procedures themselves may be informal, user testing of the concepts and features of the interface should be a regular, integral part of the design process.

---

---

## Designing for disabled people

Computers hold tremendous promise for people with many kinds of disabilities. In terms of increasing productivity and mobility, computers can have a far greater impact on disabled people than on other users. But too often, computers become obstacles rather than enablers, because many disabilities make it hard to use standard computers and software. In most cases, thoughtful hardware design is the solution, but there are things that software designers can do, too.

Many of the modifications that make programs easier for disabled people to use are simple and inexpensive to make, and they often have a welcome and unexpected side effect—the programs are easier for *everyone* to use. Although sidewalk curb cuts are designed to help people who rely on crutches or wheelchairs, they are used and appreciated almost as much by skateboarders and stroller-pushers.

This section describes some of the ways you can design with disabled users in mind. For more information, contact Apple's Office of Special Education Programs.

---

## Vision disabilities

People with vision problems have the most trouble with the output display. The ability of the Macintosh to handle different sizes of text makes it easy to accommodate the needs of many people with vision problems. Software can be designed with a "zoom" feature that automatically increases the size of characters on the screen.

Color is a problem for many people. Don't let people's ability to use your software depend on their ability to distinguish one color from another. Be sure that all information conveyed by color coding is also presented in some other way (by text, position, or highlighting).

Many people have difficulty using the instruction manuals that usually accompany software products, either because they have difficulty reading small print or because they physically can't handle books. These people appreciate having at least the most important part of the manual's text available in electronic form, so that they can display or print it in oversize characters, print it with a Braille printer, or have it read to them through a speech synthesizer. All users benefit from manuals in electronic form, which can quickly be searched for specific topics and keywords.

---

## Hearing disabilities

Hearing problems are generally no obstacle to using computers, except when important cues are given only with sound. Aside from the obvious exceptions of music or voice-synthesis applications, software should never rely solely on sound to provide important information. Supplement all audible messages with visual cues, or allow the user to choose visible instead of audible messages.

---

## Other disabilities

People with cognitive or verbal impairments are greatly helped by clear and simple language, icons with obvious meanings, and carefully designed displays. Don't make the user's success depend on his or her ability to *remember* many different things.

Another way to make computers easier for both disabled people and others is to provide macros, making it possible to combine a number of keystrokes and mouse movements into *one* keystroke. The way macros are created and accessed must be clear and simple. It shouldn't be easy for a user to invoke a macro accidentally.



***Appendix I: User Profiles, Pioneers in American Graphic Design,  
Catalogue Outline, Walker Show: Graphic Design in America***

Graphic Design Archive on Videodisc

The Users	The Use	How Used
Historians (Art & Design) Researchers Writers Editors	LT research Orientation Comparison Archival <i>Search the Postcard Archive</i>	Network Access Database Disc
Teachers	Course Preparation Models & Examples Exams Workshops/orientation ST Research LT Research	Write own program Network Access Database Disc
Professional Designers	Project Visual Audit ST Research Client Orientation Staff Orientation	Network Access Database Disc
Students	ST Research LT Research Tests/exams Orientation to field	Network Access Database
Non-Designers	Browse	Disc

# Pioneers in American Graphic Design

1890 1900 1910 1920 1930 1940 1950 1960 1970 1980 1990

KAUFFER, E. McKnight 1890-1954	_____											
AGHA, Mehemed Femy 1896-1978		_____										
SUTNAR, Ladislav 1897-1971		_____										
BINDER, Joseph 1898-1972		_____										
BRODOVITCH, Alexey 1898-1971		_____										
COINER, Charles 1898-		_____										
BAYER, Herbert 1900-1985			_____									
CASSANDRE, A.M. 1901-1968			_____									
BEALL, Lester 1903-1969			_____									
KEPES, Gyorgy 1906-				_____								
EAMES, Charles 1907-1978				_____								
EAMES, Ray 1907-				_____								
MATIER, Herbert 1907-1984				_____								
BURTIN, Will 1908-1972				_____								
GUISTI, George 1908-				_____								
NITSCHKE, Erik 1908-				_____								
HURLBURT, Allen 1910-1983				_____								
LIONNI, Leo 1910-				_____								
GOLDEN, William 1911-1958				_____								
THOMPSON, Bradbury 1911-				_____								
TSCHICHOLD, Jan 1912-1974				_____								
RAND, Paul 1914-					_____							
LUSTIG, Alvin 1915-1955					_____							
BASS, Saul 1920-						_____						
DANZIGER, Louis 1923-							_____					



