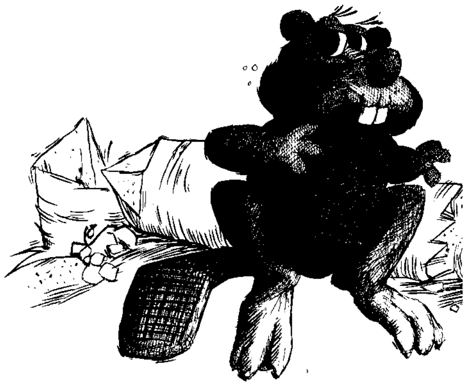

A Survey of Research in Computer-Based Menus

Darrell R. Raymond
Data Structuring Group

CS-86-61

November 1986

A Survey of Research in Computer-Based Menus†

Darrell R. Raymond

Data Structuring Group
University of Waterloo
Waterloo, Ontario, Canada
N2L 3G1

ABSTRACT

This document surveys papers on computer-based menus and menu systems. It consists of a brief overview of research trends, an alphabetical bibliography, and a concise description of each paper.

November 18, 1986

† This research was funded by grant G1154 of the Natural Sciences and Engineering Research Council of Canada.

A Survey of Research in Computer-Based Menus†

Darrell R. Raymond

Data Structuring Group
University of Waterloo
Waterloo, Ontario, Canada
N2L 3G1

Introduction.

This survey is the outcome of some three years' investigation of literature on computer-based menus and menu systems. My original motivation for looking at the literature was to find scientific guidelines for designing menus, hoping this might help me to improve videotex systems. I found that while much research had been done, the knowledge was diffuse and unorganized, and hence progress towards a comprehensive understanding of menus was lacking.

One good reason for this situation is that sometimes researchers are unaware of previous work. Papers relevant to menu design appear in journals and conferences about ergonomics, cognitive psychology, human-computer interface studies, and computer systems. Many of the papers described in this guide were not located through citations but instead by brute-force searching techniques. Perhaps researchers cannot be blamed for repeating or contradicting work they are unaware of; on the other hand, how can progress be expected? As Jim Foley puts it, we step on each others' toes rather than on each others' shoulders. One purpose of this guide is to provide the necessary "shoulders" in the form of a comprehensive list of research efforts.

One bad reason for the lack of progress is that it seems difficult to say something new and worthwhile about menus and menu systems. One of the most personally frustrating experiences I have is to find a new paper about menus that contains only unscientific suggestions (e.g., "choose meaningful option labels"), or is only a re-packaging of existing research. Papers of this type have been deliberately left out of this guide. I have also chosen to exclude papers on systems which use menus but make little contribution to advancing the state of the art in menu design.

The survey consists of three main sections. The first section describes some research areas or trends, placing the papers in these trends. This overview is short enough to be read in its entirety, and will serve to point the reader towards papers of specific interest. The second section contains summaries of each paper which concisely describe the major results or interesting techniques of the research. More information can be found by locating the papers themselves with the third section, an alphabetical bibliography.

Any misinterpretation of the papers should naturally be attributed to me, rather than to the work cited or the people who have contributed to this guide. I would appreciate being informed of omissions or errors; improved, expanded and corrected versions of the survey will be produced as regularly as seems worthwhile.

There are many ways to describe the three basic components of menus; I have chosen to use three terms consistently. A menu provides a set of choices or logical alternatives I refer to as *options*. An option is represented or identified by a *label*, which may be a text string, an audio waveform, or an icon. Options are chosen by entering a *selector*, perhaps an identifier associated with the option, or the position of the label in the menu.

1. Menus.

Menus are frequently studied as independent entities. Issues for menus include choice of labels and selectors, and the organization of options.

1.1. Labels.

The most common type of label is a text phrase. Several studies have confirmed the need for careful selection of textual labels.[55] An important and frequently mentioned technique for improved labelling is the use of subordinate options as components of a label. The use of subordinate options to clarify a textual menu label was first investigated by Latrémouille and Lee, [32,35,36] who showed this was both effective and preferred by subjects. The effectiveness of such labelling was confirmed by Snowberry *et al.*,[59,60] Tolle *et al.*,[69] and by Dumais and Landauer.[13] Subjects in this last experiment performed as well with menus containing examples as they did with either menus with general labels or menus with both general labels and examples. Raymond [52] proposes that subordinate option labels used as descriptors should be themselves selectable, in order to reduce the number of choice which need to be made; such a system is described by Penna.[48] Koved and Shneiderman [30] describe *embedded* menus, in which the labels are an integral part of a larger piece of prose text. Greenberg and Witten[20] investigated six different ways of labelling menus for alphabetically organized information, and obtained the best performance with labels that delimited the upper end of the range of each option.

Iconic menus are possible if the system has sufficient graphical capabilities. Hemenway[24] describes factors governing the selection of icons for command menus. Bewley *et al.* [5] report experiments undertaken to choose an appropriate set of icons for operating system commands. Subjects chose the set of icons that were most distinguishable from each other. On the other hand, Engel *et al.*[14] describe a system in which each object has the same icon — a black square. The icons are clustered in logical groupings, each with a textual label for context, but users must remember the content of an object by its icon's position in the group. Feiner *et al.*[15] describe a system which uses small graphical depictions of the target display as menu labels. Herot[25] describes a system which uses icons to display data maintained in a relational database. The values of selected attributes determine the physical characteristics of a tuple's icon (e.g., its size or colour).

Other types of labels include *aural* labels for telephone-based menus as described by Kidd.[28]

1.2. Selectors.

Options are typically selected by typing a number or letter associated with the option. Goodwin[19] describes a system in which either letters or numbers can be used as selectors; furthermore, the letters associated with a particular path of options in a hierarchy can be entered as a single unit to move directly to the designated position. Perlman[49] found distinctions between using numbers and letters as selectors, especially if the selector was incompatible with the label (i.e., the selector letter was different from the first letter of the label). In Perlman's experiment subjects performed more poorly with incompatible letters than incompatible numbers, but also more accurately with compatible letters than compatible numbers. Pollard and James[51] investigated several types of numeric codes for alphabetic selections.

Options can be selected by entering the position of the label, if a touch screen or other pointing device is employed. Penna [48] and Engel[14] describe videotex systems using a touch screen. Penna's system includes subordinate options both as descriptors and as fully selectable options, while Engel's system attempts to exploit the user's spatial abilities to organize the selectors. Options can also be selected with a light pen, as described by Uber *et al.*[73] or by a mouse; Bewley *et al.*.[5] describe experiments done to optimize mouse design for an iconic menu system.

1.3. Structure of options.

Option structure can reflect the underlying domain of the labels (e.g., alphabetic, chronological), it may exploit the physical characteristics of the display, or employ a combination of both.

Card[8] investigated functional, alphabetic, and random organizations of options in menus with 18 options. He found that alphabetic arrangements were superior but that extensive practice rendered all three organizations roughly equivalent. McDonald *et al.*[40] performed a similar experiment on menus with 64 options and found that categorical organizations were superior. This study also showed significant differences between presenting the target itself and presenting a description or definition of the target. Somberg[62] compared positionally constant option arrangement with alphabetic and probabilistic arrangements and found that positionally constant arrangements produced faster search after moderate practice. Perlman[49] showed that the time taken to select a menu option is a linear function of the size of the menu, as did Allen[2] and Somberg *et* *al.*[63] In a followup study, Somberg and Picardi[64] attempted to isolate the effect of subject familiarity with the target on search time.

Parkinson *et al.*[46] investigated the interaction of spatial organization and the structure of the underlying domain. Columnar menus were searched more quickly than row-organized menus, especially if subgroups were separated by space. Somberg's [62] study of positional constancy also included a spatial component: menus were presented either as a single column or four columns displayed in the corners of the display. No significant effect of spatial condition was observed. Allen[2] observed that more errors were made in selecting from the right column than the left of two-column menus. Raymond *et al.*[53] conducted an experiment in which subjects constructed their own hierarchies. They observed that subjects tended to arrange menus in columnar form and temporal order, rather than alphabetically or semantically. Engel *et al.*[14] structure iconic menus by spatially clustering semantically related groups. Feiner *et al.*[15] discuss several types of menu structure, using space to reflect alphabetical, chronological, and functional relationships. Teitelbaum and Granda[65] varied the location of titles and other ancilliary information in menu frames. Subjects who searched menus with ancilliary information in constant positions had faster response times when queried about the ancilliary information.

Young and Hull[80] observed that the mislabelling of options can confuse subjects not only about the content of the options, but the strategy of the menu. Furthermore, the subject's preconceptions about possible strategies may lead to an incorrect or incomplete reading of the labels.

Tennant *et al.*[67,66] constructed a system in which the structure of options is described by a context-free grammar. A full-screen syntax-directed editor is used to control access to menus which are used to produce English-like queries on a relational database.

Apperley and Spence [4] describe a system with *multiple* menus; in these the options are interconnected so that cancellation of a selected option results in automatic selection of the next option in the sequence.

2. Menu structures.

Certain aspects of menu design require consideration of the structure in which menus are arranged e.g., the number of options to be placed on a given menu. Issues for menu structures include hierarchical design, depth/breadth tradeoff, and non-hierarchical design.

2.1. Hierarchical design.

Menus are frequently arranged in hierarchies. Though not the earliest, the ZOG system [1,38,54,79] is the most widely studied example of text-based menu hierarchies. ZOG menus contain textual labels that are selectable by touch screen. ZOG's developers claim that ZOG is more than an information retrieval system, in fact an interface philosophy. The distinguishing mark of ZOG is the speed with which menus are presented, and hence the corresponding rapidity with which users can access the information. This speed of interaction was also present in an early system reported by Uber *et al.*[73]

Slower and more conventional text-based hierarchies have been extensively studied by Frankhuizen and Vrins[16] as well as Lee *et al.*[36,35] The most prominent conclusion from this body of work is that the probability of error in navigation of a large hierarchical database is quite high; one error or more per search is common.

Whalen and Mason [76] investigated defective hierarchies by introducing three types of defects: miscategorization, ambiguous labels, and synonymous labels. The most serious defect was miscategorization; retrieval of miscategorized data was highly improbable. Whalen and Latrémouille [75] conducted a study in which subjects searched for information after being informed that the presence of the information was not guaranteed. Subjects conducted fairly extensive unsuccessful searches and failed to answer some queries for which information was present. The authors suggest that defects in the database may render some information unlocatable since searches are not exhaustive. Young and Hull[80] observed that subjects will make errors if their perception of the strategy of the structure is different from its actual strategy e.g., if the menu choices overlap but the subject expects a partition. The subject's expectations for the structure may also result in incorrect perceptions of a given menu.

McEwen's[41] study of retrieval in a videotex environment showed that the majority of errors were made at or near the root of the hierarchy, indicating that improved root menus would lead to a significant improvement in overall search performance. Latrémouille and Lee[32] constructed root menus for a given hierarchy by several methods and submitted them for evaluation by both experts and novices. Novices were almost unanimous in their choice of an appropriate menu; their choice also resulted in the most accurate performance. Experts were much less unanimous in their choice of menu.

Tombaugh and McEwen[70] compared alphabetic hierarchies to semantic hierarchies, and found no significant difference in performance of retrieval tasks. They also found no improvement in performance if both hierarchies were provided to subjects. A related study was conducted by Liebelt,[37] who constructed an organized and a random version of a small hierarchy, with arbitrary text strings as labels. Subjects made fewer errors in searching the semantically organized hierarchy, even after extensive practice. McDonald *et al.*[39] conducted a similar study in which multidimensional scaling techniques were employed to create a structured hierarchy. Significantly fewer errors were made in learning a structured hierarchy than a random one.

Iconic menu hierarchies can be much broader than those which rely on alphabetic labels; for example the system described by Herot[25] is virtually a single broad menu.

2.2. The depth/breadth problem.

Usually many different types of hierarchies can be constructed for a group of objects or utilities. One of the most natural questions is whether it is possible to identify an optimal number of options per menu and a corresponding optimal number of levels in the hierarchy. This, .the so-called "depth/breadth" problem, aims at choosing the hierarchy that promotes the shortest average search time, the fewest errors, or some weighted combination of these measures.

