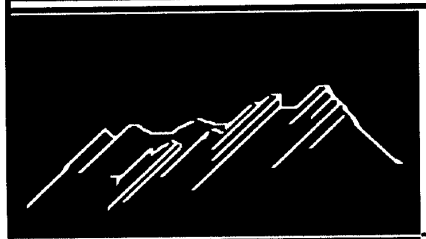
INSTITUTE OF COGNITIVE SCIENCE



Technical Report

University of Colorado, Boulder

Exploration and Experienced Performance with Display-Based Systems

Marita Franzke

Institute of Cognitive Science
University of Colorado
Boulder, Colorado 80309-0344

Technical Report #94-05

EXPLORATION AND EXPERIENCED PERFORMANCE WITH DISPLAY-BASED SYSTEMS

by

MARITA FRANZKE
Vordiplom, Ruhr-Universität Bochum, 1986
M.A., University of Colorado, 1989

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Psychology

1994

Abstract

The research reported in this dissertation investigates how interface-literate users explore and use new display-based computer applications. It aims at identifying characteristics of present interface designs that help or hamper these activities. Task-oriented exploration of a new application is conceptualized as search through a problem space of legal interactions. Three interface characteristics were derived on the background of this conceptualization, and their combined effects on task performance were investigated.

In the experiment, display-based interaction of experienced users was investigated for the case of graphing tasks with commercially available computer applications. Performance during task-oriented exploration and two further trials at two different delays was analyzed globally and on a very low level of detail.

The overall results show that subjects were able to complete the exploration phase when help was provided. Knowledge acquired in the exploration phase could be used successfully later on. Specifically, this knowledge helped to prevent search behavior in a second trial. Forgetting effects were relatively small even after one week. Mechanisms for learning and possible causes for the rare occurrence of forgetting effects are discussed.

The detailed analyses show evidence for the use of a search heuristic, called label-following (Engelbeck, 1986; Polson, 1988). Search through a problem space of legal interactions can be constrained by semantic matches between interface labels and the users' goal representation. Two design characteristics, the use of unlabeled direct-manipulation interactions, and the use of 'poor' labels, are shown to impair the use of the label-following heuristic, and therefore hinder discovery of the correct actions during exploration. A third design characteristic, the provision of large sets of possible operators, is shown to complicate search in these situations. These characteristics are shown to impair performance during exploration as well as after long delays.

Possibilities for application of the results in guidelines for evaluation of display-based interfaces are discussed. Finally, the chosen empirical methodology is evaluated on the dimensions of generalizability, applicability, and usability.

ACKNOWLEDGMENTS

"With a little help from my friends."

I would like to use this opportunity to express my heartfelt gratitude to Peter Polson and Walter Kintsch for their academic advice and personal support during what seems to be an eternity of graduate studies. Thank you for getting me in and out of the program and for letting me learn from your insights on human cognition in the process. I have also benefited greatly from discussions and support of my other committee members: Clayton Lewis, Reid Hastie, and Gerhard Fischer. Thank you for making my dissertation an experience I will be glad to remember.

Terry Roberts and Catherine Marshall, then of U S WEST Advanced Technologies, acted as role models, just in the right moment. Thank you for pioneering and for peaking my interest in human-computer interaction.

I would not have been able to reach the finish line without the support of great friends and colleagues: Evelyn Ferstl, John Rieman, Catherine Ashworth, Adrienne Lee, Peter Foltz and Paula Messamer: Thanks for sharing the experience. Finally, thanks to Christopher Tuerpe for helping me to get away from it all whenever necessary.

Last but not least, I would like to express my gratitude to my parents, Friedhelm and Marianne Franzke for their continued patient support and for helping me to discover the value of education early on.

Abby Harrison and Troy Davig assisted in running the experiment. This research was supported by NSF grant # IRI-9116640.

CONTENTS

APPENDIX

TABLES

FIGURES

CHAPTER I

INTRODUCTION

Display-Based Interaction

When graphical user interfaces (GUI's) were first introduced as a new alternative for data processing technology, they promised a great change in the way in which users could learn and use new applications. Researchers in Human Computer Interaction (HCI), as well as business reviewers, anticipated a great success of these interfaces, (e.g., Shneiderman, 1982; Merkin, 1983). Until then, common user-interfaces had two very limiting constraints: For one, they only allowed for a very limited display of the entered information to the user. For example, a common class of editors, line editors, did not dynamically update displayed text as it was edited. If users wanted to modify a portion of text that had been entered before, they had to rely on their memory of what they had entered and where in relation to the current line that information would be found. Second, all commands, such as the commands to move around in the text, to insert a line, etc., had to be memorized, and entered in a special command language. Clearly, both of these constraints put a heavy load on the users' memory, and made acquisition of skill with such systems quite cumbersome (i.e., Singley & Anderson, 1985; Polson, 1988). Consequently, command-based systems were usually learned in training seminars, or through heavy use of accompanying manuals and training materials. Playful exploration of these systems was not possible, at least not for the average, non-technical user.

Graphical user interfaces promised to lift these constraints on both counts. Instead of putting the burden of knowing the content and the structure of the entered information onto the mind of the users, newer systems were able to display larger amounts of information on the video display unit (VDU), and furthermore they could display it in the same way as it would be printed on a piece of paper (what-you-see-is-what-you-get, WYSIWYG). This by itself made word processors, for example, easier to learn and easier to use (Roberts & Moran, 1982). Secondly, commands that had to be memorized and retrieved in command-based systems could now be shown on the display in form of menus, or graphical objects that could be manipulated directly with a mouse-driven pointer. This lifted the second burden on memory, because the display reminded users of command names, which could be accessed by a general set of procedures or by direct manipulation.

Some of the expectations that the introduction of these new systems, which we will call *display-based systems*, raised are summarized quite well in Shneiderman (1983, p. 57, italics added):

> "The enthusiastic users' reports are filled with positive feelings regarding
> - mastery of the system,
> - competence in the performance of their task,
> - *ease in learning the system originally and in assimilating advanced features,*
> - *confidence in their capacity to retain mastery over time,*
> - enjoyment in using the system,
> - eagerness to show it off to novices,
> - *desire to explore more powerful aspects of the system.*"

Three of these topics, learning through exploration, ease of learning, and retention over time, in the context of display-based systems are the central themes of this dissertation.

Motivation

Relevance to Human-Computer Interaction

Relating Design Characteristics to Performance. With the success of the Apple Macintosh and Windows for the PC, display-based systems as described above have become nearly ubiquitous. More and more systems offer a standard combination of windows, menus, icons, and mouse input. However, even though this success may demonstrate a clear overall usability advantage of display-based systems over command-based ones, there are few empirical studies that relate specific aspects of display-based design to task performance. Previous studies have addressed either comparisons between overall performance on display-based and command based systems, or comparisons between particular features of menus and icons in the context of simple search tasks (see literature review below). There are simply no detailed studies of user performance with full-scale display-based applications in service of complex task goals. In this dissertation we aim at filling this gap by relating design parameters embodied in commercially available and widely used graphing applications to task performance at a low level of detail. As a result, we hope to identify dimensions on which to evaluate existing systems, and which could guide the design of new display-based applications.

Defining Needs of Discretionary Computer Users. System developers are now designing applications for an new generation of "discretionary" users. A growing proportion of users of new software are now computer-literate, and will have used a

variety of other computer applications before. Unfortunately, this class of users has received only little attention in the current HCI literature (e.g., Rosson, 1986, and Santhanam & Wiedenbeck, 1993). Santhanam and Wiedenbeck define this class of discretionary users as intermediate, casual users of software. They are office workers who use computers intermittently, and sometimes at long time delays. The computer is not the focus of their work, but simply a tool to achieve other goals. Santhanam and Wiedenbeck found, for example, that discretionary users of software are more likely to attempt learning new functionality by exploration than novice users. Discretionary users of systems will also use systems only sporadically, sometimes at long time delays. Design for present and future contexts of use should therefore support both exploratory learning and retention of acquired interface knowledge. In essence, well designed systems would support learning and retention "on demand" (Fischer, 1991). The current study focuses on these problems by investigating exploration and retention performance of subjects who were selected to represent the population of interface-literate, discretionary users of office systems. We hoped to relate particular design characteristics to task-oriented exploration, and to experienced performance, both at short and long time delays.

In summary, we believe that the current study is relevant to the area of HCI because it will help to identify design characteristics of fully functional display-based systems that support or hinder discretionary use of software in service of complex task goals. Detailed empirical observations to support claims about usability in this situation are not available today.

Relevance to Cognitive Psychology

Simon's Ant. In his book *The Sciences of the Artificial*, Simon (1981) compares human performance in service of any type of goal to the surprisingly complex path of an ant on the beach. In the context of that analogy, Simon (1981, p. 65, italics in the original) made the claim that:

> *"A man, viewed as a behaving system, is quite simple. The apparent complexity of his behavior over time is largely a reflection of the complexity of the environment in which he finds himself."*

In the development of this argument, Simon claims further that most of human behavior is a learned adaptation to artifacts in the environment. Studying human behavior then becomes, in large part, a process of studying the artifacts and determining how particular adaptive strategies interact with the structure of the artifacts to mediate the goals of the individual. Simon points at two strategies that make behavior in complex

environments more efficient, namely the use of *search heuristics* and the use of *memory*. If a person tries to reach a goal in a new environment (for example, completing a task with a new type of software for a computer novice), he or she may not know how to search the environment efficiently to achieve the goal. If search is over a known type of environment (a new application with a known type of interface), people can use general search heuristics acquired to satisfy the constraints of this particular type of environment. One of the goals of this research is to identify such heuristics in the context of using display-based systems.

Furthermore, if a similar goal needs to be accomplished in a known environment (accomplishment of repeated tasks in an application used for a similar task before), then the buildup of specific memory traces containing previous interactions should help in accomplishing the goals most efficiently. A second goal of this study then is to investigate how quickly and efficiently such memory traces can be built and used.

In summary, we consider the current study to be an investigation of adaptive search heuristics and memories for the special case of a human information processing system with a dynamic "external memory".

External Memories. Early in the history of cognitive science, Newell and Simon (1972) appealed for the investigation of problem solving behavior of a human information processing system with access not only to internal memories (STM and LTM), but also to external memories (EM). The idea behind this appeal was that humans use their environment as repository for information that they need to access during problem solving, to reduce the burden on their limited capacity short time memory. Some widely used examples are the use of scratch paper and sketches to help as reminders and as inference support during geometry problem solving (e.g., Larkin & Simon, 1987) or the use of various states of kitchen equipment to support the process of brewing coffee (Larkin, 1988). Interactions with such external memories are difficult to observe, largely because externally encoded information is spread out over large, unconstrained environments (offices, living quarters, or libraries).