**Catalogue Outline:**  
**Graphic Design in America: A Visual Language History**  
**For the Exhibition to Open 4 November 1989**

Preface

Caroline Hightower

The role of the graphic design profession in the evolution of American design, from the vantage of the Director of the American Institute of Graphic Arts

Introduction

Mildred Friedman

The role of graphic design in American art and life and the need for a critical evaluation of the history of this discipline are the focus of this book and of this introduction. The essays that follow are outlined and the structure of the book and the exhibition it accompanies are discussed by the exhibition's curator.

Time Line

Ellen Lupton, J. Abbott Miller

A visual and verbal depiction of the history of American graphic design from 1829-1989 analyzes the cultural and technological developments of the discipline and relates developments in design with those in American history. The time line uses the terms of American presidents as a base grid.

Technologies and Design

Estelle Jussim

In the past 100 years we have seen the development of photography, film, television, and electronic technologies which have radically affected the process and product of graphic design. As the moving image has replaced print as the dominant means of communication, the role of graphic design in this evolution has constantly changed.

## Design for the Public Good

Neil Harris

One area in the public realm that has had little attention in design discourse is that of graphic art for the public good -- or design as a social force. Professor Harris concentrates his analysis of this genre in the so-called Progressive Era, between the turn of the century and World War I, when the communication of information about social issues such as health and welfare was the concern of both private and public agencies.

## Communicating for Commerce

Maude Lavin

Advertising design and its role in the growth of today's consumerist society will be discussed in this essay. The corporate client's responsibility to the consumer in the most visible realm of the graphic arts is analyzed. Ms. Lavin suggests that designers can themselves produce a kind of advertising that addresses social and cultural issues.

## Europeans in America

Lorraine Wild

World War II's political refugees played a key role in the development of American design after 1940. The New Bauhaus in Chicago, and the activities of such powerful figures as Herbert Bayer, Herbert Matter, László Moholy-Nagy, Ladislav Sutnar, and Alexey Brodovitch are the subjects of this essay.

The influence of European Modernism on the evolution of American design is central to this topic.

## The Political Poster in the 1980s: From the Poster of Protest to the Poster of Liberation

David Kunzle

Communicating ideology through graphic design has a long and vital history. America, in the twentieth century has witnessed a surge of activity in the production of political graphics during the World Wars, the Vietnam War, and the civil rights movements of the 1950s and 60s. The current troubles in Central America have elicited a new series of

political images, as has the AIDS crisis. Professor Kunzle analyzes this material and compares today's posters with previous examples in the realm of American political protest.

#### A Zero Degree of Graphics

Joseph Giovannini

This is a cautionary tale, in which Mr. Giovannini warns against design overwhelming the word in the print media.

He urges a renewed concern for the relationship of word and image, and looks critically at the bent in many current periodicals toward a confusing mixture of editorial and advertising content. He reminds us that the designer has a double obligation: one to the content of the material at hand, and the other to himself as designer.

#### Interviews with Designers

Steve Heller

Speaking directly with the protagonists, Steve Heller provides personal insights into the evolution of recent graphic design. The group of established designers with whom he has discussed critical design issues includes: Paul Rand, April Greiman, Robert and Richard Greenberg, and Ivan Chermayeff. The interviews will be distributed throughout the book.

#### Plates

In addition to the illustrations that accompany the essays, a section of plates will be devoted to materials from the exhibition.

#### Biographies, Bibliography, Index, Lenders, Acknowledgments, Credits

The catalogue will be approximately 248 pages long and will include the essays and over 100 pages of illustrative material.