One trend in study of the depth/breadth problem is experimental analysis of various types of hierarchy. This type of research usually involves constructing several hierarchies of different depths and breadths, and then measuring subjects' ability to navigate the hierarchies. Much experimental work dates itself from Miller,[42] who found that a hierarchy of depth 2, breadth 8 resulted in better retrieval performance than with three other hierarchies on 64 items. Kiger [29] confirmed this result and also indicated that subjects preferred such a hierarchy. Snowberry *et al.*[61] confirmed Miller's result for random menus, but showed that if a menu with 64 options were structured into eight groups, then performance improved as a function of breadth (hence the 64 option menu was optimal). Landauer and Nachbar[31] tested highly structured hierarchies and confirmed Snowberry's results.

Schultz and Curran[57] showed that subjects preferred a single-page menu to a menu split into three pages. However, Dray *et al.*[12] showed that performance was better with a multiple menu design if subjects were initially trained with a single, broad menu. Tolle *et al.*[69] varied menu breadth and number of windows on the display, observing that subjects select more quickly from fewer windows and smaller menus.

On the other hand, Tullis[72] reports on large hierarchies used to access operating system functions and finds no significant difference in access time between broader and shallower hierarchies. Savage *et al.*[55] conducted a similar study and found that subjects preferred shallower hierarchies. Though the above experiments were conducted on pre-defined hierarchies, Raymond *et al.*[53] conducted an experiment in which subjects were permitted to create their own hierarchies on 200 items. The types of hierarchies constructed varied widely, and neither depth nor breadth was correlated with subjects' ability to retrieve from their structures.

A second trend in the study of the depth/breadth problem is formal mathematical analysis of the search process. This often results in an equation for total search time, whose derivative can be used to identify the optimal depth and breadth. The earliest work in this area is that of Thompson *et al.*[68] who suggested that the optimal breadth is between 3 and 5 options per menu, independent of the number of objects to be indexed. An analysis due to Uber *et al.*[73] suggests that the minimal optimum breadth is 8 options, and that greater breadth is preferred for expert users. Lee and MacGregor[34] conducted an analysis quite similar to that of Uber *et al.* much later but concluded that optimal breadth was between 4 and 8 options per menu under a wide variety of conditions.

Lee and MacGregor also showed that the optimal breadth was not dependent on the expected number of errors. Norman[43] proposes an analysis of breadth and system display time which maximizes user satisfaction, deriving optimal breadth results which are dependent upon the type of user. A similar analysis was performed by Sisson *et al.*[58] who suggested that optimal breadth was dependent upon communications delay.

The results obtained to date in depth/breadth research are inconclusive. At most, the studies have shown that it may be possible to identify a preferred depth and breadth *in a given situation*; what the studies do not provide is a clear definition of the situations in which their conclusions are valid. Thus system designers are left with the knotty problem of identifying the study which most closely resembles their proposed design. For example, which of the above studies is most applicable to auditory menus, as addressed by Kidd?[28] The set of semantics to be structured, the set of intended users, and the characteristics of the I/O devices are more significant than the inherent "optimality" of a particular hierarchical arrangement.

An analysis related to depth/breadth tradeoff was conducted by Witten *et al.*,[78] who investigated algorithms for constructing hierarchies given the maximum breadth of each menu and a probability distribution of the frequency of access of the leaves. The results are not strictly applicable to depth/breadth tradeoff, since the algorithms invariably choose the maximum breadth for as many menus as possible. Greenberg and Witten[21] experimentally showed that a hierarchy adapted to empirically determined frequency of access measurements results in a substantial increase in search speed and reduction in number of errors. Uber *et al.*[73] also propose an algorithm for constructing optimal hierarchies that is a simple adaptation of Huffman encoding.

2.3. Non-hierarchical designs.

Several kinds of non-hierarchy or relaxations of hierarchical structure have been proposed or implemented. ZOG hierarchies have non-hierarchical cross-links, but an explicit distinction is made between these and the hierarchical options.[38] Furnas[17] discusses the possibilities of directed acyclic graphs, where users can navigate upwards (i.e., towards the root or maximum) in several ways. Tompa[71] discusses characteristics of non-hierarchical systems in which several menus may be in effect at the same time. Two non-hierarchical systems were implemented and tested by Schabas and Tompa,[56] these being a *multiple hierarchy* similar to the directed acyclic graph proposed by Furnas, and a hierarchy with cross links. Schabas and Tompa report an experiment comparing the advantages of the two structures. Raymond [52] proposes relaxing not the hierarchy but instead the restrictions common for menus. If multiple selections are permitted at each node and subordinate options are shown, then multiple path navigation can be carried out in parallel. Brown[7] proposes controlling the design of non-hierarchical menu structures by principles similar to those restricting control flow in programming languages. Chan[10] describes a iconic menu system for organizing and and accessing operating system utilities which is based on an unrestricted network. Leclerc *et al.*[33] describe a documentation system based on directed acyclic graphs.

3. Menu systems.

A menu system is a software package which supports or automates the development of menu interfaces. Issues in menu systems include the design of menu systems, history and display mechanisms, and comparisons with non-menu systems.

3.1. Design of menu systems.

A frequent use of flexible menu systems is provision of a common interface to large bases of existing applications software. Such systems must provide an index to the various applications, as well as a means to enter parameters and display the output of the computation. Akscyn and McCracken [1] describe the use of the ZOG system for maintaining documents, software, electronic billboards, and project management. Poggio[50] describes the use of an object-oriented environment to develop menu-based interfaces to engineering software packages. Heffler[23] describes the design of a menu support system in a UNIX environment. Novatchev and Gabrovsky[45] discuss features of parameter passing. Goodwin[19] describes a system in which menus have been optimized for input by numeric keys, function keys, and alphabetic keys. Caswell and Gordon[9] describe an editor for personalizing an existing menu hierarchy which is itself menu driven.

3.2. History and display mechanisms.

A frequently employed feature in menu systems is the ability to "back up" to a previous selection and change it. I refer to these tools as *history mechanisms* since they resemble the history mechanism of the UNIX *csh* command interpreter. An early mention of this facility is in Clauer.[11] Snowberry *et al.*[59,60] conducted experiments which indicated that users performed more slowly in the presence of such a history mechanism than if they were shown upcoming options instead. On the other hand, Engel *et al.*, [14] suggest that the history mechanism is quite useful, though they point out that it is necessary to disengage the history when previous pages are being investigated (i.e., "the last page seen" means the last page exclusive of any consultation of the history mechanism). Apperley and Spence[4] discuss a history mechanism that was used successfully in a computer-based CAD system, but evidence from early experiments with the same mechanism in a general information database suggests that little improvement can be expected.[3] A more powerful mechanism with history-like capability is described in Schabas and Tompa;[56] this system not only maintains the path of the user's navigation, but can suggest other possible paths that could have been employed to reach the current menu.

The graphical display of the menu hierarchy in a tree-like form has been investigated as a training mechanism by Parton *et al.*[47] and Norman *et al.*[44] In neither of these experiments were statistically significant differences found which would indicate the superiority of graphical displays as a training method, though subjects seemed to prefer tree diagrams. Billingsley,[6] on the other hand, found a significant improvement in subjects' ability to search when trained with a tree diagram. Kastner and Widdel[27] suggest providing both history and a plausible future path; the notion of a default path is also present in Apperley and Spence.[4]

Yoder *et al.*[79] describe how the history of a user's session is maintained as part of the hierarchical database in the ZOG system.

3.3. Comparisons with other systems.

There have been several studies comparing menu systems to other types of interface. Geller and Lesk[18] compared menu hierarchies to keyword systems in two retrieval situations. They found that keyword systems were preferred when the desired information was known and the domain was predictable, whereas menu systems were preferred when the domain was unpredictable. Vrins *et al.*[74] compared menu hierarchies to both online and paper-based alphabetic keyword lists; their subjects performed better with (and preferred) the paper-based keyword list, but performed equally on the two computer-based indexes. They suggest that a hard copy index is better at prompting keywords when subjects are unsure which word to use. Hauptmann and Green[22] compared menu, command, and natural language interfaces and found no significant effect of language type on time, user satisfaction, or number of errors. Hauptmann and Green suggest that the task is a greater determinant of performance than language type. This result was confirmed by Whiteside *et al.*[77] who compared menu, command, and iconic languages and found no correlation between language type and performance. They suggest that the quality of an interface language is more important that its type. Isa and Ogden[26] describe the development of a product which incorporates menu, function key, and command line languages for accessing a hierarchical command structure. Subjects preferred different methods for different functions within the same product; as the product evolved, subjects' preferences changed. Isa and Ogden suggest that the interaction of these languages with each other and with specific application characteristics is complex, and that no simplistic recommendation is dependable.

4. Acknowledgements.

My thanks to Frank Wm. Tompa, who continues to patiently and effectively supervise the research of which this survey is a part.

5. Short Summaries.

Several of the papers describe experiments on different types of hierarchies, varying the number of options per menu and the number of levels in the hierarchy. I use a simple shorthand notation to describe these hierarchies: a sequence of numbers denoting the number of options at each level. Thus 5-5-5-5 is a hierarchy of depth 4 with 5 options at each level, while 2-3-3 is a hierarchy of depth 3, with 2 options at the root and 3 options at the other levels.

Akscyn and McCracken[1] discuss the use of ZOG as a database management system. Each ZOG frame can be considered to be a set of records of different types, including title, test, local and global options. Each record consists of text, a pointer to some other frame, an optional action, and a display position. Given this simple format ZOG is used to maintain documentation, software, electronic communications, project management and other applications. Documents are maintained as sub-hierarchies; the hierarchical nature makes browsing simpler and can be used to automatically generate formatting commands for a hardcopy version. A hierarchical arrangement also makes it convenient to directly attach cross-references, referee's comments, and other ancilliary information. When used as a software maintenance system, each ZOG frame represents a block of code, and each option on a frame is a statement. ZOG can also support electronic communication; instead of "sending" messages, a user adds frames to the relevant part of the database. The authors discuss the advantages of using a hierarchical navigation system for standard database problems such as multiple versions, aggregation, reliability, flexibility, and accessibility.

Allen[2] conducted an experiment to investigate cognitive factors in the use of hierarchical menu systems. 10 subjects searched a menu hierarchy in two sessions, with a two-week interval between sessions. The hierarchy was at most 4 levels deep, and contained 60 menus and 240 leaves. 42 of the leaves were chosen as targets; 21 were used in the first session and all 42 were used in the second session.

Subjects averaged 3.5 seconds to respond to a menu. Response time was significantly correlated to the number of options per menu, the serial position of the option, whether or not the subject committed an error at the menu, the distance of the menu from the root, and the distance of the target from the root. Increases in any of these factors led to increases in response time, except for distance of the target from the root, which was a U-shaped function. Subjects averaged 0.062 errors per menu. Error rate was significantly correlated to the distance of the target from the root (increasing with increasing distance), the spatial position of the option (left or right), and the type of information being categorized.

Apperley and Field[3] describe the design of an experiment to evaluate two types of menu system. The control system provides simple hierarchical access to a videotex database; the enhanced system supplies in addition feedback about the current context and a means to retreat to any previous position in the search path. It is intended that subjects conduct a fairly complex search task requiring retrieval of several pages. *Note: Apperley discussed some results of the experiment at the conference. The same amount of time was taken for searches in the two systems, but searches in the standard system usually involved more frames.*

Apperley and Spence[4] discuss a number of features of good command dialogues, and describe a hierarchical menu command system which illustrates these principles. *MINNIE* is a command system for a computer-aided tool for design of electronic circuits. Its main features include a continuous presentation of previous choices, the ability to retreat to any previous choice, and provision of a default selection mechanism. This last can provide either a predefined path component in the hierarchy, or can dynamically remember the most recent choice from a given node. *MINNIE* also includes *multiple* nodes, in which cancellation of a selected option results in automatic reselection of the next option in the sequence, and *parameter* nodes, which permit the user to adjust parameters which do not affect the choice of the next subordinate (i.e. there is only one subordinate from parameter nodes). The use of parameter nodes permits adjustment of several parameters without imposition of an artificial hierarchy on the information.