Display-based systems offer an exception to this rule. The visible display offers a history of previous interactions (e.g., the document under creation). Available functions are displayed directly, or can be uncovered with simple interactions ("menu-flicking"), and display some backup information when under closer investigation (cancel buttons, highlighting of root menus, etc.). However, even though this type of environment exhibits some of the characteristics attributed to external memories, its immediate extension does not usually reach beyond a 19 X 19 inch area, and is therefore more accessible to scrutinized observation.

The domain of interactions with display-based systems, then, offers a good opportunity to collect information about adaptive strategies (search heuristics, and the use of memory) in the context of an external memory that provides information about some history of the problem solving trace, rich information about further action options, and may be used to cue memory traces of previous interaction sequences.

Overview of the Research

In the experiment we investigated performance with four widely used, commercially available systems in the context of three realistically complex graphing tasks (graph creation and modification). The subjects were selected to represent the population of discretionary users, and were familiar with the general class of interfaces investigated (display-based, Macintosh-style). Each subject performed three similar tasks. The first task counted as exploration trial, whereas later trials represented situations of 'experienced use'. Later trials were performed at either a short (several minutes) or a long (one week) delay. Performance was analyzed at a very low level of detail, and related to several design parameters. These parameters were derived from an analysis of display-based interactions as a search through a problem space.

The Display as Problem Space. If the use of a new application is defined as search through a problem space (Newell & Simon, 1972), how can this particular space be characterized, and what search heuristics are likely to be used? Consider the case of the user of a graphing application who is faced with the task of changing the symbols in a given graph into a different format (say, from white to black). The goal state is to have black plot symbols displayed on the screen, and the start state is the current graph with white symbols. The application has a typical set of about ten menu-items, a tool-bar with twenty tools, controls for the window (scroll bars, etc.), and the objects in the graph are directly manipulable. Without a reasonable heuristic to constrain the search, the user would have to try serially each one of these operators. Selection of any one operator may uncover another set of about ten possible choices, and so on. It is clear that search in this environment would be quite complex if no adaptive search heuristic could be used to guide it.

Label-Following as Heuristic. In the context of menu-based systems (Engelbeck, 1986), of "Walk-Up-and-Use" systems (Polson, et al. 1992), and of Macintosh-type systems (Kitajima & Polson, 1992), it has been suggested before that computer users may make use of a selection heuristic, called *label-following*, that guides search through the large space of possible operator selections. The idea is that users base their choices

primarily on semantic similarity matches between their representation of the task goal, and the labels and objects presented to them. If this strategy indeed drives interactions with external displays, then the quality of a label used in an interface should have an observable effect on user performance during first use of a system. For this reason the investigated interfaces were coded for the quality of the semantic match that they provided in the service of a particular task. We expected that labels which provided poor matches would be associated with longer performance times during first use (exploration).

Size of the Problem Space. At each step during the interaction, and in different interfaces, users have varying numbers of possible operators to consider. After the popping-up of a dialog-box, for example, there may only be three choices available, but during selections with the menus, many possible choices are accessible. One can generally expect that search times increase with the number of alternative operators that need to be evaluated. In our interactions, we coded each action for the number of operators available at the time, and expected performance times to increase with the number of these choices.

Type of operator. Many types of basic operators are used in the context of display-based systems. We call these operators basic-level actions. They define classes of operations, by tying basic physical actions (like single-clicks, mouse button-presses, or mouse button-releases) to basic classes of display-objects (menu-bar items, iconic tools, buttons in dialog boxes, etc.; see Kitajima & Polson, 1992). In this way, we coded interactions as 'select-item-in-menu-bar', or 'click-on-button', or 'double-click on graph object', for example. We expected that some of these operators would be more likely to be selected than others, but we did not formulate a clear hypothesis about which ones. Therefore we expected general performance differences between the classes of operators.

Experienced Use of Display-Based Systems. We also see this experiment as a test bed to see how quickly external memories can be used to encode traces of previous interactions. In our experiment, we investigate performance on three similar graphing tasks. If the use of the display serves as a memory aid to recover previous problem solving sequences, then use during the second and third task should be greatly improved. In particular, the degree to which memory traces guide performance in later trials should be expressed in the strength of the effects of the parameters that effect the difficulty of search during explorations. For example, if the number of operators to be searched has no effect on performance during the second trial, we can conclude that a memory-trace of the previous episode could be used to avoid renewed search of the display environment. Additionally, performance at the short and long delay intervals can be compared in a similar matter to allow estimation of decay of memory traces built in this context.

Summary. We see interactions with display-based systems as a special case of human problem solving in an environment marked by the availability of dynamic external memories of previous choices and available operators. This environment offers search criteria in form of verbal labels. Therefore a search heuristic that uses a similarity match between those labels and a representation of the goal is proposed that may guide the search process. If this search heuristic is indeed used, then the difficulty of the search should vary with the quality of the provided labels, and with the general number of options to evaluate. Exploration performance should therefore suffer with provision of poor labels, and many alternative choices. We also expect that different basic choice probabilities may be associated with certain classes of operators. Observing performance across two trials and two different delay intervals will inform us about the efficiency with which the external memory can be used to constrain behavior after some experience, and will inform us about the basic decay characteristics associated with the use of such memories.

Overview of the Thesis

In the second chapter, our simplified assumptions will be related to more sophisticated models of display-based interaction. The third chapter summarizes and discusses relevant empirical work. Chapter five explains the method of the current experiment in detail. Chapter five presents the results. Chapter six summarizes the current findings and discusses their theoretical and practical relevance. Finally, the chosen methodology will be evaluated and its broader use for research in HCI will be discussed.

CHAPTER II

RELATED THEORETICAL WORK

The empirical work presented here is based on a few simple theoretical assumptions. These are assumptions about what types of knowledge are needed for successful display-based interaction (base-level action knowledge, general background knowledge, and application-goal knowledge), about a preferred mode of learning (exploration), and about the search mechanisms underlying exploration. Furthermore, it is assumed that in display-based interaction, exploration and experienced performance are governed by the same principles, with different sources of knowledge accessible. These basic assumptions are simplifications of ideas detailed in other analyses and computational models (Howes & Payne, 1990; Young, Howes, & Whittington, 1990; Kitajima & Polson, 1992, 1994a, 1994b, Howes & Young, 1993; Howes, 1993, 1994; Carroll, 1990; Newell & Simon, 1972; Rieman, 1994). In this chapter, the relevant assumptions are spelled out, and supported by arguments found in previous work. Finally, it will be shown how these conjectures lead into the empirical work described in later chapters.

Knowledge Analysis

It is assumed that three broad classes of knowledge are needed for successful performance with display-based systems. These can generally be described as (1) procedural knowledge, which encodes basic-level action knowledge, (2) declarative knowledge, including general background knowledge of interface concepts and basic semantic knowledge, and (3) application-goal knowledge, which consists of specific episodic memory traces capturing previous interactions with a particular applications.

Base-Level Action Knowledge

Base-level action knowledge allows subjects to move around in the interface freely. These are basic productions, or plan-elements (Kintsch & Mannes, 1991; Kitajima & Polson, 1992), that specify actions which can be performed on display-objects. For example, one could imagine a generic production that recognizes a dialog-box button and specifies the relevant action, namely single-clicking. Another example would be a production that recognizes a menu-bar item and specifies clicking-and-holding as the

appropriate action. Such basic action knowledge is included in almost every theoretical account of display-based interaction (Young, et al., 1990, *screen interpretation and affordances*; Howes & Young, 1993, *primitive actions*; Kitajima & Polson, 1994b, *generic plan elements*; and Howes & Payne, 1990, *actions*.). The specific content and definition of these procedural elements differs somewhat from proposal to proposal. Here we will assume the following properties:

First, action elements are generic, that is, they apply to any object within the specified object class. This means that a procedure for button-pressing applies to any button with any label in any interface, as long as it is recognized as an instance of this class, and fulfills certain conditions (is not grayed-out, for example). In Kitajima and Polson's (1994b) model these generic plan elements are stored in LTM, and are applied to different display-objects during the execution of a task. In Howes and Young's (1993) TAL, actions are stored as generic operators, for example for pressing and clicking. During learning of a new task, their model learns to associate a specific display object, say the OK-button, with a basic operator, namely press. Even though this new operator may have been learned in the context of a specific display object or class of display objects, this elementary rule can be reused in service of any object within the specified class (synthetic generalization, Howes & Young, 1993, p. 27).

Second, an extension of the above principle, action elements are nonspecific with regard to overall task goals. This means that an action class may serve any task goal, and goal information does not determine directly what type of action will be chosen. In both approaches described above, stored action elements (or operators) do not specify the particular goal or task they serve, and can therefore be used in any task context (Kitajima & Polson, 1994b; Howes & Young, 1993).

Third, our empirical investigation assumes that basic actions are already available to the user. To some degree, this simplifying assumption is mirrored in the computational models. Howes and Young (1993, p. 14), for example, argue that "... things in the world offer affordances for action, e.g. buttons afford being pushed, sliders afford being slid, and knobs afford being turned." In their model an advisor helps the system to link basic actions, like pressing, to certain object classes. Kitajima and Polson's model does not model learning, and therefore does not address this problem.

Empirical observations suggest that initial learning of these basic-level actions may be quite difficult (e.g. Carroll & Mazur, 1986; Whiteside, et al., 1985). Carroll and Mazur observed six novice users during initial exploratory acquisition of skill with the Lisa, an early version of the Apple Macintosh. These users were provided with full system documentation and training materials; however, use of these materials was at their

discretion. Observer interactions and help were held at an absolute minimum. Observations from this study suggest that object classes and basic operations are initially difficult to distinguish, and therefore it is difficult to acquire legal object-action pairs. Carroll and Mazur found, for example, that subjects had trouble distinguishing between the concepts 'menu' and 'menu bar', and actions such as 'clicking' and 'double-clicking'. Such initial difficulties have been documented for acquisition of other procedural skills, where the acquisition of new procedures is critically dependent on the differentiation of a large new concept base in a previously unknown concept domain (e.g. Pennington, Nicolich, & Rahm, 1993, and for a discussion of this phenomenon see Franzke, in press). Detailed empirical work analyzing this stage of skill acquisition with display-based systems is missing.

However, the current work is constrained to the investigation of interface-literate users, and we assume that these users have a stable, and transferable basis of basic-level action knowledge. For our experimental work, we will therefore assume, that our users have a well practiced base of basic procedures allowing them to perform all available operations on the given interface objects. We will assume furthermore that this knowledge can be accessed independent of task and application context.

General Background Knowledge

A second type of knowledge that we assume in our subjects is general declarative background knowledge of the domain. Under this we understand any type of knowledge that would help users to make sense of the application in the context of specific task. Young, et al. (1990) have listed several classes of knowledge that would fall into this category: a general conceptual model about the functions of computers, everyday semantics to interpret labels and symbols in the display, special Macintosh meanings (for example the semantic and functional meanings of general Macintosh commands, such as copy and paste), and knowledge of decomposition (for example, the convention that objects need to be selected before a function can be applied to them).