# GRAPHIC DESIGN IN AMERICA: A VISUAL LANGUAGE HISTORY

A Major Exhibition Organized by Walker Art Center, Minneapolis, in  
Collaboration with the American Institute of Graphic Arts, New York

National Tour Begins November 1989

Graphic Design in America is the first large-scale museum exhibition to explore the evolution of this pervasive art form. This exhibition will bring together the numerous disciplines of graphic design from print through electronics, from the late nineteenth century to the present, and examine these in aesthetic terms and as a barometer of the society they reflect.

Creating communications with words and images that provide information in an artful, imaginative way is the central purpose of graphic design. To accomplish this goal, ideas and events are reacted to and acted upon by designers who translate concepts into cogent visual messages. Because these messages are ubiquitous, they are, as a rule, not seen as significant design achievements. Furthermore, unlike architecture, for example, for which a vast written history exists, graphic design is a very young discipline, practiced as a profession in the U.S. only since the late 1920s. Graphic Design in America will provide a critical evaluation of this growing body of material, and insights into the expressive powers of graphic design will be gained through the exhibition itself, its catalogue, and related programs.

At Walker Art Center the exhibition will occupy 8500 square feet in three galleries and the entry concourse. In addition, a continuous slide-tape program will be shown in the Information Room. Because so much material exists in each design category, the exhibition will be organized into "case studies" which will exemplify various types of graphic communication. All elements of the exhibition's installation are designed to travel.

#### Walker/Guthrie Concourse:

A "moving" poster will involve the linking together of eighteen video monitors via a computerized element. These images in several forms will constitute the visitor's initial immersion into graphic communication, which is the substance of the exhibition. The poster will be reproduced in printed form, and the same image will be modified to create a 14 x 35 foot billboard for the north facade of the Walker building.

The exhibition will be organized by genres of design rather than in chronological arrangement. However, each section will include a time line, using the terms of American presidents as a base grid, and placing the materials included in each section within the appropriate time frame. This system will present graphic design within the social, political, cultural, and commercial contexts of American history and contemporary life, and will be accessible through both verbal and visual means. Four interactive computer games will be located in the exhibition to provide visitors with insights into the evolution of signs and symbols, typography, print technologies, and the moving image.

#### Gallery 1:

##### Environmental Graphics

Environmental graphics provide a direct way to introduce the topic of visual communication, taking the visitor on a metaphorical journey into the city. This section will include graphics related to signs and symbols, trademarks, billboards, posters and broadsides, ephemera, packaging, guides and maps, fairs, expositions, and a variety of public events.

## Gallery 2:

### Mass Media

The mass media are the most familiar carriers of words and images and offer a broad diversity of expressive means to the designer. Included in this gallery will be the print media (newspapers, magazines, books), and the media that employ moving images (film, television, and computers). In addition, typography will be examined in terms of the history and creation of typefaces, and the evolution of reproduction technologies.

## Gallery 3:

### Design for Institutions and the Institution of Design

Institutional design will include that for corporations, government, and cultural organizations, using a number of exemplary cases in which graphic design has been the primary public means of communicating the character and quality of an organization.

A final section will pay homage to a number of outstanding American designers who have made significant contributions to the history of this art form.

### Restaurant:

On the west wall of the Walker restaurant we shall recreate the 8 x 40 foot mural that was originally designed for the cafeteria of the CBS building in New York City. The mural will be photographic in part, and one 4 x 8 foot section will be reconstructed at full scale. The mural uses a diversity of three dimensional letter forms, and actual cooking utensils and foods are incorporated into this remarkable design.

\*\*\*\*\*

14 November 1988

Walker Art Center

Graphic Design in America

The Exhibition

"The function of the designer is to increase the legibility of the world."

--Abraham Moles

A. Introduction and Purpose

Walker Art Center, in collaboration with the American Institute of Graphic Arts, proposes the first large-scale exhibition of American graphic design. The relatively brief history of graphic design began in the 1890s with the development of the means to create virtually unlimited numbers of copies by mechanical reproduction processes, first with print and later with photographic and electronic technologies. As a consequence of this technological revolution, for many people the "real" experience has been replaced by a "vicarious" one, the situation so eloquently expounded by André Malraux in his seminal 1951 work, The Voices of Silence, in which he predicted the now ubiquitous "museum without walls." Graphic designers play a central role in the world of the vicarious image; and their contributions to its burgeoning visual languages will be the focus of this exhibition.