Bewley et al.[5] describe experiments conducted on the Star workstation's interface. The first experiment tested various schemes for selecting text elements with a mouse. The features leading to most error were use of three buttons, use of multiple clicking, and "drawthrough" (holding the mouse button down, moving the mouse to the desired position, releasing the mouse button). The second experiment compared four sets of icons. Subjects provided short descriptions for the icons and were asked to match up names with icons. The distinguishability of icons was tested by having subjects search for them in a random display. Once learned, there was little difference in correct recognition of an icon in the sets. However, icons in sets with more visual variety were more quickly selected than those in sets with less variety. The third, more informal experiment tested a scheme for drawing lines with a mouse.

Billingsley[6] conducted an experiment to test the advantages of providing subjects with a pictorial map of a menu hierarchy. Subjects were required to search for 18 targets in a hierarchy of 30 menus. Subjects were split into three groups, and performed an initial set of searches on the hierarchy, including 9 of the targets. Subjects in the data index and data map groups were then permitted to study paper descriptions of the hierarchy, which were a non-spatially organized index (list) and a spatially organized ("tree diagram") index, respectively. Subjects in the control group were given no additional information. All groups then performed searches on all 18 targets. Subjects who had studied the manual aids made fewer choices than those in the control group. Subjects who had studied a data map performed significantly better than either of the other groups when searching for the 9 targets that had not been included in the initial set of searches.

Brown[7] proposes the restriction of menu networks according to principles similar to those that restrict control flow in programming languages. It is argued that if networks are composed of a small number of structured subgraphs which can be combined by concatenation or nesting, the network structure will be both easier to understand and easier to debug. The structured subgraphs correspond roughly to programming language *case*, *if — then*, and *while(case)* constructs. Several structures which do not fit the proposed methodology are considered.

Card[8] investigated alphabetical, functional, and random order of command menus with 18 options. The alphabetical arrangement resulted in the fastest retrieval, followed by functional. The random arrangement was four times slower than the alphabetical arrangement. Monitoring of subjects' eye movements showed that visual search can be adequately modeled as random sampling with replacement. Differences between arrangements disappeared when subjects had engaged in extensive practice with an arrangement. Card discusses subjects' learning behaviour with a perceptual chunking analysis.

Caswell and Gordon[9] describe a menu-driven editor for personalizing hierarchical menu structures. The editor is invoked by using a fixed selector provided on every menu. The editor contains facilities for browsing the hierarchy, for locating menus by name, and for finding menus which have become disconnected from the hierarchy during editing. Options can be added or modified with a small form which is completed by entering menu selections. Selectors can be rearranged with a "sticky finger" operation; in this mode a selector is deposited to a holding location and the next selector chosen is moved to its previous location.

Chan[10] describes the *Rooms* interface to the Waterloo PORT operating system. A room is a collection of up to 28 icons which represent executable activities. One such activity is a door to another room. Activities in a room can be moved, copied, or examined; in this latter activity various parameters of the activity can be adjusted. All icons in a room are the same size and occupy one of the set of predefined positions (i.e., icons do not overlap).

The rooms are used both by the system and by the individual to organize the working environment. Each individual enters the system in the *Lobby*; a correctly entered password leads to the user's *Office*. The user can obtain needed activities from the *Supply* room, or attempt to use remote workstations by entering the *Network* room. Users can also make their own rooms. Rooms have a minimum of one door, and there is no restriction on the destination, hence a collection of rooms is a network rather than a hierarchy.

Clauer[11] conducted an experiment in which subjects searched a database of terms in microbiology, organized as a hierarchy of some 22,000 nodes. Subjects were required to find 8 targets, and were informed that these targets were located at the fifth level of the hierarchy. The system maintained a record of the user's previous selections. The major determinant of search time was the number of errors made through misunderstanding of option labels. Clauer detected no effect of option order or breadth on search time.

Dray et al.[12] conducted an experiment to evaluate depth/breadth tradeoff in menu hierarchies. Two menu systems were constructed, the first with 24 options on a single menu, the second with six options on the root menu and between 3 and 5 options on each of the six submenus. Ten subjects were tested in a within-subjects design for response time in locating stimulus targets. The best performance was obtained in the latter trials on the multiple-menu system, but only by those subjects who learned the single-menu system in the initial trials. The best performance in the initial trials was with the single-menu system. The authors suggest that functions which are rarely used should be presented in broad menus to avoid the need for memorization, while frequently used functions should be placed in multiple-menu form.

Dumais and Landauer[13] investigated two related problems: partitioning of data into categories, and selection of appropriate names for the categories. A subset of Yellow Page headings was given to 310 subjects, who were asked to organize them into five reasonable categories. From these organizations, five consensus "supercategories" were determined by hierarchical clustering. Fifteen experts were asked to label the categories with a name or short description. Examples of the categories were also chosen by random selection, human selection, and computer selection of the most representative headings. Various combinations of headings and examples were given to 700 subjects, who then chose the appropriate category for subsets of the headings. Three examples were as effective a label for a category as either category names or names plus examples. A single example was a poorer label than a name. Dumais and Landauer noted that performance improved significantly in the single example condition if the category "miscellaneous" was removed.

Engel et al.[14] describe a videotex system which attempts to reduce the difficulty of searching menu hierarchies by increasing the breadth of the hierarchy, displaying a diagram of the hierarchy, and maintaining a history of the user's traversal. The system also permits a limited form of cross-links between pages.

The breadth of the hierarchy is increased by using a two-display system. Users employ both a primary screen which displays the current page, and a secondary screen which displays an iconic menu of several categories of pages, including the current page. The secondary screen represents pages with a small icon (a black square), and labels to identify groups of pages. Users employ a light pen or touch screen to select pages on the secondary display.

The secondary display also contains functions to access a history mechanism. Users can step back through their session a page at a time, or can jump to specific points in the lesson that they had previously flagged. The secondary display is also integrated with the cross-links; if the current page has a cross link to some other page within the domain of the secondary display, then the secondary display highlights the icon of the referenced page.

Feiner et al.[15] describe a system for creating and accessing graphical documents. Navigation through the document is performed by selecting *buttons*, which are iconic labels for menu options. The result of selecting a button is the display of its associated page (and perhaps the execution of an associated action). The most common button icon is a miniature of the associated page. An *index* menu contains a list of page miniatures generated by keyword selection. A *neighbour* menu contains a page and its set of neighbours (i.e., predecessors and successors) in the structure. A *timeline* menu contains miniatures of the most recently viewed pages arranged in chronological order.

Frankhuizen and Vrins[16] conducted two studies with hierarchical videotex systems. In the first experiment 12 subjects carried out six general search tasks; performance was highly correlated with the type of task performed. Generally, subjects made many errors and consulted irrelevant pages frequently. A majority of errors were committed within information providers' index pages. Subjects noted a desire to back up to previous index pages where difficult choices had been made.

30 subjects participated in a second experiment, which was designed to test whether subjects would follow the expected search path in solving a task. Only 60% of the problems were solved correctly. 26% of the choices made at the root of the hierarchy were incorrect; 16% of the choices at the second level were incorrect. The recommendations for improving the hierarchy include addition of explanatory material to the trouble-prone menus.

Furnas[17] discusses the characteristics of menu structures in which subtrees may have multiple parents (i.e., directed acyclic graphs). This type of representation is suitable when subclasses of the information belong to several superclasses, thus making it difficult to define a unique hierarchical organization. Menus in directed acyclic graph networks contain multiple "up" options as well as the traditional multiple "down" options. Though relaxing the strict hierarchical partition can make some searches less tedious, it also results in several new problems (difficulty in searching exhaustively, different user interpretations of menus due to different traversal histories, the possibility of several reasonable choices at each menu, lack of transitivity of class properities).

Geller and Lesk[18] compared menu and keyword-based retrieval in both a library environment and an electronic news service. The menu system was based on the Dewey Decimal hierarchy, but the category labels chosen were developed locally (rather than employ the official designated labels). Large lower-level categories were subdivided by date; if subjects knew a particular Dewey number it was possible to move directly to the subclass. The hierarchy contained over 3000 categories and was up to ten levels deep.

The keyword system performed exact matching on words in the title, author, or subject fields (after suffix stripping), and could also match date ranges. Subjects using the keyword system could also obtain a limited form of relevance feedback (the system would suggest as new keywords any information-bearing words in the titles of previously selected books or articles).

Three-quarters of library users preferred keyword searching, and this preference increased after experience with the menu system. Twice as many subjects located the information they were searching for with the keyword system as with the menu system. Similar experiences with an electronic news system showed opposite results: users preferred menu systems. The authors suggest that menu access is preferable when the database is simple and unpredictable or dynamic, while a keyword system is preferable when the database is complex and predictable or static.

Goodwin[19] describes the design of a hierarchical menu system to provide access to several distinct software packages. A key feature of the system is its optimization for three types of input device: programmed function keys, numeric keys, and alphabetic text keys. Options are provided with both numeric selectors and alphabetic selectors. The numeric selectors permit selection by either a programmed function key or the numeric key of the same number. The alphabetic selectors are one or two letter codes chosen from key words in the label of the option. These selectors can be combined to form pathnames or phrases, permitting fast access to known parts of the hierarchy.

Greenberg and Witten[20] investigated six methods of presenting a menu on alphabetically organized information. Each menu consists of a set of alphabetical ranges as its options, and may span over the whole alphabet or only a restricted subset. Each range can be displayed by showing both the upper and lower value or delimiter for the range (i.e., the lexicographically smallest and largest word in the range), or only the lower delimiter, or only the upper delimiter. The delimiter may be the full word or a maximum truncation of the word that still defines a partition. Combining the range techniques with the truncation techniques results in six possibilities.

48 subjects participated in an experiment which measured their scanning time and error rate in selecting target options from range menus. Subjects were classified as computer experts and computer novices. Each subject was assigned to one of the three range techniques and was exposed to both truncation techniques as well as two spans (the whole alphabet and a single letter).

Using both upper and lower delimiters resulted in slowest scanning speed. Using only lower delimiters resulted in the greatest error rate; using only upper delimiters resulted in the fastest scanning rate. Subjects scanned narrow span menus more slowly than wide span menus. Novice subjects had slower scanning speeds than experts and produced more errors with the lower delimiter technique. Truncation techniques do not affect speed or errors.

Greenberg and Witten[21] describe an experiment designed to test the validity of adaptive personalization of telephone directories. Two hierarchies were constructed on 2611 directory entries, each balanced so that average expected number of selections would be minimized, under different probability of access assumptions. In the *static* system the probabilities of access were all equivalent; in the *adaptive* system the probabilities of access were determined from empirical data. 160 telephone calls made by a single person were noted; 122 of these constituted the empirical data, while the remainder were used as stimuli. 26 computer science students used both systems to locate the 48 remaining names. Trials with the adaptive system resulted in 35% less scanning time and 60% fewer errors than with the static system. 32% fewer menus were scanned with the adaptive system, and 18 subjects indicated a strong preference for the adaptive system.

Hauptmann et al.[22] experimented with three types of input language for control of a graphing program. After training, subjects were asked to create pie, bar, and line graphs. None of time, user satisfaction, or number of errors was significantly dependent upon language type. In terms of input, the command language was most concise, followed by menu; natural language was most verbose. Problems noted with the menu language included rigid presentation, the need for examples, and the tedium of complete redisplay of each menu. Hauptmann notes that the extra effort of traversing the menu hierarchy did not seem to affect subjects' attitudes or time. A major conclusion is that the task and the underlying system are more significant than choice of pre-processor language.

Heffler[23] describes the MCIS system for developing and maintaining menu-based interfaces to applications software in a UNIX environment. MCIS menus are either general purpose (i.e., select a single option) or parameter menus, which are controlled sequential dialogues. Parameter menus can provide an explanation of the meaning of each parameter (and/or its possible values); parameters can also be altered before the menu is terminated. Parameters may be set to defaults which can be either static or dynamic. A static default is a preassigned value whose prompt is suppressed, whereas a dynamic default is a preassigned value whose prompt is displayed; hence dynamic defaults can be overriden by user. Each general purpose menu has a name and hence can be directly accessed from any point within a dialogue.