Payne, Squibb, and Howes (1990) present a theoretical model, called the Yoked State Space (YSS) that implements this type of knowledge. In this account, users of software have access to two types of problem spaces, a device space and a goal space. A minimal device space relates simple goal states to device states. Elaborated device representations include hidden device states, such as buffers, or important concepts, such as the concept of a string. If these elaborations of the device space are linked in relevant ways to concepts in the goal space (such as copy and paste), they may lead to the use of more efficient strategies, and better transfer.

Kitajima and Polson's model (1992, 1994b) also implements these types of background knowledge as propositions stored in LTM. They represent specific knowledge about Macintosh conventions, such as dialog-boxes, icons, interface objects, menus, and the mouse. There are also propositions encoding general knowledge about applications, types of applications such as spreadsheets, and functions such as editing. For further details and examples of these propositions see Kitajima and Polson (1992).

For our study we will simply assume that discretionary users of new applications will have acquired a fair amount of general Macintosh knowledge, as it is requisite for exploratory acquisition of new application-goal knowledge (see below). Similarly, as for the cased of basic action knowledge, we expect that the initial learning of these declarative sources of interface-literacy is by no means trivial. The observations by Carroll and Mazur (1986) suggest, for example, that new users of Macintosh systems have great difficulties learning about the metaphors used in the desktop and their functions. We conjecture that such conceptual elaborations are acquired as predecessors and as byproducts of learning basic-action knowledge (e.g., Franzke, in press).

Application-Goal Knowledge

Under application-goal knowledge we understand a collection of episodic memory traces that connect task descriptions (such as: 'I want to create a graph') to interface objects that were relevant in previous similar task contexts. Rather than containing information about how to act, it contains semantic information describing previously experienced episodes relating task information to interface objects. For example, during initial exploration of a new graphing system, the application of a display-object with the label **File** may have somehow lead to the creation of a graph. Seeing this display-object during a second task may remind the user of an elaboration created during the previous use ("creating a new graph is like creating a new file"). The user may also remember general locational information, such that the first action was initiated in the upper-left hand corner of the display. Or, the user might recall that "something else" needed to be done before using the menus.

Kitajima and Polson's model includes elaborations like this in its general knowledge base as specific knowledge about particular domains. However, here this knowledge has been generalized from singular episodes and is encoded in propositions, such as 'Y-axis is a Display-Object'. This conjecture fits well with their conceptualization of their model as a model of an expert, rather than intermediary user. (Kitajima & Polson, 1994b, p. 2).

Young, et al. (1990) name two classes of knowledge that fit our description, specific item knowledge, which simply points to the importance of a label name or object, given a

specific goal and application, and locational knowledge pointing to a certain location, given a task and an application. The computational model by Howes and Young (1993) assumes that locational knowledge is the key information driving expert performance.

The sections below describe how basic-action knowledge and general background knowledge could be used during exploration to acquire application-goal knowledge, and how the use of application-goal knowledge in return may improve performance in later trials.

Learning by Exploration

The Importance of Exploratory Learning

Various empirical studies have demonstrated that users of new computer software prefer learning by doing, or learning by task-oriented exploration over the reading of manuals and training materials. Carroll and Mazur (1986, pp. 36-37) for example, observed users who had classified themselves as cautious manual-readers trying things freely only a few minutes into a learning session. Various other studies have confirmed this preference (e.g. Carroll, 1990, Carroll & Rosson, 1987).

In a recent dissertation, Rieman (1993, 1994) provided further evidence for users' preference for trying things on their own, rather than following training manuals or visiting classes. In his diary-study of fourteen interactive computer users, new discoveries of technical functionality was aided by 'trying things out' in 37 percent of the cases followed by reading the manual (26 percent of the cases). In many cases, users reported that they tried things out first and only consulted other sources after they had been unsuccessful with that approach. In-depth interviews confirmed these learning strategies. Very few users reported engaging in task-free exploration, but all of them were willing and able to explore in the context of a task.

There is also evidence that task-oriented exploration leads to better learning results than rigid training schedules, or task-free exploration. Experiments by Charney and Reder (1986) and Charney, Reder, and Kusbit (1990) showed that for new users of spreadsheets a form of exploration, in which tasks involving the use of new commands were experimenter provided, lead to longest learning times but best training results. This type of learning was more successful than exploration of the interface without clear set goals, or problem-solving training in which guidance to the correct solution was provided.

Unfortunately, exploratory use of systems may lead novice users into committing mistakes that are difficult to correct. Often, users do not notice the results of an operation

immediately and are not able to learn the correct causal relationships between actions and their consequences. In Carroll and Mazur's (1986, p.38) study, one user became so frustrated after a long sequence of actions whose results he could not understand, because he could not possibly map how the actions had been interpreted in (hidden) system states, that he quit the experiment. Carroll and his group developed the 'training wheels approach' as an answer to this problem (Carroll & Carrithers, 1984; Carroll, 1990). In training wheel interfaces, only the most basic functions are accessible, and more advanced functions are blocked, which keeps the new user on a path of minimal error-risk, by providing an otherwise freely explorable interface. Experimental tests comparing learning a commercially available word processor with learning a scaled-down training-wheels version favored the training wheels approach. The training-wheels subjects in this experiment were later better able to explore more sophisticated functions in the full-scaled version of the application than subjects who had used the full-scaled version during training.

We have recently suggested that old, functionally simple versions of software could be used in a similar way to provide users with a less sophisticated, and therefore less error-prone exploration environment (Franzke & Rieman, 1993). In our experiment, subjects who used an early, basic version of a graphing application during training were well able to transfer the basic knowledge acquired to the higher-functionality later version of the system. In fact, learning and transfer times added together were less than the exploration times of subjects who started with the more complex, later version.

Exploratory Learning in the Context of Display-Based Systems

The widespread availability of display-based systems has undoubtedly increased the dominance of exploration as a strategy for learning new software. In Rieman's study, for example, about 90 percent of the software used was display-based, rather than command based (Rieman, personal communication), and there is a suggestion in the data that exploration is used much less frequently with command based systems. How do display-based systems then support exploration, and how could they be designed to prevent some of its pitfalls?

During the acquisition of a new application-goal memory trace, information about what needs to be accomplished (i.e., the goal: 'I want to create a graph') needs to be mapped onto display-objects in the interface (i.e., the application). This mapping process is one of searching for possible operators, and evaluating the operators in turn. We conceptualize this search in the general terms of the problem-space hypothesis (Newell & Simon, 1972; Newell, 1980). A problem space consists of a set of problem states, and a

set of operators that apply to these problem states. Problem states have a goal state and an initial state. During problem solving, a sequence of operators is applied that transform the initial state stepwise into the goal state by traversal of several intermediate states. Search in a problem space is necessary when the sequence of operators that transform the initial state into the goal state is not known. In that case, available operators need to be sought out and evaluated for their potential to lead to the desired goal state. This search can be complicated by (1) the size of the problem space, and (2) the effort of testing whether an operator is eligible for execution (Newell & Simon, 1972, p. 96). For the case of finding the next correct step toward the goal in the context of using a new display-based application, the size of the problem space is determined by the number of object-action pairs that need to be evaluated. Evaluation of the operators is achieved by a semantic match of display information and task description, a strategy called label-following (Engelbeck, 1986; Polson, 1988, Polson, Lewis, Rieman, & Wharton, 1992).

Label-Following. The model of Kitajima & Polson (1992, 1994a, 1994b) suggests a computational implementation of interpretation and decision processes in service of label-following, based on the construction-integration model for text comprehension (Kintsch, 1988). Briefly, it is assumed that the task description and the display-representation is used to sample memory elements from LTM. This elaborated task-display representation is used to construct a network of propositions in which the links are established by argument overlap. Activation is spread through the network of connections using the task description and display nodes as a source, until the spreading activation process settles. The plan element with the highest activation will be selected for execution. In principle this should be the plan element that best fulfills the constraints of the task description and the display, given the relevant memory links.

Consider the two following examples. Figures 1 and 2 represent comparable a screen states from Cricket Graph and MS Excel.

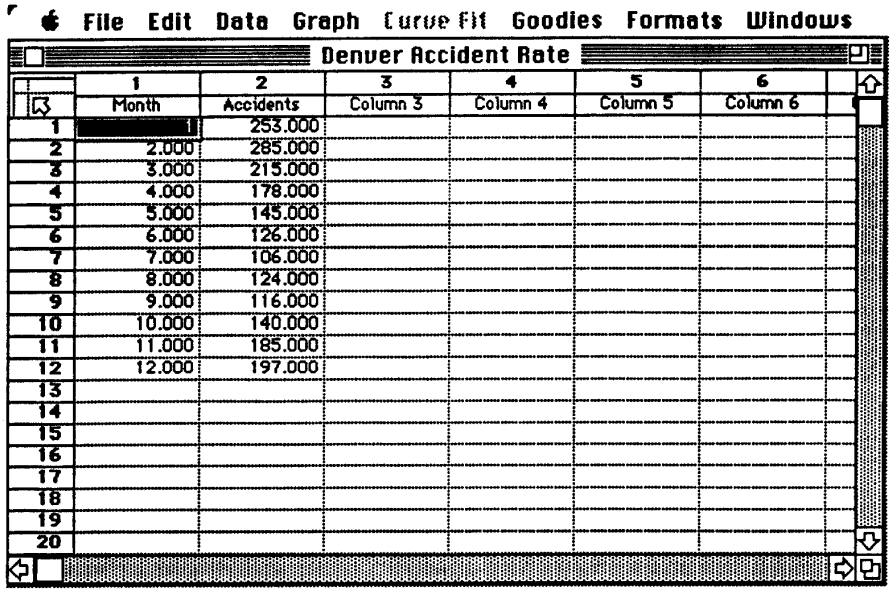


Figure 1. Example display from Cricket Graph 1. Task description is 'create a graph from the data in the file'. The appropriate next action is to select the **Graph** menu-bar item.