The substance of the message of graphic design may be information, propaganda, art, or commerce. In the exhibition, each of these areas of communication will be examined in its various forms: print, including books, magazines, posters, newspapers; environmental design for signage, billboards, packaging, and

Walker Art Center

Graphic Design in America

p 2

public information systems; and finally, the moving images of film, video and the computer.

The proliferation of communications in our time in all its forms--for good cause, for profit, as information, as art--has created a visual "overload" of unprecedented proportions, thus careful, informed selection of the materials to be included is essential to the success of this effort. The criteria to be used must provide an inclusive view of the discipline's most significant examples. In making these choices we recognize that graphic design is a verbal-visual expression and that to reach an audience a designer must create an image that is accessible on several levels. In Objects of Desire, the English critic Adrian Forty recently wrote of industrial design, "no design works unless it embodies ideas that are held in common by the people for whom the object is intended." Similarly, the visual language employed by the graphic designer must, to be effective, have a recognizable formal vocabulary, as frame of reference is all in communication.

Initially graphic design attempted to imitate painting, as its aesthetic criteria were the ones it knew. Later, design developed its own aesthetic and the imagery and verbal language of graphic design have had a significant impact on the other visual arts. The incorporation of letter forms and popular imagery into 20th-century painting and printmaking is well-known. Coming out



Graphic Design in America

p 3

of Cubism, we know that Americans such as Stuart Davis and Robert Motherwell have used lettering and segments of printed matter in their works. In the 1960s, Pop artists took imagery from billboards and other commercial forms to create their startling insights into the temper of the times.

In recent decades American design has grown in tandem with our increasingly consumerist society, and design in the service of commerce has created both real and imagined needs, overwhelming the much smaller, yet still vital areas of political commentary and art that were at the center of graphic design in earlier times.

It was the influx of European émigrés in the late 1930s that catalyzed the design community in America. The reincarnation of the Bauhaus in Chicago and the move to the U.S. of such innovators as Alexey Brodovitch, Herbert Bayer, Herbert Matter and Ladislav Sutnar brought a completely new sensibility to American design. Where design for the print media had been essentially narrative illustration combined with a classical use of 19th-century book typography, the Europeans brought to it all of the new attitudes about form and pictorial space that had provided the intellectual bases for Cubism, Dada and Surrealism in the post-World War I years. The visionary ideologies of De Stijl, and the Bauhaus were utopian movements in which the striving to break with the past to create a new physical and social environment led to

Walker Art Center

Graphic Design in America

p 4

innovation in all the arts. When that spirit was united with a politically and socially less sophisticated, yet ebullient American sensibility, the marriage of wisdom and energy laid the groundwork for the vital, innovative generation that was to emerge after World War II.

B. Background

A museum exhibition of American graphic design has not previously been undertaken in the depth proposed here. A number of exhibitions with narrower focuses, such as The Museum of Modern Art's 1968 poster exhibition, Word and Image, Yale University's Herbert Matter: A Retrospective of 1978, and the Walker Art Center's 1984 The 20th-Century Poster: Design of the Avant-Garde have included some American works. In addition, American graphic design has been treated as an aspect of a period or movement within exhibitions such as The Smithsonian's Bicentennial Nation of Nations. European graphic design has had far more attention in, for example, the landmark series of Centre Pompidou exhibitions of the late 1970s: Paris-New York, Paris-Berlin, 1900-1933, and Paris-Moscou, 1900-1930.

One reason for the limited amount of museum attention to graphic design in America may be that as yet there is not a large body of scholarly or critical literature in the field. Unlike architecture, in which a vast written history

Walker Art Center

Graphic Design in America

p 5

exists, graphic design is a very young discipline practiced as a profession in the U.S. only since the late 1920s. For that reason this exhibition will attempt to focus on an evaluation of this growing body of material and provide insights into the expressive powers of its visual forms.