Menus can be created either by use of a controlled dialogue program or by a menu editor, which also permits modification of an existing menu system. The output of the creation program and the editor is a set of files containing the menu labels, help texts, and other data input by the designer, C functions used to interpret menu input and invoke run-time routines, and a shell script to compile and link the applications programs with the menu interface.

Hemenway[24] discusses psychological issues involved in the selection of icons for command menus. Icons typically depict either the operations carried out by a command or the objects operated on by the command, but can also depict the command by showing a "before-and-after" comparison. Paramaterized commands can be represented by paramaterizing an appropriate dimension of the relevant icon, and combinations of commands can be represented by combining the relevant icons, if possible. Paramaterization and combining can improve the user's learning and retention of the whole icon set, as well as supporting the learning of new command structures by permitting the user to make predictions about them. The effectiveness of icons depends upon the user's ability to discern what the icon depicts and which command is implied. If the icon is not a direct representation of the command tool or object, the effectiveness of the icon will depend on the quality of the analogy.

Herot[25] describes a data management system which employs icon menus. Data is presented as a set of icons which are spatially organized in an extremely broad but shallow hierarchy. Two display devices are used, one to show an overall view of the data space and one to show a magnified view of some small part of the data space. The data to be displayed can be derived either from a relational database or from a videodisc player; in this latter case either still frames or short sequences of frames may be viewed. If the data is relational, the shape and colour of icons can be dynamically determined from the values of the relevant tuple.

Isa and Ogden[26] describe four experiments undertaken to select an appropriate command language for a software product. In the first experiment, three methods of navigating a hierarchy were compared: dynamically-mapped function keys, command line, and menu selection. The hierarchy was a 5-x-y, where x was either 2 or 3 and y was either 0 or 2, depending on the category. Subjects were trained and practised in all three navigational schemes; approximately 4 hours was spent on the experiment. There were no significant differences in either the average time to complete the tasks or the number of extra panels viewed. When permitted to choose from the three methods, subjects selected the function key method 80%, the command line 20%, and the menu was never selected.

In the second study, function key and command line navigation were compared for an unbalanced hierarchy. The hierarchy contained 5 options at the root, of which 4 were leaves and one was a further branch of 4-5. Subjects were trained in both schemes and were permitted to use either at any time in the experiment. There were no significant differences between the two navigational methods. Function keys were used 88% of the time when direct access was possible, but usage dropped to 35% if two keys were necessary, and 18% if three keys were necessary.

The third study attempted to evaluate a navigation scheme using a combination of menu, command line, and function keys for a hierarchy which was too complex to permit fixed mappings of function keys. The hierarchy was an 8-x, where x was either 0, 5-5, or 7. The menu permitted cursor-based selection of options; command line navigation could be performed at all panels; limited sequential navigation could be performed with function keys. Several function keys were assigned to fixed functions, such as online help, "go", etc. Subjects used the menu 14% of the time, commands 34% of the time, and function keys 52% of the time.

The fourth study evaluated the navigational system tested in the third study, but in the context of a working application. This required a modification so that navigational commands were always identified as such. In addition, the menu panel did not support cursor-based selection of options. Subjects varied widely in use of the menu, from 1% to 23%. Subjects thought the menu should present more information than simply the names of the options.

Kastner and Widdel[27] describe the design of an experiment to investigate the effect of providing a graphical depiction of a menu hierarchy on subjects' performance in searching. The subject's current position in the hierarchy and a plausible future path would be presented in the graphical depiction at each step in the search.

Kidd[28] explores problems involved with the design of aural menus. Aurally presented menus have several disadvantages, including the inability of the user to control the rate at which information is presented, the increased processing load of both remembering and comparing the options of the menu, and the user's inability to make judgements about the menu's complexity (i.e., its size and format) until the whole menu has been presented. These considerations suggest that the factor controlling the success of auditory menus will be the degree of problem solving required to choose an option; two experiments were conducted to investigate this hypothesis.

In the first experiment, subjects were told to prepare a dinner for some business associates by contacting a recipe advice service on the telephone; menu options were chosen by dialing an appropriate number. Menus varied between two and six options, and subjects could request either a "help" message or a repeat of any menu. It was predicted that minimal problem solving would be necessary, since the domain of the options on each menu was well partitioned and the menu structure was sequential, rather than hierarchical. Subjects completed the task successfully, and response time appeared to be independent of either serial position of the solution option or the length of the option list.

In the second experiment, subjects attempted to retrieve specific items of information from a videotex database, in which information was organized hierarchically, and each menu contained between two and ten options. It was predicted that significant problem solving would be required since the domain of the options on each menu was overlapping and the menu structure was hierarchical. Subjects completed the task successfully, but found it very difficult, and often requested repetition of menus.

The results of the two experiments indicate that a brief introduction to each menu could help to provide the user with cues about the upcoming classification. Additional recommendations include using auditory cues (e.g., different tones or voices) to distinguish between questions and options and user control over pacing of the menu presentation.

Kiger[29] examines depth/breadth tradeoff for several different types of hierarchies, including 2-2-2-2-2-2, 4-4-4, 8-8, 4-16, and 16-4. Subjects were not presented with a target word to locate, but instead a task which necessitated some information retrieval. Subjects were permitted to navigate both up and down the hierarchies, and each subject was tested on each hierarchy. Preference, speed, and accuracy were best with the 8-8 hierarchy, and worst with the 2-2-2-2-2-2 hierarchy.

Koved and Shneiderman[30] describe several types of *embedded menu* systems, in which menu options are integrated with other display information (e.g., text or graphics). An experiment was conducted which compared an embedded menu system to an explicit menu system. Subjects answered more questions correctly and viewed fewer screens when using the embedded menu system. A second experiment compared embedded menus to sequential paging through an online manual. Subjects solved problems more quickly with sequential paging, but preferred the embedded menus. The authors suggest that the result of this second experiment was affected by poor design of the embedded menu system. In a third experiment with an enhanced embedded menu design, subjects performed twice as fast, viewed fewer pages, and spent less time on individual pages than with the sequential paging design.

Landauer and Nachbar[31] studied depth/breadth tradeoff for hierarchies of 16-16-16, 8-8-8-8, 4-4-4-4-4-4, and a breadth-two, depth-sixteen hierarchy. Two sets of hierarchies were constructed, one with random words at the leaves, the other with integers at the leaves. Menu pages showed the ranges of the options (alphabetically or numerically). Subjects were shown a target and then asked to navigate through the hierarchy; the program disallowed erroneous selections. Landauer and Nachbar hypothesized that menu selection would obey the Hick-Hyman law ($t=c+k\log b$ where b is the number of alternatives, t is the average response time, and c and k are constants), and Fitts' law ($mt=c+k\log\frac{d}{w}$ where d is the distance to be moved, w is the width of the target, and mt is the movement time). These laws closely described the experimental data.

Latrémouille and Lee[32] report the results of two experiments designed to evaluate alternative menus for the root of a menu hierarchy. In the first experiment, 10 subjects familiar with an existing videotex system were asked to rank menus chosen from a set of 12. 6 of the 12 menus were produced by experts working independently. The remaining 6 menus consisted of the original root menu, a version modified to avoid errors that had been observed in experimental searches, a menu created by a professional librarian, and augmented versions of each of these three. The augmented menus were constructed by adding extra descriptions of the options' content to the menus. There were no significant preferences noted for any of the menus.

In the second experiment, subjects who were unfamiliar with the videotex system were asked to solve queries by selecting options from the 12 menus used in the first experiment. The queries were generated by choosing randomly from among the 900 documents contained in the database identified by the alternative root menus. Subjects were also asked to rank the menus. These "naive" subjects had a high degree of preference for menus with descriptors, and a high degree of consistency in the number of errors made on menus; significantly fewer errors were made on pages with descriptors. Moreover, subjects' preferences and performance were correlated to a high degree. The best menu overall was the modified version of the original menu with descriptors.

Leclerc et al.[33] describe a menu-based system for browsing computer documentation. The structure of the documentation is a directed acyclic graph, which is based on assigned keywords and is stored separately from the content of the database. The system permits the user to view up to four levels of the hierarchy simultaneously, which can be navigated in the usual fashion or can be searched with keywords.

Lee and MacGregor[34] investigate depth/breadth tradeoff analytically. If subjects perform exhaustive search of all options on a single menu, then the optimal number of options a is given by $a(\ln a-1)=\frac{(k+c)}{t}$, where t is time to examine one option, k is the time to press a key, and c is the computer's response time. If subjects employ a terminating search on a menu's options (the correct option is recognized when it is seen), the optimal number of options is given by $a(\ln a-1)=1+\frac{2(k+c)}{t}$. The optimum number of options not affected by errors in the search process. The optimum number of options for a wide range of k, c, and t is between 4 and 8.

Lee et al.[35] report on the experiments as described in Lee *et al.*[36]

Lee et al.[36] present the results of six experiments in design of videotex hierarchies.

In the first experiment, subjects searched a simulated hierarchy (presented on index cards) of 900 pages for solutions to 16 queries. Errors at any menu resulted in a prompt to choose another selection. Subjects made at least one error on half the queries; most errors occurred on a relatively small number of the menus; over half of the errors were made within the top two levels of the hierarchy. Representative users were asked to reclassify or rename six menus (those on which 80% of errors had occurred), and the experiment was repeated with a different group of users. 40% fewer errors were made in the repeated experiment.

The second experiment duplicated the conditions of the first except that it was presented online and four of the queries had as the appropriate answer "no solution". Subjects averaged twice as many page accesses in attempting to answer the "no solution" queries as queries for which a solution was present. Users located almost 90% of the solutions to queries for which a solution was present, averaging 4.5 menu pages per query.

The third experiment extended the second by modifying the hierarchy with three types of deliberate defects: miscategorization, synonymous labelling, and vague labelling. Four of each type of defect were introduced into the hierarchy. Subjects located only 60% of the solutions to queries for which a solution was present, averaging 10.6 menu pages per query. Miscategorization reduces the probability of correct location to less than 0.01. Synonymous or ambiguous labels had much less effect.

The fourth experiment repeated the second, except that an operational videotex system containing some 1500 pages was employed instead of a simulation, and less specific queries were included (queries whose solution could be found within the index pages or at several places in the hierarchy). Subjects verbalized during their searching process. Users successfully completed 72% of solutions to general and specific queries for which a solution was present, averaging 4.7 pages per query. The probability of committing an error (i.e. a navigational step which was not on the shortest path to the solution page) was 0.35, higher than in the companion second experiment.

The fifth experiment was based on the observation that most errors were committed at the first two levels of the hierarchy. Twelve root pages were constructed, including the root and modified root from experiments 1 and 2. A third root was constructed by a professional librarian. Three more were constructed by adding descriptive phrases to the first three root pages; six others were contributed by independent experts. Ten videotex experts were asked to rank the twelve root pages; no significant agreement was observed.

The sixth experiment repeated the fifth except that naive subjects were employed to rank the root pages; the subjects were also required to search for information with each of the twelve root pages. There was a high degree of agreement among naive subjects for root pages, preferring pages with descriptors to similar ones without descriptors. Subjects also performed significantly better on pages with descriptors than those without.

Liebelt et al.[37] tested the effect of changing the semantic organization of a fixed 2-2-4 hierarchy. Six conditions were generated by choosing 16 targets from one, two, or four categories, and arranging them at the leaves of the hierarchy in both random and organized fashion. Subjects navigated the hierarchy by selecting from menus of single alphabetic characters, and were required to perform the task until each of the 16 targets was correctly located. Subjects committed significantly more errors at level 3 than at the first two levels of any hierarchy. Subjects committed fewer errors in organized menus and showed quicker selection at the first level. There was no recognizable effect over number of categories.