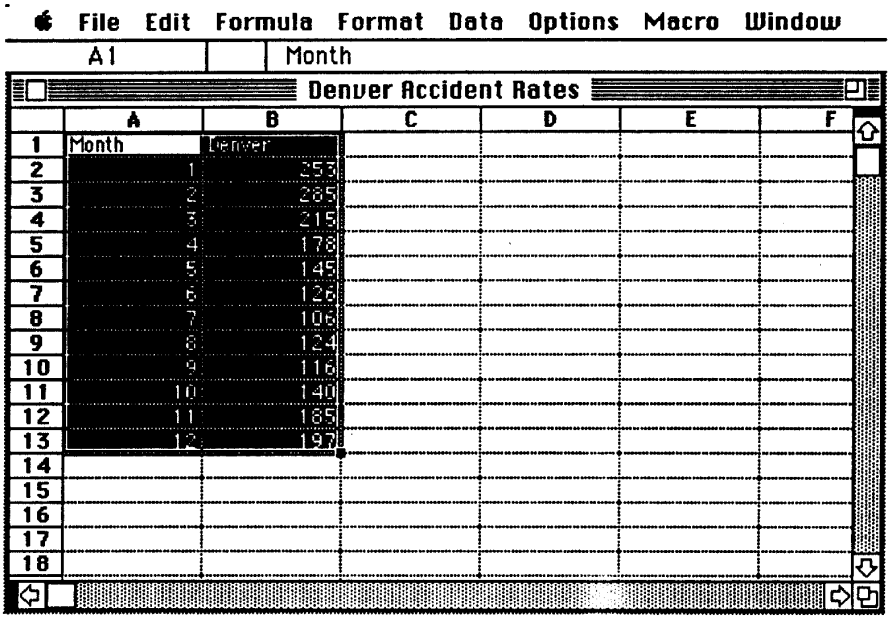


Figure 2. Example display from Microsoft Excel. Task description is 'create a graph from the data in the spreadsheet'. The appropriate next action is to select the **File** menu-bar item.

Imagine the task-description 'create a line-graph from the data in the spreadsheet'. In both cases the correct next action consists in the selection of a menu-bar item, namely **Graph** for Cricket Graph, and **File** for Excel. For the case of Cricket Graph, the task description as well as the menu-label contain the argument 'graph'. The model therefore predicts no problems with the selection of this item. In the case of Excel, on the other hand, there is no obvious direct overlap between the relevant menu item, **File**, and the task description. One could imagine a possible re-expression of the task-description as 'create a new file that contains a graph', that would lead to an eventual overlap between the task and the display description. But inferring such a mediating representation is dependent on access to relevant memory structures.

If a user does not have the background knowledge necessary for this inference (Polson, et al., 1992) or if an experienced used simply does not elaborate deeply enough because of time constraints (Kitajima & Polson, 1994a), then the relevant action may never be selected by label-following. In fact, in the Excel example illustrated in Figure 2 above, it is likely that a novice user will select the **Data** menu-bar item for further inspection, because this item represents a clear link to the task description. This will most likely lead to further searches of irrelevant display objects, and prolong the time until the relevant objects are discovered.

Rieman (1994) did an exploratory analysis of search strategies in exploration data collected in an earlier study (Franzke & Rieman, 1993). Since these search strategies were identified in the context of a similar experimental task as used here, we will review his findings briefly. Subjects were provided with the Cricket Graph software, a set of data, and the task goal to create a graph. They were also given a formatted sample graph and asked to liken the format of the default graph to the sample graph. In the situation described above in Figure 1, roughly 35 percent of the subjects demonstrated some type of label following, by either selecting **Graph** directly or after moving over other parts of the display. Another 25 percent first selected other display-objects briefly, but eventually selected **Graph**. An analysis of a later menu-search sequence, where the overlap between the goal and the relevant display-object was less straightforward, revealed a search strategy that is also generally in agreement with the Kitajima and Polson model. Here, subjects were observed to engage in *attention deepening*, that is they first spent little time selecting menu-bar items and scanning items in the drop-down menus. When no semantically promising label had been found in this first pass, they re-selected some of the menus, now sliding the cursor down to individual menu items and considering each in turn. In the protocols, 75 percent of the subjects were observed using this strategy. Rieman identified several other search strategies, such as a spatial right to left, top-to-

bottom scanning patterns, as well as a strategy he called label avoidance. In the latter case, subjects tended to avoided labels that were clearly not task related.

It is clear how a combination of these strategies would lead to longer exploration times given the two constraints defined by Newell and Simon (see above). First, if many display-items need to be scanned, it will take longer, on average, until the relevant one will be found. Therefore, overall, search should be shorter in well-defined search-sets (such as dialog boxes) than when the search set is unconstrained (at the beginning of a task, where toolbars, menu-bars, display objects, etc. all are potential search-sets). Within defined search sets, search should be shorter, on average, when the set contains few, rather than many objects. Second, exploratory search should be faster when the relevant object and label are semantically well linked to the goal. In these cases (as in the case described in Figure 1 above), the label-following heuristic works even with a very shallow elaboration. This means that search should terminate sometime during the first pass through the selected search set. If the link between label and goal is less obvious, subjects will have to engage in further scans, deepening their attention, and elaborating the display information. This by itself will lead to longer search times, and the situation may be complicated by the fact that vaguely related items will be wrongly selected, adding to the exploration time by backup and error recovery.

Experienced Performance

If a user has successfully completed an exploratory use of a function in a new interface, she now has the benefit of memory traces that have built up during the exploratory episode. In the case of graph creation with Microsoft Excel, for example, an elaboration of the task-description as 'create a new file that contains a graph', or some proposition suggesting the upper-left hand corner of the display as the relevant search area, or simple memory elements, connecting the successive display-objects **'File-New'** and **'New-Chart'**, would help in establishing the needed links. Even for items where the semantic evaluation had been simple, as for **Graph** in the Cricket Graph example, some knowledge that constrains the scan to the relevant part of the display may be of help. Overall, we would expect faster performance during later trials.

In Kitajima and Polson's model the failure to select the correct display-object may result from a retrieval failure for the expert user of an application. In our example this may happen when a needed knowledge element, say **'File-New'** simply did not get sampled during retrieval. In this case, there would be a missing link that chains the task description to the label, and the experienced user would have to search the interface again further. In Kitajima and Polson's (1994a, 1994b) simulations, retrieval failures were produced by

using a small sampling parameter, or in other words by a shallow memory search. This is discussed as an implementation of speed/accuracy trade-off. Another way to affect sampling would be to assume a decay parameter for memory traces concerning previous interactions with the application. If a user were to use a system after a say, a week-long delay, the previously established memory links would have lost in strength, and would therefore be less likely to be sampled. Again, long brittle inference chains would be most likely to be affected by this.

For our purposes, we will expect that application-goal knowledge is most helpful in situations where the link between label and task description is not straightforward. Exploration times for such situations should be long, and new users will need more help. For these problematic interactions, the performance increase between the first and a second use of the same application should be large, given that the relevant knowledge can be retrieved (after a short delay). However, if memory traces decay with time, then performance after a longer delay should be subject to retrieval failure, and performance should decrease to some extent.

CHAPTER III

RELATED EMPIRICAL FINDINGS

There are a variety of empirical studies that speak to the subject of display-based interaction. In this chapter we will show that the findings are in general agreement with our simple knowledge analysis and process assumptions. In particular, there is evidence for the necessity of procedural skill acquisition to establish a set of generally applicable base-level actions. Furthermore, results from several studies investigating skilled performance suggest that display-based interaction affords little detailed planning and thus little cognitive effort. Results from studies testing experienced users' recall memory for detailed display-information adds to this suggestion. These findings are in general agreement with the processing assumptions detailed in chapter two and Kitajima and Polson (1992).

These previous empirical studies are concerned with two extremes on the dimension of skill: they either consider novice users that have never used display-based systems before, where most of the performance difficulties have to be attributed to missing base-level actions and relevant background knowledge. On the other hand are studies that investigate use of a known set of methods in the application domain to achieve goals in the task domain. Neither case investigates exploration of new applications, or use and retention of newly acquired application-goal knowledge. The current study is an attempt to fill this gap by relating exploration and experienced performance to design criteria as discussed before.

We will also briefly review the empirical literature on menu and icon design. The main argument will be that experiments in this area are generally too limited in scope and methodology, and therefore do not bear any meaning for the questions under investigation here.

Ease of Learning

Several studies have investigated learning to use display-based systems, since one of its early promises was that it would ease the learning process (see introduction). This expectation was built on the idea that affordances in the interfaces (buttons 'want' to be clicked, objects 'want' to be moved, etc.) could be perceived with no or little training and

thus establish a feeling of 'directness'. In interaction with command-based systems this situation could only be achieved through prolonged practice and automatization of methods (e.g., Hutchins, Draper, & Norman, 1986).

The first study to challenge this point of view compared a range of systems in the context of simple file manipulation tasks (Whiteside, et al., 1985). Several command-based systems were compared to two iconic systems (direct manipulation) and one menu-driven system. For each system, old system users, transfer users, and new users were tested. Performance scores based on general completion rates for the experimental tasks showed that new computer users had just as much trouble learning the display-based systems (iconic and menu systems) as they had with command-based systems, with performance on the former group of systems even being slightly worse. Qualitative results from this study suggested that new and transfer users had trouble acquiring "...rote knowledge of a surprisingly complex syntax whose elements included mouse position, number of clicks, timing of clicks, up-stroke vs. down-stroke, choice of button, pressing while moving, and not pressing while moving. ..." (Whiteside, et al., 1985, p. 188). The authors conclude that neither of the systems supported a quick and easy start for new users. Basic interactions with the iconic representation were not picked up naturally, and the basic operations were difficult to learn and to distinguish.

This result is extended by results reported in a study by Frese et al. (1987). Here, learning and performance on two word processing systems (a command-based system: Word star, and a display-based system: MacWrite) were compared. The study method differs from the Whiteside, et al. approach in observing acquisition and performance over a longer time period (seven trials over 2 hours), rather than observing the performance of users with varying levels of experience. With this approach, Frese, et al. found that new users had great difficulties using either system during the first learning session. This results confirms the Whiteside, et al. finding. Display-based systems are evidently not more 'natural' to learn than command-based systems. However, this study shows that the display-based system outperformed the command-based system after the third session. Users of the display-based system were overall faster and required less help than users of the command-based system. It is possible that the methods that need to be learned at high cost during the first session are more widely useful and less prone to forgetting than the methods that are learned for command-based systems.

A similar finding is reported by Shneiderman & Margono (1987). Here, subjects' performance using a command-based and a display-based system for file manipulation tasks was compared. Subjects were given an introduction into using each system, and two practice tasks. It is possible that this learning alleviated some of the initial learning

problems associated with acquisition of the base-level actions, and initial training times are indeed very similar for users of both systems. Performance on a test trial however, shows advantages for the display-based system, both in terms of performance time and subjective ratings.

This line of argument is further supported by Ziegler, Vossen, and Hoppe (1986), summarized in Ziegler and Fähnrich (1988). In this experiment subjects were taught simple display-based interaction techniques in one application domain (text editing) and were then transferred to another domain (drawing). The tasks were analyzed on the background of the cognitive complexity (CCT) framework (Kieras & Polson, 1985), and it could be shown that basic interaction techniques transferred from one to another application, if the low-level syntax of the methods (e.g. selecting text and selecting graphical objects) was compatible.