In the early 1900s, print design in America was produced by printers, book typographers and commercial artists. As advertising grew to be a lucrative business, it provided essential support for the developing magazine publishing industry. The perfection of offset lithography and four-color printing processes by 1912 enabled illustration to be reproduced in full color, leading to the so-called "golden age" of American illustration.

Graphic art of the 1920s reflected the postwar mood of consumerism and prosperity. A preoccupation with progress and the machine inspired the application of the exuberantly decorative geometry of Art Deco architecture and advanced product design to graphics. By the late 20s, modern graphic design had taken root and was in widespread use in such publications as Vanity Fair and Harper's Bazaar. Then worldwide depression and the oppressive European social and political life of the 1930s brought a wave of European designers to this country. From their point of view, because of its refreshing lack of history and entrenched traditions, America was fertile ground for the new modernism that had already made significant inroads into European design.

Walker Art Center

Graphic Design in America

p 6

Between 1930 and 1955, the first generation of self-conscious graphic designers made possible the transformation from "commercial artist" or "layout man" who executed someone else's concepts, to graphic designer. The European avant-garde brought a new formal language to America. They favored asymmetrical typography as a natural extension of the mechanical typesetting process; they saw abstracted symbols and bold sans-serif typefaces as more appropriate to the accelerated pace of the Machine Age.

With the initiation of the Works Progress Administration in 1935 came the WPA poster project that resulted in some 35,000 designs addressing cultural, and public service issues. Photojournalism became increasingly important at this time and its role in graphic design was heightened by the advent of Life magazine in 1936. World War II political propaganda became a forum for the designer; posters and their billboard cousins were once again active genres.

Modern typography and layout were reinvented in American magazine design by Alexey Brodovitch, art director of Harper's Bazaar from 1934 to 1958. Like the Dada or concrete poets, he made typography interpret and express content. Just as radical as Brodovitch's work was that of Ladislav Sutnar for architecture's Sweet's Catalog Service in the early 1940s. Will Burtin, art director for Fortune magazine from 1945 to 1948, combined an innovative use of illustration with a functionalist attitude toward type.

Walker Art Center

Graphic Design in America

p 7

The belief that modern design was the appropriate expression of the Machine Age was explored by László Moholy-Nagy, who had come to Chicago to establish the New Bauhaus in 1937. Two of Moholy's associates, Gyorgy Kepes and Herbert Bayer, did much in this country to link graphic design with science and technology. By allying graphic design with science and technology they also hoped the prestige of those social forces would help legitimize design. By stretching graphic design beyond the limits of aesthetics, they made it more accessible to an audience of informed clients and helped build the acceptance of modern graphic design as the visual language of American business, thus building a base for the profession itself. Later, Herbert Matter synthesized the typographic ideas of Moholy-Nagy with the imaginative, illustrative freedom of the French Purists with whom he had studied during the 1920s.

After World War II a generation of significant American graphic designers came into its own: Paul Rand, Lester Beall, Alvin Lustig, and Bradbury Thompson who had produced significant work much earlier were then recognized as major innovators. They were proponents of the theory that graphic design was meant to embody ideas instead of simply depicting them, and that form and content were integral to each other.

walker Art Center

Graphic Design in America

p 8

Symbols played an important role in modern graphic design during and after World War II, when European and American designers were concerned with improving worldwide verbal and visual communications. As Rand stated in Thoughts on Design in 1947, "it is in terms of symbolic, concrete forms that the designer ultimately realizes his perceptions and experience, and it is in a world of symbols that the average man lives." The role of the corporation as a user and patron of design became more clearly defined in the 1950s. The corporate logo, an elemental glyph or abstract sign, which functioned as a new character in the world of symbolic forms, took on increased importance. The widely recognized logos for such firms as IBM, International Paper and Chase Manhattan Bank testify to the ubiquitous role of graphic marks in contemporary society. William Golden's iconographic eye guided CBS Television into the era of electronic imagery.