McCracken and Akscyn[38] describes the ZOG philosophy of menu system design, which includes a strong preference for hierarchies, fast response, no hidden options, and no scrolling within a menu (although the authors mention a recent version of ZOG which permits several frames (i.e., menus) to be displayed at one time). Application areas include information systems of various types, including datbase, management information, electronic mail, and document systems. ZOG also manages its own software by equating frames with blocks of block-structured programming languages. A major project involving ZOG was an application system developed for the USS Carl Vinson. This system maintains the ship organization and regulation manual and interfaces to an expert system aiding in launch and recovery of aircraft. ZOG's strengths are its ability to assimilate many applications, its ease of learning and use, its support of large databases, and its simple view of multiple processes. ZOG's weak points are its sacrifice of efficiency for integration, its dependence on fast mass storage and communications technology, the tendency for users to get lost, and its inability to handle highly dynamic data.

McDonald et al.[39] report on the success of a technique for organizing menu hierarchies. In the first experiment ten subjects rated the similarity of all possible pairs of 34 word processor functions. Of these subjects seven produced consistent ratings, from which four clusters of four functions each were extracted to use as stimuli in the second experiment.

In the second experiment two 2-2-4 menu hierarchies were constructed, one using clusters derived from the first experiment and one with random assignment of options. The labels on the menus of the hierarchies were randomly selected consonants. 24 subjects learned these hierarchies in either a sequential response (one consonant at a time) condition or all-at-once (all three consonants). Subjects practiced with a random organization until they could select all targets correctly during a single block.

Subjects made more errors on the last menu than on the first two; subjects also made more errors on all menus in the random condition. Subjects took more time to select from the root menu than from other menus. The authors suggest that the lack of significantly improved performance for the clusters is partly a result of the subjects' unfamiliarity with word processing operations, and the small size of the sample producing the clustering in the first experiment.

McDonald et al.[40] conducted an experiment to investigate whether alphabetical ordering is better than semantic ordering of a menu. Five types of menu were constructed, each containing 4 columns of 16 options. The types of organization were random, alphabetical, and three types of categorical organization — each category or column was itself organized categorically, alphabetically, or randomly. Two types of targets were employed: an explicit option label (i.e., as shown on the menu) and a brief definition of the desired option. 109 subjects were assigned to a single type of menu and target for 320 experimental trials.

Subjects were significantly faster with explicit targets than defined targets, and the categorical-categorical organization was the optimum. With definition targets all categorical organizations were faster than the alphabetical organization during the first block of trials, which was not the case for explicit targets. The random organization resulted in shorter times for definition targets than for explicit targets in every block except the first. The authors suggest that this last result is due to the need for more involved processing to solve definition-type queries.

McEwen[41] conducted an experiment to evaluate retrieval in an existing videotex system organized as a menu hierarchy with approximately 500 menus and 900 pages of information. 24 subjects solved 16 queries of which 10 required specific answers, three were general, and three were for information not contained in the database. Half of the subjects were asked to verbalize their search process. Three types of error analysis were considered, differing in the counting of errors. In the least stringent analysis, subjects averaged at least one error per search problem, and the probability of making an error on any menu was 26%. In the most stringent analysis subjects averaged more than three errors per search problem and the probability of making an error on any menu was 35%. Subjects viewed on average twice the number of pages necessary to locate information by a minimal path. Subjects successfully completed 72% of the queries for which a solution was present. 22% of all errors were made on the root menu; 47% of all errors were made in the top two levels of the hierarchy. However, normalization of error rates by the number of times a menu was viewed indicated that one of the menu pages had an error rate of over 70%.

Miller[42] investigated depth/breadth tradeoff for 2-2-2-2-2-2, 4-4-4, 8-8, and 64 (i.e. one level) hierarchies. Subjects were required to locate a target word; time to locate and accuracy were the dependent variables. Subjects performed best with the 8-8 hierarchy. No tradeoff between speed and accuracy was detected. Miller emphasizes that the results were based on experimental conditions requiring rapid performance from the subjects.

Norman[43] analyses the effects of different menu breadths and system response times on "user satisfaction". User satisfaction or $U(x)$ is defined to be $U(x)=kx^p$, where x is menu breadth or system response time, and k and p are parameters to be determined. Norman gives personal estimates of satisfaction in order to identify the parameters. The relationship between menu size and time to present the menu is $S{:}T=\sigma+\frac{S}{\beta}$, where S is the number of characters, T is the time in seconds, σ is the system response time and β is the display rate in characters per second. Overall user satisfaction is a weighted sum of the satisfaction measures for breadth and response time, where the weights are user dependent. Optimal satisfaction is obtained either by maximizing menu breadth (and hence a maximal system response time, i.e. for novices) or by minimizing system response time (and hence a minimal menu breadth, i.e for experts). Norman also presents a brief comparison of the attributes of menus and command languages.

Norman et al.[44] conducted an experiment in which four different training strategies were used for teaching subjects a 3-3-3 hierarchy. Subjects either practised unguided experimentation, or studied three kinds of manual aid: a list of all possible paths through the hierarchy, a randomly arranged presentation of all menus, and a tree diagram of the menus. The non-leaf nodes of the hierarchy were labelled with nonsense phrases. Subjects were permitted to study the material or experiment with the hierarchy for 5 minutes, then performed retrieval; the study-test sequence was then repeated. After the experiment subjects were asked to recall as much of the hierarchy as possible. Only minor differences in performance were obtained across training strategies. Significant correlations were found between recall of the hierarchy and performance for all strategies except the list of possible paths. The authors recommend that infrequent users of a menu system learn the hierarchy by studying a tree diagram.

Novatchev and Gabrovsky[45] describe the menu support system for an interactive environment. The basic features of the menu support system are dynamic menus, user profiles, and independence of the dialogue from the underlying semantics. Menus are created dynamically from menu descriptions by the menu interpreter, which controls the dialogue. The dynamic menu is then presented with fields preselected or filled in according to the user profile, which uses either default values or the most recent (correct) user input. Novatchev and Gabrovsky suggest that the independence of the menu interpreter and dialogue from the semantics of the programs simplifies applications programs and permits rapid prototyping.

Parkinson et al.[46] conducted an experiment to investigate various types of structuring of within a menu. 64 options were chosen from eight natural categories and presented in 4 columns of 16 options each. 10 different arrangements were constructed, the first 8 grouping together words by category, and the last two organizing alphabetically by column and row, respectively. The 8 category groupings were the possible combinations of using alphabetical or categorized order within categories, spacing or not between adjacent categories, and grouping by column or by row. Subjects achieved over 97% accuracy in all configurations, so differences were confined to reaction times. Menus organized by columns were searched more quickly than those organized by rows; menus with spacing between groups were searched faster than menus with no spacing. These two factors were additive (menus with spacing and columnar organization had twice the improvement). Organization by alphabet or categorization did not influence performance.

Parton et al.[47] compared four different training strategies for teaching subjects a 3-3-3 hierarchy. Subjects either practised unguided experimentation, or studied one of three types of manual aid: a list of all possible paths through the hierarchy, a randomly arranged presentation of the menus, or a tree diagram of the menus. The menus were reasonable semantic clusters of their subordinates. Subjects studied the training materials for 12 minutes, and then were required to search for target words in the hierarchy. Subjects who studied the tree diagram recalled more options after the experiment, and rated this method highly. Though differences were noted in both number of targets located and number of selections to reach a target, these were at best marginally significant. The authors suggest that unguided experimentation and tree diagrams are superior training strategies.

Penna[48] describes a videotex system with enhanced menu access and personal structuring features. Users select options by touching appropriate areas on a touch screen. Menus may contain both immediate subordinates and grandchildren, both to provide context and so that users can navigate quickly to frequently-accessed parts of the database. Users can save and retrieve pages from a personal store, which maintains up to eight user-definable categories and a scrollable list of pages. The scrollable list contains both the user-defined name and a miniature representation of the page. The scrollable list can be viewed in its entirety, or its contents can be filtered by category.

Perlman[49] reports two experiments involved with presentation of menu options. In the first experiment, subjects searched for targets in several sizes of menus of either arabic numerals or words, in sorted or random condition. The time taken to select an option was a linear function of menu size; the slope of this function was greater if the menu was random. In the second experiment, subjects searched for targets in a menu, indicating their choice by entering letters or numbers that were compatible (i.e. first letter of the selection, ordinal position of the selection) or incompatible with the choices. Subjects took longer with incompatible letters than incompatible numbers, but took longer with compatible numbers than compatible letters.

Poggio[50] discusses the advantages of using object-oriented programming for developing a general menu interface to be used on a wide variety of display devices. EAGLES is written in Objective-C, and is an attempt to provide a uniform interface to a large collection of engineering software. The menu system consists of object classes for menu items, horizontal menus, vertical menus, and menus. The class menu is a subclass of the base class OrderedCollection already provided in Objective-C. Various other classes are defined to organize and control the execution of menus and associated windows.

Pollard and James[51] describe several experiments conducted on menus in hierarchical videotex systems. The first experiment investigated three formats for menu layout common to the Prestel system. The authors do not describe the layouts, but found no significant differences between them.

The second experiment investigated three different types of alphabetic coding for keywords to be accessed in a hierarchical menu format. In the first type of coding a specific two-digit number was assigned to each letter; in order to enter a word it was necessary to enter the codes for each letter in the word. In the second type of coding the alphabet was recursively divided into ten sections; at each menu only a single digit was entered. In the third type of coding a specific two-digit number was assigned to the first letter of the word, but thereafter only the grammatically possible letters were coded, and they were numbered consecutively. 100 words were indexed in each of these hierarchies, and 66 subjects searched for 30 of the words. Subjects keyed their choices with a standard videotex keypad; after each selection they were presented with a menu of the next set of choices in the condition to which they had been assigned. The second type of coding resulted in significantly more errors and significantly slower search times than the other types. The third type resulted in significantly lower error rates than the first type.

In the third experiment 40 subjects evaluated two types of numeric coding for alphabetic characters. In the first type of coding the alphabet was simply numbered from 01 to 26, with leading zeroes for the first nine letters. In the second type the alphabet was numbered from 11 to 36. Subjects were required to enter the codes for forty words. Eight subjects omitted leading zeroes altogether; of the remainder of the subjects, the majority of errors were omission of leading zeros. The authors recommend the second type of coding.

Raymond[52] presents an analysis of videotex systems, classifying various problems and proposed solutions. A major unsolved problem is the need for multiple means of accessing videotex pages, and it is suggested that users should be able to change or create structures themselves. A menu variant known as multi-menus is proposed. Multi-menus display several levels of a hierarchy and permit more than one selection at each level. Multi-menus permit some flexibility in search of menu hierarchies, and reduce the need for optimum selections at each menu. A general tool for modifying structure is also proposed.

Raymond et al.[53] report an experiment in which 10 subjects organized 200 proverbs as a menu hierarchy and then conducted retrieval of selected proverbs from the structure. A measure of subjective quality of categories known as *variability* was developed and used to rate the subjects' structures. Subjects' retrieval performance was strongly correlated to the variability of the structure, but not correlated with objective measures of the structure (i.e., depth, breadth, number of categories in the root menu). Subjects performed more poorly in the computer-based version than in a previous manual experiment. Though subjects were encouraged to use a spatial dimension to organize their structures, little evidence was found of this type of organization.

Robertson et al.[54] describes the design of the ZOG menu system. ZOG consists of a network of *frames*, controlled by a very high-speed terminal employing touch-screen input. Users navigate between frames by simply touching an option on the frame's menu. ZOG design principles emphasize frame simplicity, speed of response, and a large network of frames organized into subnets. ZOG can be used to both organize and maintain information (data, programs) and to activate it (i.e., control the execution of programs). Variants of ZOG are described, including one with response time of 0.1 second per frame. This response time creates a qualitatively different kind of interface than the standard ZOG system (response rate of approximately 5 seconds per frame). Some problems with ZOG are the difficulty of creating large networks, the ease with which users become lost in a network, and the fact that users often do not read frames properly or completely. The authors suggest that the basic disadvantage of menu systems is the low rate of information transfer from user to machine. This rate is governed by the complexity of choosing from a large menu or the communication delay when choosing from a sequence of small menus. This basic disadvantage is alleviated by ZOG's use of high communication speed and large networks of small menus.