Finally, a study by Streitz, Lieser, and Wolters (1989) adds to these observations. One of the factors in this study compared command-based with display-based editors and system software. Here, subjects also received an introduction, a demo and practice tasks, before encountering the experimental trials. Again, subjects using the display-based system did overall better than the subjects using the command-based systems. However, this effect was modified by another factor, the use of appropriate metaphors in labeling and depicting display objects. If an appropriate metaphor was used, subjects were faster using the display-based system, but when semantically meaningless (to the subjects) technical terms were used, their performance was just as poor as with the command-based system. This study therefore also includes evidence for our hypothesis that the provision of appropriate labels improves performance.

If base-level actions are indeed difficult to acquire, what is the nature of the difficulties? In the previous chapter we already mentioned work by Carroll and Mazur (1986) that addressed this problem at a greater level of detail. In this study, six new users of the Lisa, an early ancestor of the Macintosh series, were observed during various activities connected with learning a new system: getting started (reading manuals, turning the system on), interactions with an on-line tutorial, and interactions with particular parts of the interface (the desktop, and some of the applications). This study details some of the confusions that arose from having to learn the 'complex grammar' of mouse-interactions (see Whiteside, et al., 1985, citation above), such as the timing of clicks, distinguishing between different types of clicks (single-clicking, click-and-pressing, double-clicking, etc.), and the consequences of these actions depending on the context of their application. Similarly, difficulties in the acquisition of concepts and objects used in the desktop are reported (see also summary in chapter two).

All of the reviewed studies document situations in which new users of display-based systems are learning a new application and the basic interaction techniques in the same session. They support our argument developed in chapter two above, that basic-level action knowledge is of procedural kind, and will therefore be acquired through similar learning mechanisms proposed for other procedural skills (e.g. Anderson, 1987). In particular, the observation from the Whiteside et al. and Carroll and Mazur studies provide evidence for a declarative learning phase, in which the concepts of the new domain need to be acquired and distinguished; and where potential operators have to be learned and differentiated. Out of such knowledge elements action knowledge can be acquired, such as 'if you want to open MS Word, double-click the MS Word icon'. Rieman, Lewis, Polson, and Young (1994) describe how memory traces of such singular interactions could be slowly generalized by a process of analogical reasoning. The product of this gradual learning process would be compiled productions (or base-level actions), for, say 'selecting text', or 'opening an application', that apply across task contexts and applications, if the interface is consistent.

In summary, the learning results documented in the empirical literature, are in agreement with the knowledge analysis brought forward in chapter two. However they mainly concern the learning of basic action elements and general background knowledge associated with Macintosh systems. The current study investigates exploration of new interfaces on the background of such knowledge, and the use of application-goal knowledge acquired in this situation.

Ease of Use

How do display-based applications support performance in the problem domain (e.g. text editing, creating of graphs) once the basic-level actions and application-goal knowledge have been learned? The answer to this question seems to be: it depends. Secondary (problem domain-) performance seems to depend on the complexity of the tasks that are performed in the problem domain.

Two series of studies have investigated performance on secondary tasks mediated by use of either a display-based or a command-based system (Svendsen, 1991; O'Hara & Payne, in preparation), and found poorer performance with display-based systems. In all of these cases, the investigated problem domain was relatively static and simple:

Svendsen (1991) investigated the learning of the Tower-of-Hanoi (ToH) problem in the context of either a command-driven or direct-manipulation interface. Subjects either typed commands, such as "from 1 to 3", or moved the disks by dragging them from one

peg to another. He found that subjects were overall faster to reach the learning criterion in the direct manipulation condition, however this success came at the cost of more overall moves and a greater amount of errors. Furthermore, he found that only about 35 percent of the direct manipulation subjects were able to verbalize a rule governing the solution of the ToH problem, compared to approximately 80 percent of the subjects who had used the command-based interface. Nevertheless, subjects in this experiment preferred the direct-manipulation system and greatly underestimated the number of errors they committed with this interface.

O'Hara and Payne (in preparation) argue that Svendsen's result could be explained by a cost/benefit analysis. Since subjects had to pay a higher implementation cost of each move in the ToH problem space when using the command-based system, they might have planned each step more carefully, and therefore spent more time analyzing and learning the problem domain. To investigate this hypothesis, they report four studies in which they varied the cost associated with operator implementation along three dimensions (number of keystrokes, system lockout time, and availability of undo commands). In these experiments (where subjects were either doing a series of 'Eight puzzles', or a similar problem solving puzzle), they established repeatedly that high cost moves will reduce the number of moves to solution. In a transfer experiment (where subjects were transferred to a problem isomorph of the Eight puzzle), subjects who used the high-cost interface during training performed better than subjects in the low-cost condition.

Contradictory results are reported in a study by Benson, et al. (1989). These authors compared two versions of a system that allows inventory control for a large manufacturing plant. Problem solving in this domain is complex and dynamic, since the conditions of manufacturing change with a several more or less predictable parameters, such as machine failures, or availability of production materials. Unfortunately, the design of the two compared systems varied on several dimensions. The display-based system presented information more efficiently, and also allowed for 'lower cost' mouse-input. Within these confines, results from this study show that the display-based system allowed for better achievement of goals in the problem domain, namely achieving a low-cost factor in the process of controlling manufacturing parts movement. Users of the display-based version were intervening more often, therefore executing more control over the manufacturing process. Evidently the better display of critical information and the lower of cost of interaction helped subjects to intervene when the situation called for it.

A study by Ballas, et al. (1992) compared performance in complex and simple decision making tasks directly. Furthermore, this study controlled the effects of graphical representation of critical information and input mode in a factorial design. In the context

of a tactical assessment system used in military aircraft, they show that graphical representation of aircraft information outperforms tabular display of the same information in situations where information from many sources needs to be combined. When subjects simply had to confirm the location of pre-classified aircraft, tabulated information was more efficient. But when subjects had to decide whether a target was hostile or neutral, based on information about its speed, bearing, and distance, graphical display of this information was easier to interpret.

With respect to the input mode (command-line or mouse-driven) this study reports an overall better performance for system designs that allowed for direct manipulation input. This performance was measured as the time loss on a primary task after interruption and use of the tactical assessment tool (secondary task). If the tactical assessment tool allowed for direct manipulation input, performance on the primary task was less impoverished than when the input mode was command-based. This result is complemented by a study by Eberleh, Korfmacher, and Streitz (1992). Here, subjects performing a routine task on either a command-based or direct-manipulation system were interrupted to perform secondary tasks of varying levels of difficulty. Performance on both tasks was measured. Performance on the primary task was impaired more by difficult secondary tasks (arithmetic problems) than by simple tasks (feature match or simple target detection). Performance on the secondary task, however, was only impaired when subjects were using the command-based system, and only for the most difficult problem group (arithmetic).

The reviewed studies focus on different aspects of display-based interaction (implementation cost for operators vs. availability of graphical displays for complex information) or even confound these two factors. A simple interpretation of the results is therefore difficult. Nevertheless, we would like to suggest the following hypothesis: In situations where task-domain performance is relatively simple and static (problem solving in the context of the Tower of Hanoi, or the Eight puzzle) it is possible that the low cost of implementing moves in direct manipulation systems leads subjects from thoughtful planning of moves to a less effortful trial-and-error strategy of accomplishing task goals. In such contexts, interactions with display-based systems may diminish overall productivity in the long run, because users lack the rules to perform efficiently in the problem domain, or will acquire such rules more slowly. If the problem domain is complex and performance is time-critical, (manufacturing parts control, or military target classification), cognitive resources may already be spent at their limit in analyzing the problem domain. In situations like such, using mouse-driven input may improve performance in the problem domain, because less cognitive effort is needed to plan actions

in the application domain. Investigation of this hypothesis would require control over the initial skill-level in the problem domains, and a separation of the effects of display-mode and input-mode.

While none of these performance studies address particular processes producing interactions with display-based systems, they provide general evidence for the hypothesis that these interactions do not have to be planned carefully, and therefore require less cognitive effort. This is in general disagreement with theories that propose retrieval and execution of complex hierarchical plans to achieve task goals.

Memory for Display-Based Interactions

Early cognitive theories of Human Computer Interaction successfully predicted behavior in the context of command-based systems by modeling it in form of complex hierarchical productions systems. In general, these task representations break high-level task goals into subgoals, and these again into methods and operators for their execution. These structures are encoded in productions and have to be recalled from LTM. Several successful examples of such theories are GOMS (Card, Moran, & Newell, 1983) CCT, (Kieras & Polson, 1985), and on the side of competence theories, TAG (Payne & Green, 1986). All of these theories make one critical assumption: Successful interaction with computers or software is critically dependent on *recall* of the necessary action plans, methods, and operators.

This basic assumption has been challenged by three studies that provide evidence against the view that display-based interaction is also entirely based on recall detailed action plans. They suggest that performance in this context is guided by cued recall or recognition of relevant screen objects, at least at the level of methods and operators.

Mayes, et al. (1988) asked subjects, ranging in MacWrite experience from occasional to frequent (all had at least one year of Macintosh experience) to reproduce exact screen states from a typical editing task using MacWrite. His recall questions asked subjects to draw screen states given a general description of that state (i.e. "If you select MacWrite what appears on the screen? Please draw it in as much detail as possible", Mayes, et al., 1988, p. 279). Amongst some of the to be recalled items were the contents of the Finder menus, MacWrite menus, the Desktop, etc. Even the frequent users of McWrite could only recall about 50 percent of the requested items. In a follow-up study, in which the same subjects were asked to perform tasks using the non-recalled items, they displayed no difficulty or hesitation.

Similar results were recently established in a pilot study reported in Smelcer &

Walker (1993). These authors asked frequent users of MS Word to recall the names and locations of 12 commands, given the names of menu bar-items as a help. These authors found that only items used with the highest frequency were named and placed correctly, all other items could be placed approximately, and in many cases synonyms were given as their label.

Payne (1991) followed Mayes et al.'s research up by asking more specific questions about singular display-consequences of particular actions. In particular, he asked for the exact location of the cursor after cursor-movement commands associated with the word processor that his subjects had used the most (an average of about 700 hours). His results show that rather than relying on exact knowledge of the consequence of the cursor-movement commands, subjects merely guessed a plausible location, and this lead to a large amount of mispredictions in his data. In a second study he investigated subjects knowledge of interactions in the dialog-box associated with the Find command in MS Word or MacWrite. Here, he asked to specify the sequence of actions needed to find a word, and asked for descriptions of screen states as consequences of a few specific requests. His data show that subjects have an approximate memory of the type and order of actions that have to be performed, but they had difficulties reproducing the exact labels connected with these actions. The questions targeting knowledge about the effects of commands were answered very poorly, indicating weak knowledge at this level of detail of action consequences. Payne concludes that users of display-based systems do not have access to details of screen states that would be necessary in planning the next step ahead. For example, subjects may remember correctly that the next correct action is a button press, but they rely on the screen state for the exact labeling of that button.