During the turbulent 1960s, a decade of upheaval and social change, graphic design expressed the issues of the day. A new "poster mania" developed as graphic designers, and often non-professionals, created broadsides on issues of war, civil rights, and the environment. Near the end of the decade there was a marked revival of earlier European design termed "the international style of typographic design," which used sans-serif type on a strict geometric grid to achieve clarity and order. While the genesis of this movement occurred in Switzerland, many young American designers, who studied in Basel, brought

Walker Art Center

Graphic Design in America

p 9

this systematic approach back to America. Typographic expression was also fueled by the development of new phototypesetting technologies, freeing designers from the constraints of metal typesetting. Herbert Lubalin, perhaps more than any other designer, defined new typographic possibilities during the 1960s.

In the 1970s newspapers recognized the need to create a new graphic approach to content, and papers such as The New York Times and The Minneapolis Star and Tribune were redesigned. This activity in newspaper design, fueled by the computer revolution, was accompanied by increased interest in information design, those graphics designed to explain, codify, and simplify complex data. In 1974 the Department of Transportation commissioned the American Institute of Graphic Arts to develop a set of thirty-four symbols for public facilities. Following this, the Department of Labor, the Internal Revenue Service and NASA all created visual identification programs and graphic standards manuals.

The introduction of the microcomputer in the 1980s has provided graphic designers with a powerful new tool. Exploring the verbal and visual potential of digitized type has become an important field of activity that will undoubtedly not only change the way we design and produce graphic material, but will have a profound effect upon the teaching of the basic skills of literacy.

Walker Art Center

Graphic Design in America

p 10

C. The Exhibition: A Walk Through the Floor Plan

The exhibition will occupy approximately 9500 square feet in three galleries and the Walker entry Concourse. In addition, film and slide-tape programs will be shown in the Information Room of the Art Center. (Provisional floor plans follow.)

Concourse:

A twenty-five foot high, skylit space, this area will contain a large-scale environment in which video tape and electronics will provide a series of changing images and messages about the exhibition on a three-dimensional structure. Thus, the visitor will be exposed to graphic design in its newest and most pervasive form.

Information Room:

A slide-tape presentation will provide a historical introduction to the exhibition; a film loop, to be shown alternately with the tape in the same space, will bring together a collection of film titles of the past forty years. Historian Philip Meggs will be a consultant for this overview of design history.



Walker Art Center

Graphic Design in America

p 11

Gallery I:

This gallery will introduce the visitor to anonymous 19th- and early 20th-century environmental materials such as billboards and circus posters. The outside walls of a U-shaped structure will have a timeline from 1890-1990 ~~that will be created for the exhibition by J. Abbott Miller, Ellen Lupton and George Sedak of Cooper Union's Herbert Lubalin Study Center.~~ In words and images it will trace the evolution of graphic design, design technology, significant designers, and cultural, social and political events of the past one hundred years.

The interior of this structure will contain small-scale materials from the 19th and early 20th century: books, periodicals, broadsides, posters and numerous examples of print ephemera.

Gallery II:

Print from 1920-1990 in all its forms will be shown in this space: books, posters, magazines, newspapers. These will be grouped according to media; chronological order will be followed within the various sections. For example, the east wall will be devoted to newspapers, from the traditional formats of the early 20s to the computerized grid systems used today. In every

Walker Art Center

Graphic Design in America

p 12

case, distinctive examples will be chosen and only major innovations will be included.

Significant 20th-century figures emerged in this period, and a number of these designers will have areas in which their accomplishments in the print media will be shown. Among the designers who will be singled out in this gallery are: Herbert Bayer, Lester Beall, Alexey Brodovitch, Lou Danziger, E. McKnight Kauffer, Gyorgy Kepes, Herbert Lubalin, Alvin Lustig, Herbert Matter, László Moholy-Nagy, Paul Rand, Ladislav Sutnar, Bradbury Thompson.

Gallery III:

This area will include environmental graphics and the moving image. One of the most effective ways in which designers play a role in our daily lives is through signage systems that guide us on our highways, in our cities, and through our public buildings. Packaging is another way in which graphics, in combination with industrial design, have a powerful presence. Examples of complete programs in these areas will be shown as case studies.