Savage et al.[55] conducted two experiments in which subjects performed three types of computer-related tasks through a menu-based interface. Twelve subjects performed programming tasks, nine performed system operator tasks, and ten performed workstation operator tasks. The menu systems were hierarchically arranged, with no more than nine options per menu. Subjects could avoid navigating the hierarchy if the name of the desired command was known. Three types of error were noted, along with the time consumed in performing each task and a probability analysis of the likelihood of error at each menu. *Inconvenience* errors were those in which subjects found the correct function but in a non-optimal fashion. *Path* errors were those in which subjects chose the wrong function. *Function* errors were those in which the correct function was obtained, but the wrong parameters were entered.

The probability analysis showed that the most error-prone menus either contained a small number of erroneous options which were highly likely to be selected, or else all options seemed unlikely and so subjects chose at random. Subjects indicated that the wording of the labels was a significant problem. Correction of these errors resulted in improved performance in the second experiment. The authors noted that subjects preferred more levels of hierarchy with fewer options per menu, and suggest that this may be due to the realistic nature of the tasks performed and the fact that menu labels were phrases rather than single words.

Schabas and Tompa[56] describe an experiment in which subjects searched two types of hierarchies on 199 items. The *multiple hierarchy* consisted of several different sub-hierarchies or "contexts"; each node in the database could belong to as many as three contexts. Subjects who searched the multiple hierarchy viewed menus consisting of the options of the current node and the contexts it belonged to. Each context was identified by showing the relevant sequence of menu options from the root to the current node. Subjects were located both at a node and within a context; they were permitted to move to the node's subordinates, to back up within the context, or to move to another context (and hence potentially display a new set of subordinates).

The *cross-link* database was based on the same set of 199 items, but was constructed in more traditional hierarchical form. In order to maintain the information about multiple contexts in some form, each node in the cross-link database contained a *shadow page* which identified by cross-links the nodes which were siblings in the multiple hierarchy database (i.e. for every pair of nodes which had a common parent in the multiple hierarchy database, a cross-link would be installed in the cross-link database).

Subjects trained for one hour, including demonstrations and practice use, and then were allotted one hour to solve twelve queries. Half of the queries required location of a single item (simple); half required location of a cluster of items (compound). For simple queries, the multiple hierarchy was more than twice as effective (in terms of number of steps and time required) than the cross-link hierarchy. For compound queries, the cross-link hierarchy was somewhat more effective than the multiple hierarchy.

Schultz and Curran[57] describe an experiment comparing a single large menu to a paged menu system. Subjects were presented with either a single menu of 56 options, or 3 sequential menus containing 15, 25, and 16 options, respectively; menus were either alphabetically or randomly ordered. Subjects were required to move a cursor to the target option in order to select it. Both *goal retrieval* time (time from display to first key press) and *goal selection* time (time from display to last key press) were measured. Subjects presented with the single menu were almost 50% faster in goal selection time. There was no advantage of alphabetical arrangement over random arrangement.

Sisson et al.[58] analyse major components of time usage in menu traversal: processor time, serial communications time, operator decision time, and response time. Based on Snowberry et al.'s[61] experiments, Sisson suggests that serial communications time can determine optimal hierarchy structure. Sisson's analysis indicates that total cumulative time is minimized by 4-4-4 at baud rates between 100 and 300, by 8-8 at baud rates between 600 and 4800, and by 64 at baud rates over 19200. 2-2-2-2-2-2 is never optimal.

Snowberry et al.[59] examined the effects of adding "help fields" consisting of upcoming options (i.e., options that might be selected after the present level) to menus. Subjects searched 2-2-2-2-2-2 hierarchies in one of four conditions: control, continual display of target word, previously selected options displayed, and upcoming options displayed. Only subjects in the upcoming options condition performed more accurately; subjects in the previously selected options condition were significantly slower than all other groups. In a second experiment, it was found that if subjects had extensive practice with the control menus (no help fields, no target word displayed), there was no significant difference between the groups.

Snowberry et al.[60] is a more detailed report of work described in Snowberry *et al.*[59]

Snowberry et al.[61] investigated depth/breadth tradeoff in five hierarchies: 2-2-2-2-2-2, 4-4-4, 8-8, and two types of 64. The single level hierarchy (i.e., 64) was shown in both randomized and categorized form; the categorized menu contained subgroups of 8 semantically related words. Subjects were presented with a target word to be located in each hierarchy. The best overall performance (lowest elapsed time and highest accuracy) was obtained with a categorized 64 menu. Subjects performed most slowly and inaccurately with the 2-2-2-2-2-2 hierarchy, and made over half their errors in this hierarchy at the top two levels. The authors suggest that lists of future options could improve performance in deeper hierarchies.

Somberg[62] conducted an experiment which evaluated four different arrangements of menu options. 32 subjects selected designated target words from a menu of 20, the words being chosen from a set of 2000. The menus were arranged in one of four conditions: alphabetic, random, probability of selection, or positionally constant. In the positionally constant condition each word in the set was assigned a specific position for any menu in which it might appear. Two different display conditions were employed, a single centered column of 20 words, and 4 groups of 5 words placed in the corners of the screen. The serial order for the display conditions was top-to-bottom and left-to-right (for the groups condition). Subjects were assigned to a single organization and display condition, and conducted 6 blocks of 82 searches.

Accuracy was extremely high for all conditions, so only response time was considered. Overall, alphabetic and probability orderings produced identical response times, while random orderings produced the slowest response times. The positionally constant condition was the only one to show substantial improvement during the session, and it produced significantly faster response times than all other conditions after the third block of trials. The two display conditions did not produce statistically significant differences. The author concludes that rule-based orderings require little training, but are less effective than positional orderings if extensive practice is possible.

Somberg et al.[63] conducted an experiment on searching within a computer menu. 24 subjects were shown a target word and then a numbered list of categories, one of which described the target. Subjects were instructed to select the appropriate category and then enter its number. The number of options per menu was 3, 5 or 7; the order of the options and the position of the target's category in the menu was randomly determined. Stimuli were chosen from both familiar and unfamiliar categories.

Since the accuracy of selection was uniformly high, the main independent variable was reaction time. Reaction time was a linear function of serial position for both 5 and 7 option menus, and not significantly different between these menus. Reaction time was significantly shorter for 3 option menus. There was a significant effect of position on response time; options closer to the bottom of the menu required more time than those closer to the top. The relationship between reaction time and position was found to be linear. Unfamiliar stimuli resulted in longer response times, but the slope of the function was the same as for familiar stimuli. The results indicate that for this task the search process was top-down and self-terminating (that is, terminated when the correct category was found).

Somberg and Picardi[64] attempted to determine how familiarity with a target word affects menu search. A previous experiment had indicated that response time was a linear function of the serial position of the required menu item. Familiarity with the target word had a significant effect on overall response time, but did not influence scan rate. It was hypothesized that the familiarity effect was located either in subjects' attempt to establish a probable category before beginning search or in the confirmation of the correctness of a category; either of these could result in longer search times with unfamiliar target words. The first hypothesis was tested by varying the delay between presentation of the target word and presentation of the menu; the second hypothesis was tested by including target words for which no category was present. Neither hypothesis was supported. Somberg and Picardi suggest that familiarity affects the distribution of response time.

Teitelbaum and Granda[65] investigated the effect of varying the position of ancilliary information within a menu. Forty subjects searched four-level hierarchies of menus with between 2 and 5 options per menu. The menus also contained a title, page number, topic heading, instruction line, and data entry area; these five pieces of information were presented either in fixed or variable positions. Subjects spent one hour performing 20 searches for targets at the leaves of the hierarchy. At random points during each search subjects were asked to locate one of the five pieces of ancilliary information, and reaction time was measured. Subjects who were presented with ancilliary information in constant positions responded 73% faster than those with the variable condition, and their reaction time continued to improve with practice. Subjects in the variable condition reached a stable response time within two or three blocks.

Tennant[66] reports on the NLMenu system as described in Tennant *et al.*[67] One additional insight is that it is more difficult to incorporate mixed input (e.g., graphical or spatially oriented specification) in natural language interfaces than in menu-based interfaces.

Tennant et al.[67] describe a menu system that facilitates construction of natural-language-like queries on a relational database. Users are presented with a single screen of menus which describe attributes and entities in the database, along with different types of grammatical and logical connective. The user constructs a query as a sentence; at each step the system parses the fragment and automatically enables and disables appropriate menus. When the fragment is complete, a query can be issued on the database, or the user may continue to refine the query.

Thompson et al.[68] investigated depth/breadth tradeoff analytically. The expected total search time T is $T=\frac{n\log N}{\log n}t_d+\left(\frac{\log N}{\log n}-1\right)t_w$, where n is the number of options on each menu, N is the number of leaves in the hierarchy, t_w is the time to move from one level to another, and t_d is the time to examine an option and make a decision about it. The above equation is for the case of relative decision-making, where all menu options must be examined before the correct one is identified. If an absolute decision-making process is applied (the correct option is recognized as soon as it is seen), then the equation is modified by multiplying the first term by $\frac{n+1}{2}$ rather than n. Differentiating these equations with respect to n results in an equation for optimum n which is dependent on t_w and t_d but not N. Reasonable values for t_d and t_w indicate that optimum breadth is generally between 3 and 5 items per menu.

Tolle et al.[69] conducted an experiment investigating search time for menu hierarchies of 2-9-9, 2-13-13, 3-9-9, and 3-13-13, presented with and without "previewing" (hierarchies with "previewing" initially display both root level options and the second level options). Subjects took longer to select from hierarchies with 3 root options than with 2, independent of whether previewing was employed. Reaction time was significantly faster for menus with 9 rather than 13 options at the second level. Subjects also seemed to select more quickly at the second level in hierarchies with previewing. The authors suggest that when previewing is employed, subjects make a partial determination of the option to select at the next level.

Tombaugh and McEwen[70] conducted an experiment to compare two types of hierarchical menu organization: alphabetic and categorical (or semantic). A categorical hierarchy consisting of 470 menus and a corresponding alphabetic hierarchy were constructed. 30 subjects participated in a within-subjects test of the two hierarchies that was divided into 4 phases. In the first and second phase subjects solved 4 retrieval problems with one or the other of the hierarchies. In the third phase subjects solved 4 retrieval problems while being directed to use both hierarchies. In the fourth phases, subjects answered 8 retrieval questions and were permitted free choice of hierarchy. Measures were taken of the mean search time, number of keypresses, and number of menus accessed.

No statistically significant differences were found in or between any of the phases except that in the fourth phase subjects tended to switch from categorical hierarchy to alphabetic hierarchy twice as often as the other way around. Subjects accessed on average twice as many menus as the minimum necessary, and made errors on 40% of the search problems. No improvement was noted with either practice or choice of access method.

Tompa[71] defines some fundamental characteristics of menus and menu networks, especially as they pertain to videotex. Two important new concepts are *scope*, which is the set of pages from which the menu labels are selectable, and *range*, or the set of potential target pages. The scope of most menus is simply the menu itself; however, if the system permits the use of global identifiers (e.g., "root"), then the scope of the global identifier menu is the set of all menus. The range of most menus is the set of immediate neighbours; however, if the system permits the use of cross-links, the range of a menu that is the source of a cross-link includes the sink as well.

Scope and range become important when several menu structures exist on the same data, as is the case with private or personal menus. Private menus are necessary in large menu hierarchies to reduce the tedium of traversal to frequently accessed pages. If the scope of menus intersects, a policy must be adopted to resolve the ambiguity of the selection.

Tullis[72] reports experiences in designing a menu system for control of over 700 operating system functions, including choice of logical groupings for the functions and choice of an appropriate hierarchy for the groupings. Choice of logical groupings was done in three phases. First, subjects highly familiar with the system functions sorted them into 271 groups. Second, subjects with some familiarity with the system functions did pairwise estimations of the similarity of the functions. Third, the similarity matrices for the subjects were combined and analyzed to produce a hierarchical clustering. Subjects tended to group functions according to the device they applied to, rather than by function type.