While these studies do not rule out that subjects may have simple, incomplete plans for frequently occurring task-goals (maybe 'select interface object, use menu, interact with dialog box') as the answers to Payne's Finder questions indicate, they suggest that subjects cannot have access to plans that implement interactions at a low level of operator execution. In other words, it is not likely that subjects have action knowledge of the type 'to change the text to underline, ..., and select the menu-bar item in the middle of the screen with the label **Format**, and release on the menu item **Bold** on the bottom of the menu.' Rather, it would be likely that experienced subjects will have some general high level plans of the type (to change the appearance of text, select text, use the menus, use the dialog box), where the details of the interaction have to be constructed in interaction with the display as suggested in chapter two.

Overall the empirical results available are in agreement with the theoretical assumptions developed in chapter two, without supporting any of the more detailed

processing assumptions specified in models such as Kitajima and Polson's. However, there is evidence that a set of basic productions has to be acquired in a regular process of skill acquisition. Furthermore the evidence suggests that skilled display-based interaction does not require much planning and cognitive effort, and that the details of interactions can only be worked out by reliance on information in the display.

There are no studies to date that address exploratory learning of a new application, with users that already have basic interaction knowledge, that is base-level actions and relevant background knowledge.

Of Menus and Icons

Display-based interactions are to a great part based on interactions with menus and iconic representations in toolbars, on the desktop, etc. There are numerous experiments studying particular design features of menus, such as the effects of breadth vs. depth in hierarchical menus (e.g. Miller, 1981; Lee & McGregor, 1985; Paap & Roske-Hofstrand, 1986, Norman, 1990; Parkinson, Hill, Sisson, Viera, 1988; Van Hoe, Poupeye, Vandierendonck, & De Soete, 1990), number vs. letter selection (Perlman, 1984), the semantic organization of menu-items (e.g. Card, 1982; Halgren & Cooke, 1993; Smelcer & Walker, 1993), the effect of explicit or implicit targets (e.g. Mc Donald, Stone & Liebelt, 1983; Halgren & Cooke, 1993; Smelcer & Walker, 1993), varying display-techniques for menus (e.g. Apperley & Field, 1985; Mills, & Prime, 1990; Field & Apperley, 1990; McLeod & Tillson, 1990), as well as psychomotor aspects related to selection from different menu types (e.g. Walker, Smelcer, & Nilsen, 1991; Kurtenbach, Sellen, & Buxton, 1993; Kurtenbach & Buxton, 1993). In this group of studies there are also a few that are concerned with design features of icons (Byrne, 1993), and comparisons of the use of icons to text, or combinations of both (Nielsen, 1990, Landsdale, Simpson, & Stroud, 1990; Kacmar & Carey, 1991; Bensabat & Todd, 1993).

However, we hold that the material, the tasks and the designs of these studies constrain their results greatly, so that they can not be applied to enlighten general models of display-based interaction, or exploration.

First, a large number of studies investigating design features of menus, such as the depth vs. breadth tradeoff, was not conducted on Macintosh-style pull-down menus. Instead, these studies investigate menus, where the root of the menu is represented as a vertically extended list of menu-names that can be selected by some type of keyboard input (e.g. Perlman, 1985, Apperley & Field, 1985, van Hoe, et al., 1990, Paap, & Roske-Hofstrand, 1986; Parkinson, et al. 1988). Selection of one menu-item from the root

wipes out all information about the root and displays instead the choices on the next level. Typically, users therefore only have access to information at a particular level and branch in the menu-structure, without any reminders about possible choices in other branches. This situation differs considerably from the display of information given in pull-down menus: With pull-down menus the entire menu-bar (root menu) stays visible on the screen when one of the items is selected, location of the dropped menu, and highlighting of the menu-bar item provides additional information about which particular menu has been selected. If a menu branches further into a nested menu, all of the selection history and basic information about alternative choices at the menu-bar level are visible and accessible. Even with the display of a dialog-box some of the selection history is preserved (the respective menu-item in the menu bar is still highlighted).

This difference has several consequences for information processing in the context of menu-displays. On one hand, if some information about alternative menus is available making choices in one particular branch of the menu hierarchy, this information can potentially influence the decision about selecting a particular menu item. Consider for example a situation in which a menu-name sounded very promising (say, **Graph**), when the task goal was to create a graph from a set of data. Surprisingly, the lower level menu-items displayed do not link up to this goal at all. However, there is another menu-bar item at the root level that is labeled **Data**. In this situation we would expect the subject to abandon the original menu-choice and investigate the other menu that seems to have a link to the task goal. In the case of traditional menus, information about the root menus would not have been visible at this point, and the subject might have picked one of the choices in the **Graph** branch, unless he or she remembered some of the labels in the root menu.

Furthermore, several aspects in the design of traditional menus make selection of choices a higher cost operation: first, users loose display-information about their selection history. Unless they preserve their most recent choices in WM, users are more likely to get lost in the menu-hierarchy. There are two potential drawbacks associated with this, either the cost of keeping history information in working memory, or the cost of backing out of a unknown position in the menu-structure. Additionally, the operators for menu selections and backups are more costly. Instead of using generic productions for mouse-pointing and -clicking, subjects have to enter specific keystrokes for each menu-choice. The work reviewed above by O'Hara and Payne and Svendsen points out that different cost associated with operator selection will produce different choice behavior and learning results.

To summarize, interactions with traditional menus most likely produce different behavior because of three main processing characteristics: Less information is available to

users via the display, decisions are based on a different set of information, and will therefore produce different results. This situation could be prevented by keeping relevant choice histories in Working Memory, but this would produce additional cognitive effort that may affect decision making by taking resources away from it. Additionally, each menu-selection is associated with a higher cost than in pull-down menus. This will lead to more planful and cautious choice behavior and different learning than what can be expected with pull- down menus. Finally, the interaction of all these factors, higher memory load, access to less information during decision making, and the higher cost associated with menu selections is most likely to produce behavior that is categorically different than the search and selection behavior in the context of pull-down menus.

On another level, many studies of menu- or icon-design use very narrow experimental tasks. In all cases, subjects know that the correct search space is the set of menu or icon items presented to them. In other words, there are no competing search sets (such as tool bars, graphical objects in the display, etc.) that could be scanned and searched alternatively. Furthermore, in many cases, subjects are provided with an explicit target, so for example they are told to find the **Repaginate** menu item (e.g. Perlman, 1984, Norman, 1990, and Card, 1982, Byrne, 1993). This situation, where subjects know the exact name of the function they are searching for, is however not representative even for expert users of a display-based application (see above, Payne, 1990, Mayes, et al., and Smelcer and Walker, 1993). This problem has been discussed and investigated further, leading to the distinction between *explicit* and *implicit* or *fuzzy* target search (McDonald, Stone and Liebelt, 1983; Paap & Roske-Hofstrand, 1988, Smelcer & Walker, 1993; and Halgren & Cooke, 1993). Comparisons between the two types of search (the latter being closer to our case), have shown that organization of menu items (categorical, alphabetical) has an effect on search times when the search is for an implicit target, but not when the search is for an explicit target. In cases of implicit target search the results from these studies tend to favor categorical organization of the menu items. This particular result seems to generalize across different menu-styles.

Finally, many of these studies are designed to produce rigorous factorial designs to allow for comparisons between dichotomous design alternatives and their interactions. Satisfying these experimental design constraints often results in interface designs that are artificial and unlikely to be found in systems developed for real use (one hopes). An experiment by Bensabat & Todd (1993), for example, was designed to test whether text labels work better than iconic depictions of choices, and at the same time whether menu-selections are better than direct manipulation (dragging and dropping) of the same symbols. Factorial combination of these dimensions produced menu-icon, menu-text,

direct-manipulation-text, direct-manipulation-icon. In the direct-manipulation-text condition, for example, functions could be executed by dragging text objects (e.g. 'file') onto other text objects ('print'). In the menu-icon condition depictions of these meanings had to be selected from two menus successively, then an 'apply' button had to be pressed. Furthermore, all four designs were constructed to equal the amount of button presses and mouse movements between the conditions, producing artificially constrained behavior for the menu-conditions. Even though this study managed to control a few intervening factors (such as number and layout of the display objects, number of keystrokes per condition), it produced an artificially convoluted protocol for the two menu conditions that unfairly biased the performance in these conditions. It is questionable whether results from experiments like this generalize to anything beyond the experimental designs.

CHAPTER II

METHOD

Subjects

Thirty-five men and forty-five women, for a total of eighty subjects participated in this experiment. Participation was elicited by newspaper advertisements and postings throughout the department and in Macintosh laboratories on campus. Subjects were screened for previous Macintosh and graphing experience. Only subjects with at least six months of Macintosh experience, but no previous experience in using graphing software were selected. A few subjects had experience with spreadsheets, but they had never used the graphing functionality embedded in them. Subjects were paid $15 for participation. The data of four subjects were excluded from the analyses, because of failures to complete the tasks in the exploration phase.

Design

Subjects were randomly assigned to one of eight experimental groups (four interfaces by two delay conditions). The interfaces were CGI[1], CGIII[2], EXC tool[3], and EXC menu[4]. Subjects were assigned to one interface and used it throughout the whole experiment. All subjects performed three warm-up and three graphing tasks; see Figure 3 for an illustration. The tasks will be described in detail below.

Task 1 constituted the *exploration* trial. Task 2 tested *experienced performance.* Half of the subjects performed the second task after a short delay (approximately ten minutes), and the other half after a long delay (one week) delay. Inclusion of the third task required all subjects to come in for a second session with a week delay between sessions, and therefore prevented effects of self-selection. Data from the third task have been included in overall analyses of learning effects. Analyses of delay effects rely only on data from the first and second graphing tasks. Immediately before each graphing task

---

[1]CA Cricket Graph, version 1.3.2, 1989.
[2]CA Cricket Graph III, version 1.01, 1992.
[3]MS EXCEL, version 3.0, 1990, toolbar usable.
[4]MS EXCEL, version 3.0, 1990, toolbar disabled.

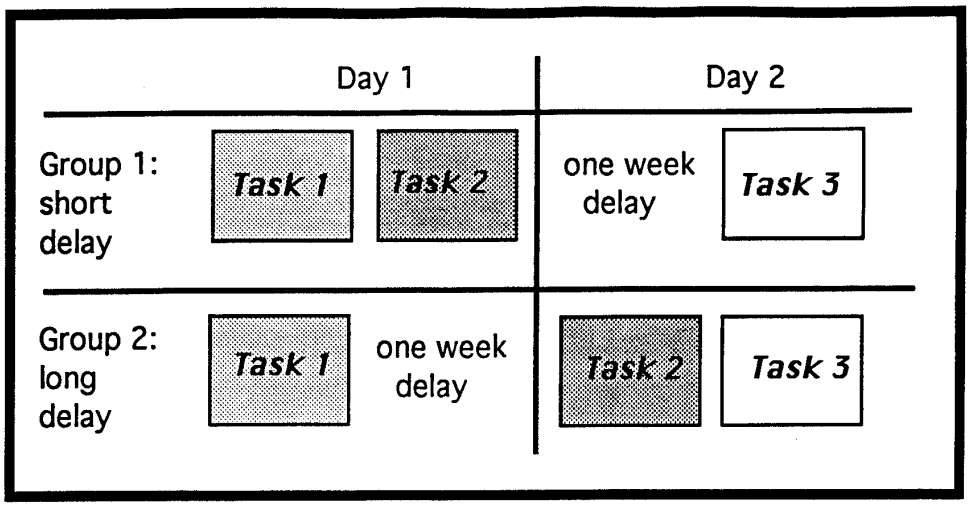


Figure 3. Basic delay design of the experiment.