The moving images of film, television and the computer are the newest areas in which graphic design makes significant contributions. Film and television titles and credits, television advertising, and the most recent challenge,

Walker Art Center

Graphic Design in America

p 13

that of computer systems, are fields in which graphic designers are increasingly active.

Designers to be singled out in this gallery include: Robert Abel, Saul Bass, the firm of Ivan Chermayeff and Thomas Geismar, Lou Dorfsman, William Golden and April Greiman. Eric Martin, who teaches at the California Institute of the Arts, and April Greiman will act as consultants to the exhibition in the area of computer graphics.

Creating communications that provide information in an artful, imaginative way is graphic design's central purpose. How well it accomplishes its goals will be the critical focus of this exhibition.

A travel schedule will be developed to several U.S. museums.

Walker Art Center

Graphic Design in America

p 14

Catalogue: Graphic Design in America: A Visual Language History

Introduction

Mildred Friedman

The genesis of the exhibition and catalogue and the need for a definitive examination of this activity that plays such a pervasive role in our daily lives.


Design Timeline

J. Abbott Miller

Ellen Lupton

A visual and verbal depiction of the history of graphic design in America, analyzing cultural and technological developments and the emergence of major innovators in the field.

Design as a Cultural Force

  
~~Alan Freshwater~~

The cultural framework for communications design in its various contexts.

Communicating for Commerce

Maud Lavin

The history and current role of advertising design as a cultural phenomenon.

Politics, Propaganda and Design

Lawrence Weschler

Communicating ideology through graphic design

Walker Art Center

Graphic Design in America

p 15

(Catalogue description continued)

Europeans in America

Ekhard Neumann

The key role of World War II's political refugees to the development of American design after 1940

Graphic Design as an Art Form

Joseph Giovannini

The emergence of graphic design as an art form in its own right and its place in the visual arts

Interviews

Steven Heller

Major figures, active in a diversity of design fields, will be interviewed, and their responses will be dispersed among the book's chapters.

Biographies, Bibliography, Index, Lenders, Acknowledgments, Credits

Walker Art Center

Graphic Design in America

p 16

Related Events

In conjunction with the exhibition, Graphic Design in America, Walker Art Center will organize a Saturday Symposium that will address pertinent issues related to the themes of the exhibition.

Workshops by two design teams will take place in the morning. The afternoon session will begin with an address by a cultural historian who will assess the role of graphic design within the larger social context. Responses to this analysis will be given from three points of view: art, commerce and politics. Finally, a panel of designers will discuss the impact of new technologies on the field.

In addition to the general public, the audience for the Symposium will include local graphic designers, students at the University of Minnesota, Metropolitan State University, the Minneapolis College of Art and Design and Hamline University.

Four fellowships will be established for young graphic designers outside the state of Minnesota, for the purpose of attending the Symposium, assisting with the workshops and participating in related activities to be scheduled by the Education Department. On Sunday, the Education Department will organize a Family Opening that will include workshops led by the young visiting graphic designers.

Walker Art Center

Graphic Design in America

p 17

The proposed schedule is as follows:

Saturday

10 am-12 noon

Two Workshops:

Ivan Chermayeff and Tom Geismar:

Designing the Environment

Richard Saul Wurman and Frank Stanton:

Access Press, and the Publication of  
Information Systems

2-3 pm

"Graphic Design as a Cultural Phenomenon,"

an address by ~~Alan Trachtenberg,~~

*Neil Harris?*  
~~Director of American Studies,~~

~~Yale University~~

3-3:45 pm

Response to ~~Trachtenberg~~ by:

Lawrence Weschler, writer (politics)

Maud Lavin, art historian (commerce)

Joseph Giovannini, architecture critic (art)

Walker Art Center

Graphic Design in America

p 18

4-5:30 pm

"New Technologies:

The Evolution of New Forms in Graphic Design,"

the topic for a panel discussion by:

Steven Heller, art director (moderator)

Robert Abel, designer

April Greiman, designer

Richard Greenberg, designer

Sunday

2-4 pm

Children's workshops organized by the  
Walker Education Department and the visiting  
design fellows.