Two different hierarchies were constructed, one with a maximum breadth 15 and maximum depth 4, one with a maximum breadth 45 and maximum depth 2. Time to completion for a predetermined set of tasks was not significantly affected by hierarchy type. There was a marginally significant increase in the number of errors made in the deeper hierarchy.

Uber et al.[73] describe a hierarchical menu system for medical data entry and and information about diagnostic tests. The hierarchy is organized under several dimensions; tests names can be found by specimen, laboratory, and disease category, as well as an overall alphabetical index. Options are selected with a light pen, and presentation of the next menu occurs within 0.5 seconds of a selection. Menus may be single-selection or employ a "forcing technique" (i.e., form-filling).

An algorithm similar to Huffman's is proposed for organizing the hierarchy in order to minimize access time. Access probabilities are assigned to each of the leaves of the hierarchy; then menus of size b are constructed by iteratively choosing the b lowest probability options. If we set the access probability of a menu as the sum of the access probabilities of its options, the process can be continued for menus until a hierarchy is constructed. At this point, options are reassigned to menus at the same level to promote a useful semantic ordering.

A derivation of optimum menu breadth is given, assuming that users search until the correct option is found (i.e., terminating rather than exhaustive search). Given a hierarchy with n leaves, t is the time to select an option, b is the number of options per menu and r is the amount of time to examine one option, then the average time to reach a leaf is $\frac{\frac{bt}{2}+t}{r}\log_b n$. Optimum b is obtained by differentiating to $r=\frac{b}{2}(ln b-1)$. This breadth is considered the minimum optimal since it assumes that the user is unfamiliar with the menu.

Vrins et al.[74] compared search of a videotex system in three conditions: menu, hard copy alphabetical keyword list, and soft copy alphabetical keyword list. 27 subjects each answered 12 of a set of 36 questions, 4 questions per condition. The hard copy index resulted in better performance than the other two conditions, which did not differ from each other.

Whalen and Latrémouille[75] describe an experiment in which 8 subjects solved 16 queries by searching a menu hierarchy; 4 of the queries were for information which did not exist in the hierarchy. Subjects were efficient at locating information which was present, averaging less than one extra menu over the minimal number to solve such queries. Subjects accessed approximately twice as many menus per search when looking for non-existent information. The authors suggest that a poorly designed hierarchy might result in terminated searches even if the required information was present. Since most errors occurred in the top two levels of the hierarchy, it is conjectured that increasing the depth of the hierarchy may not result in a significant increase in the number of errors.

Whalen and Mason[76] describe an experiment in which three types of design defects were deliberately incorporated in a menu hierarchy: miscategorized menus, synonymous menu options, and ambiguous menu options. Four examples of each type of defect were introduced, though subjects were not informed of the defects, they were told that some of the queries would not be solvable. Subjects solved 60% of queries for which a solution was present, with a probability of error of 0.4 for any one menu. Subjects averaged 10.6 menus per query; optimal searches averaged 3.6 menus. The most serious defect was miscategorization; the probability of selecting the correct option in the presence of this defect was less than 0.01. The other two defects reduced the probability of selecting the correct option as 0.45.

Whiteside et al.[77] evaluated subjects' use of four command languages, one menu system, and one iconic language for simple operations on text files. The iconic languages employed a mouse, whereas none of the command or menu languages did. Subjects were classed as new users, transfer users, or experienced users, depending upon their experience with the particular language employed in their session. Though large variances in performance and preference were found, these were not correlated to interface type. No tradeoff was found between ease of use and ease of learning. Both transfer and new users had better performance with menu and command languages than with iconic languages. Whiteside *et al.* suggest that the quality of a particular implementation is of more importance than its type.

Witten et al.[78] investigate algorithms for dynamically adjusting the depth of menu hierarchies, taking into account the frequency of access of the leaves. The authors construct several algorithms which, given a fixed maximum breadth M, attempt to split the set of leaves into hierarchic subsets. The first algorithm splits the directory into M parts by iteratively advancing through the directory and making a split as close as possible to $\frac{1}{M-k}$ of the total probability, where k is the number of splits that have already been made. The second algorithm splits by selecting the $\frac{M}{2}$ point as the first split, and then splitting the subparts until M

splits have been made at the first level. This process is carried out recursively. The third algorithm chooses M buckets which have maximum total entropy.

The three algorithms produce quite different trees for a given Zipfian probability distribution. However, the performance of the trees produced by the algorithms is virtually identical, and is significantly poorer than the entropy-determined lower bound. The authors suggest that improved performance would result from algorithms which did not treat each level of the hierarchy independently.

Yoder et al.[79] describe instrumentation developed to collect usage data for the ZOG system. Information recorded about individual users' sessions includes the amount of time spent viewing each frame (subdivided into various groups) and summary statistics about the session. Information recorded about the system during each session includes the number of times various utilities are used, frames created and deleted, the use of the mouse, selections at menus, subhierarchies accessed, disk response, and network traffic. Intermediate statistics ("snapshots") are kept as well as summary statistics in case a session is abnormally aborted, and also to provide a history of the session's activity. Since the volume of statistics collected is a significant barrier to analysis, care must be taken to record them in browseable fashion. In the ZOG system these statistics are maintained as part of a ZOG hierarchy; hence the normal browsing and searching tools can be employed to investigate and evaluate the statistics. The authors speculate that interpretation of instrumentation data resembles archaeological study of cultural artifacts.

Young and Hull[80] conducted an observational study in order to analyse strategies employed in searching a menu hierarchy. Subjects confronted a variety of problems when interpreting individual menus, such as unfamiliar jargon, need for specialist knowledge, and unforseen types of classification. Significant problems were also found in subjects' ability to correctly identify the structure of the menu's options. The main structural characteristic of a menu is whether or not it represents a partition of the substructure; within this distinction there are several variants, including the use of multiple indices (e.g. alphabetical, geographical, chronological), and collections of several sub-categories into a single intermediate category (e.g., "other"). Subjects had problems when they adopted strategies which assumed a different structure than was actually present in the hierarchy. Because subjects assume some structure, they read menus incompletely, searching only for confirmations of their assumption and options which are plausible within that assumption. The authors suggest that integrating menu layout and content is important in indicating the structure of the options to the user.

6. References

1. Akscyn, Robert M. and Donald L. McCracken, The ZOG Approach to Database Management, CMU-CS-84-128, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Pennsylvania (March 1984).
2. Allen, Robert B., Cognitive Factors in the Use of Menus and Trees: An Experiment, *IEEE Journal on Selected Areas in Communications* **SAC-1**(2) pp. 333-336 (February 1983).
3. Apperley, M.D. and G.E. Field, A Comparative Evaluation of Menu-Based Interactive Human-Computer Dialogue Techniques, *INTERACT '84: First IFIP Conference on Human-Computer Interaction*, pp. 296-301 (September 4-7, 1984).
4. Apperley, M.D. and R. Spence, Hierarchical Dialogue Structures in Interactive Computer Systems, *Software — Practice and Experience* **13** pp. 777-790 John Wiley & Sons, Ltd., (1983).
5. Bewley, William L., Teresa L. Roberts, David Schroit, and William L. Verplank, Human Factors Testing in the Design of Xerox's 8010 "Star" Workstation, *Proceedings of the CHI '83 Conference on Human Factors in Computing Systems*, pp. 72-77 (December 12-15, 1983).
6. Billingsley, Patricia A., Navigation Through Hierarchical Menu Structures: Does it Help to Have a Map?, *Proceedings of the 26th Annual Meeting of the Human Factors Society*, pp. 103-107 (October 25-29, 1982).
7. Brown, James W., Controlling the Complexity of Menu Networks, *Communications of the ACM* **25**(7) pp. 412-418 (July 1982).
8. Card, Stuart K., User Perceptual Mechanisms in the Search of Computer Command Menus, *Proceedings of the CHI '82 Conference on Human Factors in Computer Science*, pp. 190-196 (March 15-17, 1982).
9. Caswell, Deborah L. and Ronald D. Gordon, An Editor for Hierarchically Organized Menus, *IEEE Journal on Selected Areas in Communications* **SAC-1**(2) pp. 343-345 (February 1983).
10. Chan, Patrick P., Learning Considerations in User Interface Design: The Room Model, CS-84-16, Department of Computer Science, University of Waterloo, Waterloo, Ontario (July 1984).
11. Clauer, Calvin Kingsley, An Experimental Evaluation of Hierarchical Decision-Making for Information Retrieval, IBM RJ 1093 (#17928), IBM Research Laboratory, San Jose, California (September 15, 1972).
12. Dray, S.M., W.G. Ogden, and R.E. Vestewig, Measuring Performance with a Menu-Selection Human-Computer Interface, *Proceedings of the 25th Annual Meeting of the Human Factors Society*, pp. 746-748 (October 12-16, 1981).

13. Dumais, Susan T. and Thomas K. Landauer, Using Examples to Describe Categories, *Proceedings of the CHI '83 Conference on Human Factors in Computing Systems,* pp. 112-115 (December 12-15, 1983).

14. Engel, F.L., J.J. Andriessen, and H.J.R. Schmitz, What, Where and Whence: Means for Improving Electronic Data Access, *International Journal of Man-Machine Studies* **18** pp. 145-160 (1983).

15. Feiner, Steven, Sandor Nagy, and Andries van Dam, An Experimental System for Creating and Presenting Interactive Graphical Documents, *ACM Transactions on Graphics* **1**(1) pp. 59-77 (January 1982).

16. Frankhuizen, Jacques L. and Toon G.M. Vrins, Human Factors Studies With Viewdata, *Ninth International Symposium on Human Factors in Telecommunication,* pp. 1-7 (September 29-October 3, 1980).

17. Furnas, George W., Psychological Structure in Information Organization and Retrieval: Arguments for More Considered Approaches and Work in Progress, *Workshop/Symposium on Human Computer Interaction,* (1981).

18. Geller, V.J. and M.E. Lesk, User Interfaces to Information Systems: Choices vs. Commands, *Sixth Annual International ACM SIGIR Conference* **14**(4) pp. 130-135 (Summer 1983).

19. Goodwin, Nancy C., Designing a Multipurpose Menu Driven User Interface to Computer Based Tools, *Proceedings of the Human Factors Society 27th Annual Meeting,* pp. 816-820 (October 10-13, 1983).

20. Greenberg, Saul and Ian H. Witten, Comparison of Menu Displays for Ordered Lists, *Proceedings of the Canadian Information Processing Society Session '84,* pp. 464-469 (May 9-11, 1984).

21. Greenberg, Saul and Ian H. Witten, Adaptive Personalized Interfaces — A Question of Viability, *Behaviour and Information Technology* **4**(1) pp. 31-45 (1985).

22. Hauptmann, Alexander G. and Bert F. Green, A Comparison of Command, Menu-Selection and Natural-Language Computer Programs, *Behaviour and Information Technology* **2**(2) pp. 163-178 (1983).

23. Heffler, Michael J., Description of a Menu Creation and Interpretation System, *Software — Practice and Experience* **12** pp. 269-281 (1982).

24. Hemenway, Kathleen, Psychological Issues in the Use of Icons in Command Menus, *Proceedings of the CHI '82 Conference on Human Factors in Computer Science,* pp. 20-23 (March 15-17, 1982).

25. Herot, Christopher F., Spatial Management of Data, *ACM Transactions on Database Systems* **5**(4) pp. 493-514 (December 1980).

26. Isa, Barbara S. and William C. Ogden, Navigation Issues Related to Menu-Based Software Products, unpublished technical memorandum, IBM Santa Teresa Laboratory, San Jose, California (1986).

27. Kastner, J. and H. Widdel, Graphical Support for Dialogue Transparency, *INTERACT '84: First IFIP Conference on Human-Computer Interaction* **1** pp. 302-306 (September 4-7, 1984).

28. Kidd, A.L., Problems in Man-Machine Dialogue Design, *Proceedings of the Sixth International Conference on Computer Communications,* pp. 531-536 (September 7-10, 1982).

29. Kiger, John I., The Depth/Breadth Trade-Off in the Design of Menu-Driven User Interfaces, *International Journal of Man-Machine Studies* **20** pp. 201-213 (1984).