(Tasks 1-3 in Figure 3), subjects received a warm-up task that involved routine editing of text documents (using MS Word [5]). This setup allowed subjects to warm up to the procedure involved in toggling between instructions and the application (see description below), and in general to warm up to the experimental situation.

Materials and Apparatus

In the course of the experiment, subjects were provided with a printed *background questionnaire,* with *instruction material* , and with *application software* to accomplish the tasks.

Background Questionnaire.

Subjects filled out a simple two-page questionnaire that asked about basic demographic information (age, education, sex), and about relevant background knowledge (years of Macintosh and PC use, and an estimation of the number of applications used in each domain). The questionnaire also asked subjects to estimate the number of graphs that they had drawn by hand during their life. See Appendix A for the questionnaire.

---

[5]MS Word, version 4.0, 1990.

Instruction Material

To accomplish the warm-up and the experimental tasks, subjects were provided with HyperCard™ stacks containing the instructions. All instructions stacks contained four cards, containing general and specific task information. The first card started the experiment. The second card provided subjects with a general task description. The third card gave subjects more specific information and a sample of the type of goals that needed to be accomplished. The fourth card gave subjects specific information that helped them to get started, and asked them to start the procedure unless they had any questions. Subjects could page through these cards by clicking on labeled buttons. After subjects had started the experiment, they received detailed task information through interaction with the third card of the stack (see Figure 4 for an example).

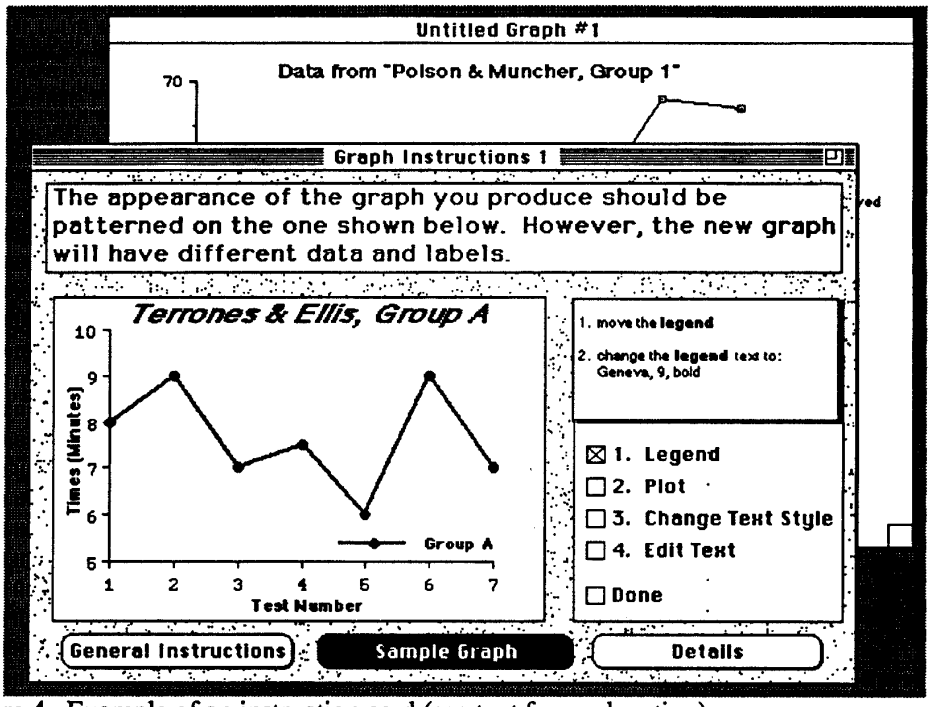


Figure 4. Example of an instruction card (see text for explanation).

This card contained a field with an example of the formats that needed to be accomplished (both, for the editing and the graphing tasks), and a field with a sequence of buttons that displayed information about the next task to be accomplished and information needed (for example, the font size, or the content of text to be edited). This set-up ensured that subjects were working on the task goals in a specific sequence, and did not have to

guess on the exact match for styles and sizes. The instructions were written to not contain any hints about the methods needed to accomplish any of the tasks. Appendix B contains examples for a complete sequence of three cards for one of the warm-up and for one of the experimental tasks. It also contains examples for the task information provided on card three by interaction with the button sequence.

Application Software

For the tasks, four software packages were used (see footnotes 1-5 for release details). For the warm-up tasks all subjects used Microsoft word to edit three text documents that were provided to them. For the graphing tasks, subjects used either an early release of Cricket Graph (CGI), a later version of the same package (CGIII), or MS Excel (EXC tool or EXC menu). For all graphing tasks subjects were given a file containing the data to be plotted. This file was a document created by the application that the respective subject was assigned to. All used applications present the data in form of a spreadsheet. The main differences with respect to the graphing functionality exist in the methods needed to create a graph from the data, and in the methods to accomplish modifications to the format of various graph objects.

In CGI, CGIII, and EXC menu, graphs are created via some interaction with menus and dialog-boxes. For examples of the complete interaction sequences required for graph creation in the four interfaces, please see Appendix C. The main difference between CGI and CGIII is in more nested menus and more complex dialog-boxes for the later version. In EXC menu, the toolbar was hidden, so that subjects had to use the menu-method for creating the graph. Here, subjects first have to select the relevant data in the spreadsheet, and then use a combination of menus and dialog boxes to create the graph (see Appendix C.3). In EXC tool, the toolbar was accessible to subjects, and subjects were guided to use the toolbar-method, if they did not discover either method on their own. This method differed greatly from all other graph creation methods, as it required data selection, tool selection and a determination of the graph area on the screen via direct manipulation (see Appendix C.4).

The main differences for graph modification (changing the formats of graph objects) are as follows. CGI allows for one standard method to access any of the graph objects, that is double clicking on the object to be modified. CGIII adds to this menu-based access to most of the graphing objects. Editing changes to text in the graph have to be accomplished by using a tool from a toolbar. Double-clicking, and menu interactions are also the standard way to accomplish modifications in both Excel cases, where EXC tool additionally offers interactions with icons in the tool bar for some of the changes. In

Excel, editing changes to text are accomplished by selecting the text to be modified and typing in the formula bar that is displayed directly under the menus.

The instructions stack and the application were partially overlapping on the computer screen (see Figure 4). If subjects wanted to consult the instructions, they could click on the stack, which brought the instructions to the front. To work on the application, this could also be brought to the front by simply clicking on it. This procedure allowed us to separate instruction reading times from times spent working in the application domain.

Apparatus

Subjects worked on an Apple Macintosh II cx with a 13" color monitor, set to black and white. A video camera was installed behind the subjects back, so that all screen interactions could be videotaped. An audio track was also recorded. The experimenter was sitting behind the subject, to monitor the camera and the experimental session (see Figure 5). A large wall clock helped the experimenter to decide when to interrupt for occasional hints.

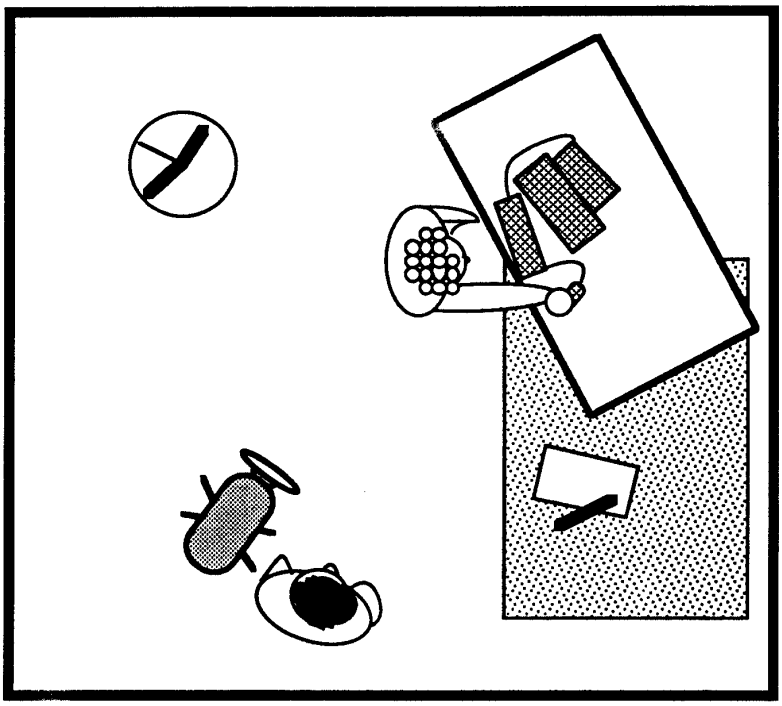


Figure 5: Set-up of video-camera and computer.

Tasks

During the course of the experiment, subjects were given a *thinking-aloud warm-up task,* a series of three *editing warm-ups,* and the three *experimental tasks.*

Thinking-Aloud Warm-Up

During the whole course of the experiment, thinking-aloud protocols were collected to monitor the subjects' goal states and to help decide when subjects had gotten stuck and needed a hint. To familiarize the subjects with thinking out loud, they were given a brief explanation of the method and were asked to perform the window-counting task (Ericsson & Simon, 1984). This task asks subjects to count the number of windows in a house familiar to them and verbalize the mental states that allow them to do so.

Editing Warm-Ups

As mentioned above, subject were asked to do three editing tasks throughout the experiment. These editing tasks were administered prior to the three experimental graphing tasks. These warm-up tasks familiarized subjects with the interactions needed to receive the computerized instructions, such as toggling between two windows, and receiving subtasks by button-clicks and integration information from the instruction text and the sample document in the HyperCard stack (see Figure 4, and Appendix B). For these three tasks, subjects were provided with three unformatted MS Word documents (a 'table', a 'book chapter' and a 'letter'), and were asked to liken these in format to equivalent samples in the instructions stack. In all of these tasks, subjects were asked to fulfill four simple subgoals, such as 'Change the font of the chapter title to 'Times'. Subjects were asked to accomplish these tasks at their own speed, while verbalizing their thought processes. During the editing warm-ups, the experimenter responded to questions, or offered help, to ensure that subjects understood the instruction-receiving procedure.