30. Koved, Larry and Ben Shneiderman, Embedded Menus: Selecting Items in Context, *Communications of the ACM* **29**(4) pp. 312-318 (April 1986).

31. Landauer, Thomas K. and D.W. Nachbar, Selection From Alphabetic and Numeric Menu Trees Using a Touch Screen: Depth, Breadth, and Width, *Proceedings of the CHI '85 Conference on Human Factors in Computing Systems,* pp. 73-78 (April 14-18, 1985).

32. Latrémouille, Susane and Eric Lee, The Design of Videotex Tree Indexes: The Use of Descriptors and the Enhancement of Single Index Pages, pp. 65-112 in *The Design of Videotex Tree Indexes,* Behavioural Research and Evaluation, Department of Communications, Government of Canada, Ottawa, Ontario (May 1981).

33. Leclerc, Yvan, Steven W. Zucker, and Denis Leclerc, A Browsing Approach to Documentation, *Computer,* pp. 46-49 IEEE Computer Society, (June 1982).

34. Lee, Eric and James MacGregor, Minimizing User Search Time in Menu Retrieval Systems, *Human Factors* **27**(2) pp. 157-162 (April 1985).

35. Lee, Eric, Thom Whalen, Scott McEwen, and Sue Latrémouille, Optimizing the Design of Menu Pages for Information Retrieval, *Ergonomics* **27**(10) pp. 1051-1069 (1984).

36. Lee, E.S., T.E. Whalen, S. McEwen, and S. Latrémouille, Human Factors in Videotex Information Retrieval, *International Zurich Seminar on Digital Communications — Man-Machine Interaction,* pp. H1.1-H1.8 IEEE Computer Society, (March 9-11, 1982).

37. Liebelt, Linda S., James E. McDonald, Jim D. Stone, and John Karat, The Effect of Organization on Learning Menu Access, *Proceedings of the Human Factors Society 26th Annual Meeting,* pp. 546-550 (October 25-29, 1982).

38. McCracken, Donald L. and Robert M. Akscyn, Experience with the ZOG Human-Computer Interface System, *International Journal of Man-Machine Studies* **21** pp. 293-310 (1984).

39. McDonald, James E., Jim D. Stone, Linda S. Leibelt, and John Karat, Evaluating a Method for Structuring The User-System Interface, *Proceedings of the 26th Annual Meeting of the Human Factors Society,* pp. 551-555 (October 25-29, 1982).

40. McDonald, James E., Jim D. Stone, and Linda S. Liebelt, Searching for Items in Menus: The Effects of Organization and Type of Target, *Proceedings of the 27th Annual Meeting of the Human Factors Society,* pp. 834-837 (October 10-14, 1983).

41. McEwen, Scott A., An Investigation of User Search Performance on a Telidon Information Retrieval System, pp. 35-64 in *The Design of Videotex Tree Indexes*, Behavioural Research and Evaluation, Department of Communications, Government of Canada, Ottawa, Ontario (May 1981).

42. Miller, Dwight P., The Depth/Breadth Tradeoff in Hierarchical Computer Menus, *Proceedings of the 25th Annual Meetin of the Human Factors Society,* pp. 296-300 (October 12-16, 1981).

43. Norman, Donald A., Design Principles for Human-Computer Interfaces, *Proceedings of the CHI '83 Conference on Human Factors in Computing Systems,* pp. 1-10 (December 12-15, 1983).

44. Norman, Kent L., Jeffrey P. Schwartz, and Ben Shneiderman, Memory For Menus: Effects of Study Mode, CAR-TR-69/CS-TR-1412, Human/Computer Interaction Laboratory, Center for Automation Research, University of Maryland, College Park, Maryland (June 1984).

45. Novatchev, Dimiter and Yuly Gabrovsky, An Attempt to Ease and Simplify the Development of Interactive Software: The Menu Support System of Interpro, *INTERACT '84: First IFIP Conference on Human-Computer Interaction* **1** pp. 49-52 (September 4-7, 1984).

46. Parkinson, Stanley R., Norwood Sisson, and Kathleen Snowberry, Organization of Broad Computer Menu Displays, *International Journal of Man-Machine Studies* **23** pp. 689-697 (1985).

47. Parton, Diana, Keith Huffman, Patty Pridgen, Kent Norman, and Ben Shneiderman, Learning A Menu Selection Tree: Training Methods Compared, CAR-TR-66/CS-TR-1409, Human/Computer Interaction Laboratory, Center for Automation Research, University of Maryland, College Park, Maryland (June 1984).

48. Penna, D.E., The Use of a Colour Graphics Display and Touch Screen to Help Naive Users Understand and Control a Multi-Function Computer System, *INTERACT '84: First IFIP Conference on Human-Computer Interaction* **2** pp. 66-70 (September 4-7, 1984).

49. Perlman, Gary, Making the Right Choices With Menus, *INTERACT '84: First IFIP Conference on Human-Computer Interaction* **1** pp. 291-295 (September 4-7, 1984).

50. Poggio, Margaret E., Developing a Portable Window/Menu Interface Using Object-Oriented Programming Tools, UCRL-94376, Lawrence Livermore National Laboratory, Livermore, California (April 1, 1986).

51. Pollard, D. and P. James, Aspects of Indexing on Prestel, *Ninth International Symposium on Human Factors in Telecommunication,* pp. 9-15 (September 29-October 3, 1980).

52. Raymond, Darrell R., Personal Data Structuring in Videotex, CS-84-7, Department of Computer Science, University of Waterloo, Waterloo, Ontario (February 1984).

53. Raymond, Darrell R., Alberto J. Cañas, Frank Wm. Tompa, and Frank R. Safayeni, Measuring the Effectiveness of Personal Database Structures, *submitted to International Journal of Man-Machine Studies*, (1986).

54. Robertson, G., D. McCracken, and A. Newell, The ZOG Approach to Man-Machine Communication, *International Journal of Man-Machine Studies* **14** pp. 461-488 (1981).

55. Savage, Ricky E., James K. Habinek, and Thomas W. Barnhart, The Design, Simulation, and Evaluation of a Menu Driven User Interface, *Proceedings of the CHI '82 Conference on Human Factors in Computer Science*, pp. 36-40 (March 15-17, 1982).

56. Schabas, Ann H. and Frank Wm. Tompa, Trees and Forests: User Reactions to Two Page Access Structures, *Videotex '83*, Online Conferences, (June 27-29, 1983).

57. Schultz, E. Eugene and Patrick S. Curran, Menu Structure and Arrangement of Menu Selections: Independent or Interactive Effects?, unpublished technical report, Jet Propulsion Laboratory, California Institute of Technology (1986).

58. Sisson, Norwood, Stanley Parkinson, and Kathleen Snowberry, Design Methodology for Menu Structures, *IEEE 2nd Annual Phoenix Conference on Computers and Communications*, pp. 557-560 (March 14-16, 1983).

59. Snowberry, Kathleen, Stanley Parkinson, and Norwood Sisson, Effects of Help Fields on Hierarchical Menu Search, *IEEE 2nd Annual Phoenix Conference on Computers and Communications*, pp. 552-556 (March 14-16, 1983).

60. Snowberry, Kathleen, Stanley Parkinson, and Norwood Sisson, Effects of Help Fields on Navigating Through Hierarchical Menu Structures, *International Journal of Man-Machine Studies* **22** pp. 479-491 (1985).

61. Snowberry, Kathleen, Stanley R. Parkinson, and Norwood Sisson, Computer Display Menus, *Ergonomics* **26**(7) pp. 699-712 (1983).

62. Somberg, Benjamin L., *A Comparison of Rule-Based and Positionally Constant Arrangements of Computer Menu Items*, GTE Laboratories, Inc., Waltham, Massachusetts (1986).

63. Somberg, Benjamin L., George J. Boggs, and Maria C. Picardi, Search and Decision Processes in Human Interaction With Menu-Driven Systems, *The Human Factors Society 26th Annual Meeting*, (October 1982).

64. Somberg, Benjamin L. and Maria C. Picardi, Locus of the Information Familiarity Effect in the Search of Computer Menus, *Proceedings of the 27th Annual Meeting of the Human Factors Society*, pp. 826-

830 (October 10-13, 1983).

65. Teitelbaum, Richard C. and Richard E. Granda, The Effects of Positional Constancy on Searching Menus for Information, *Proceedings of CHI '83 Conference on Human Factors in Computing Systems,* pp. 150-153 (December 12-15, 1983).

66. Tennant, Harry R., Menu-Based Natural Language Understanding, *Proceedings of the 1984 AFIPS National Computer Conference,* pp. 629-635 (July 9-12, 1984).

67. Tennant, Harry R., Kenneth M. Ross, and Craig W. Thompson, Usable Natural Language Interfaces Through Menu-Based Natural Language Understanding, *Proceedings of the CHI '83 Conference on Human Factors in Computing Systems,* pp. 154-160 (December 12-15, 1983).

68. Thompson, David A., Lawrence A. Bennigson, and David Whitman, A Proposed Structure for Displayed Information to Minimize Search Time Through a Data Base, pp. 452-455 in *Introduction to Information Science,* ed. Tefko Saracevic, R.R. Bowker & Co. (1970).

69. Tolle, John E., Michael J. Prasse, and Ralph Gott, *Effects of Variation in Menu Length and Number of Windows on User Search Time,* Online Computer Library Center (1986).

70. Tombaugh, Jo W. and Scott A. McEwen, Comparison of Two Information Retrieval Methods on Videotex: Tree-Structure Versus Alphabetical Directory, *Proceedings of the CHI '82 Conference on Human Factors in Computer Science,* pp. 106-110 (March 15-17, 1982).

71. Tompa, Frank Wm., Retrieving Data Through Telidon, *Proceedings of CIPS '82,* (1982).

72. Tullis, Thomas S., Designing a Menu-Based Interface to an Operating System, *Proceedings of the CHI '85 Conference on Human Factors in Computing Systems,* pp. 79-84 (April 14-18, 1985).

73. Uber, Gordon T., Paul E. Williams, Bradner L. Hisey, and Robert G. Siekert, The Organization and Formatting of Hierarchical Displays for the On-Line Input of Data, *Proceedings of the 1968 AFIPS Fall Joint Computer Conference,* pp. 219-226 (December 9-11, 1968).

74. Vrins, A.G.M., R.H. Van Velthoven, and J.L. Frankhuizen, Search Method Evaluation in the Dutch Videotex System, *Displays,* pp. 101-105 (April 1982).

75. Whalen, Thomas and Susane Latrémouille, The Effectiveness of a Tree-Structured Index When the Existence of Information is Uncertain, pp. 3-14 in *The Design of Videotex Tree Indexes,* Behavioural Research and Evaluation, Department of Communications, Government of Canada, Ottawa, Ontario (May 1981).

76. Whalen, Thomas and Candy Mason, The Use of a Tree-Structured Index Which Contains Three Types of Design Defects, pp. 15-34 in *The Design of Videotex Tree Indexes,* Behavioural Research and Evaluation, Department of Communications, Government of Canada,

Ottawa, Ontario (May 1981).

77. Whiteside, John, Sandra Jones, Paula S. Levy, and Dennis Wixon, User Performance with Command, Menu, and Iconic Interfaces, *Proceedings of the CHI '85 Conference on Human Factors in Computing Systems*, pp. 185-191 (April 14-18, 1985).
78. Witten, Ian H., John G. Cleary, and Saul Greenberg, On Frequency-Based Menu-Splitting Algorithms, *International Journal of Man-Machine Studies* **21** pp. 135-148 (1984).
79. Yoder, Elise, Donald McCracken, and Robert Akscyn, Instrumenting a Human-Computer Interface for Development and Evaluation, *INTERACT '84: First IFIP Conference on Human-Computer Interaction* **2** pp. 309-314 (September 4-7, 1984).
80. Young, R.M. and A. Hull, Cognitive Aspects of the Selection of Viewdata Options by Casual Users, *Proceedings of the Sixth International Conference on Computer Communications*, pp. 571-576 (September 7-10, 1982).