Experimental Graphing Tasks

In the three experimental graphing tasks, subjects were asked to complete a series of subtasks. Specifically, they were to (1) create a line graph from the data in the file provided to them. Then subjects were instructed to (2) move the legend to a specified location, (3) change the font size of the legend text, (4) change the line and symbol style of the plotted graph, (5) change the font and style of graph title, (6/7) change the font and style of both axis texts, and (8/9) edit the title and x-axis title content. For an example of a

sequence of these instructions please consider the example in Appendix B.

All three tasks asked subjects to accomplish these same subgoals in the same order, however the data provided, and the format of the default and target graphs varied from task to task. The formats of the default and target graphs were kept constant across the four different interfaces, however. For examples of the default and target graphs of the experimental tasks one to three, please consider Appendix D.

Procedure

During the first session, subjects were asked to read and sign the informed consent material, and to fill out the questionnaire. This lasted approximately five minutes. After this the experimenter activated the video camera and microphone. The experimenter gave a brief introduction in the thinking-aloud procedure and administered the window-counting task, which took approximately three minutes. Then, the subjects' attention was directed towards the computer, and the experimenter introduced the computer tasks, by telling subjects that they were to accomplish two (four) tasks using different kinds of applications. The subjects were shown the HyperCard stack containing the instructions for the first editing warm-up. They were asked to read the instructions out loud, and to keep thinking out loud throughout the whole task. After completion of this first editing warm-up, (which took between five and ten minutes), subjects were provided with the instructions for the first experimental graphing task, and a partially hidden window that contained the application and the data file. These files were presented in icon format. Subjects were told that they could not ask any further questions, and that they had to 'figure out how to accomplish the tasks by themselves'. They were also reminded to read the instructions aloud and keep verbalizing. Through the experimental tasks, experimenter intervention was kept at a minimum. The experimenter only intervened when subject had not made any progress towards the next correct action in more than two minutes, or when subjects had gotten stuck in a difficult error recovery. This first experimental task lasted approximately twenty minutes. After this, half of the subjects (the long delay condition) were excused. The other half of the subjects were given a second editing task, and the second experimental graphing task. Experimenter intervention during these two tasks was again kept at a minimum. This second half of the computer tasks was considerably shorter, approximately 10 to 15 minutes.

The second session was scheduled to be exactly one week later, or as close to that delay as could be made possible. No subjects were permitted to come for the second session in less than seven days, and no delays longer than ten delays were accepted. In

the second session, the subjects performed the remaining two (four) tasks involving the editing warm-ups and the experimental graphing tasks. Subjects were immediately seated in front of the computer, and experimenter intervention was minimal as specified above. This second session was considerably shorter, lasting about 10 minutes in the 'short delay' condition (two tasks), and about 15 to 20 minutes in the 'long delay' condition (four tasks). After all tasks had been accomplished, subject were completely debriefed, all additional questions were answered and subjects were paid and excused.

CHAPTER V

RESULTS

Data Transcription and Scoring Methods

All on-screen interactions were videotaped and time-stamped with an accuracy of one second intervals. These videotapes were transcribed as follows: For all of the subtasks in the graphing tasks, and for each interface, one (or several alternative) correct action sequences were determined. The action sequences for graph creation in the four interfaces are documented in Appendix C. For CGI, for example, the correct action sequence for graph creation is: click-on Graph (a menu-bar item selection), release-on Line-Graph (a menu item selection), click-on Serial Position in the list associated with Horizontal (X) Axis (a selection from a list in a dialog-box), click-on Serial Position in the list associated with Vertical (Y) Axis (a selection from a list in a dialog-box), click-on New Plot (clicking of a button in a dialog-box).

For each such action step in fulfillment of the tasks five performance measures were obtained: First, the *action time*, the time spent viewing *instructions*, and the time spent in *corrections* was recorded. From these measures the *overall completion time* was derived by adding all three components. Finally, the number of hints given per action step were recorded. An example of a raw transcript is included in Appendix E.

Corrections were all interactions that were not clearly on the solution path, such as struggling to undo an unwanted function, or interactions away from the application (some subjects closed the window that contained the relevant application and data icon, and the experimenter had to intervene). Corrections are therefore a collection of interactions particular to the experimental setting, rather than a measurement that can be related to the graphing applications. Times associated with this category are small (see below).

The results will be reported in two sections: First, global results concerning overall performance measures will be presented. These are results such as overall learning results, or the influence of previous experience on task performance. Second, results from the detailed analyses will be reported. In that section the main focus is in showing how the three design parameters, type of basic-level action, the quality of the labels, and the number of interface objects to search influenced performance during exploration and in later trials. It will be described how design parameters were extracted from the interfaces, how they were coded, and the effects on the subjects' performance will be reported.

Global Results

Effects of Training

Subjects' performance showed a large improvement between the three trials across interfaces and delay conditions (see Figure 6). Overall, subjects were able to cut their completion times between the first and the second trial by 58 percent, from an average of 19 to an average of about 8 minutes for the whole task.

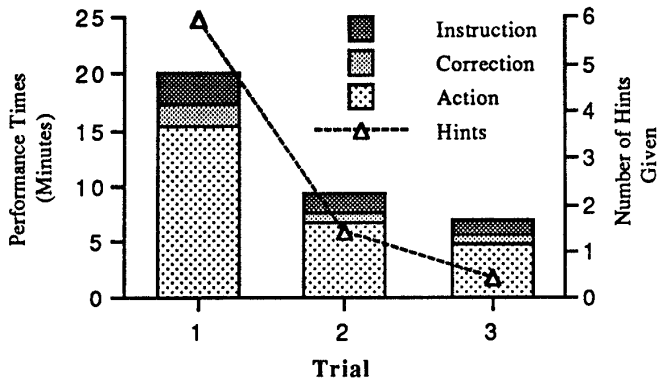


Figure 6. Performance Improvement due to Practice.

Between the second and third trial, performance decreased by another 25 percent to about 6 minutes. Simple one-factorial repeated-measure ANOVA's were used to test these differences. The differences were significant for all performance measures: action times ($F(2, 124) = 163.85$, $p. < .01$), instruction times ($F(2, 120) = 59.52$, $p. < .01$), correction times ($F(2, 118) = 14.65$, $p. < .01$), overall completion times ($F(2, 114) = 136.98$, $p. < .01$), and the number of hints ($F(2, 106) = 120.40$, $p. < .01$). These results demonstrate that interface literate users were able to discover functionality in a new system in a reasonable amount of time, without extensive use of external help, and were able to use the discovered methods efficiently in further trials.

Effect of Experience

The subjects in our experiment had quite varied previous computer experience, as exposed by the background questionnaire. This information is summarized in Table 1. Subjects' overall completion times during the three tasks were regressed on these background experience variables (years of Macintosh experience, years of PC experience,

Table 1.
Background Experience of the Experimental Population

	Average	Std Dev..	(Range)
Years of Macintosh Experience	2.7	1.7	(.05 - 6)
Number of Macintosh Applications	2.9	1.8	(1 - 8)
Years of PC Experience	1.6	1.8	(0 - 5)
Number of PC Applications	1.8	1.6	(0 - 7)
Age	24.9	6.2	(15 - 44)
Educational level (1 = High school Diploma, 7 = Ph.D.)	4.3	1.2	(1 - 7)

number of Macintosh applications previously used, and number of PC applications previously used), and the demographic variables (age, gender, and educational level). Only the number of previously used Macintosh applications emerged as a reliable overall predictor of completion times ($F(1, 46) = 9.60$, $p. < .01$).

On the basis of this screening regression the subjects were divided into two "low-knowledge" and "high-knowledge" groups. All subjects who had used more than three Macintosh applications previously were treated as high-knowledge, and all other subjects as low-knowledge. This variable was used in five two-factorial, mixed ANOVA's (trial, within subjects; knowledge, between subjects) on the five groups of performance measures. The main effect of knowledge was significant for overall completion times ($F(1, 57) = 8.59$, $p. < .01$), as shown in Figure 7.

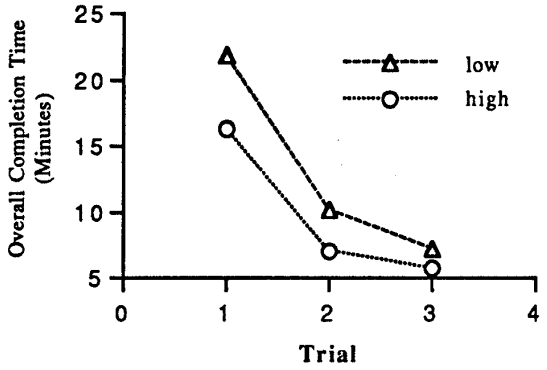


Figure 7. Overall Completion Times for Trials 1-3 by Knowledge Level.

There were also significant differences between high and low knowledge subjects with respect to action times ($F(1, 62) = 8.52$, $p. < .01$), instruction reading times ($F(1, 60) = 7.86$, $p. < .01$), and correction times ($F(1, 59) = 4.74$, $p. < .05$). For the number of hints the differences were marginally significant ($F(1, 53) = 3.69$, $p. = .06$). None of the interactions between the factors trial and knowledge level were significant below the .05 level, even though two of the interactions were marginally significant (action times, $F(2, 124) = 2.59$, $p. = .08$, and instruction times, $F(2, 120) = 2.61$, $p. = .08$). Table 2 includes the means for high and low knowledge subjects across all three trials for all measurements.

These results indicate that breadth of experience with Macintosh applications transfers to learning a new application. Breadth of experience helped during all trials of the experiment, and there is a Table 2. Mean Performance Scores by Knowledge Level and Trial

Measurements / Knowledge Level	Experimental Trial 1	2	3
Action Times			
high	12.65	4.99	4.06
low	16.66	7.48	5.08
Instruction Times			
high	2.32	1.52	1.23
low	3.19	1.87	1.47
Correction Times			
high	1.30	0.53	0.64
low	1.98	0.93	0.84
Overall Completion Times			
high	16.27	7.04	5.86
low	21.88	10.31	7.38
Number of Hints			
high	4.89	0.44	0.06
low	5.96	1.39	0.42

trend indicating that it helped most during the exploration phase. There was also a trend towards less hints provided to more experienced subjects. All subjects needed a few hints (for about ten percent of the action steps), however more experienced subjects were faster in discovering new methods on their own, and were faster in the later trials.

Effects of Interface and Number of Steps

When performance during trial one was compared between the different interfaces, there were significant differences due to this variable. Five simple one-factorial between-

subjects ANOVA heeded the following results: $F(3, 69) = 4.36$, $p. < .01$ for overall completion times, $F(3, 71) = 4.30$, $p. < .01$ for action times, $F(3, 69) = 4.85$, $p. < .01$ for instruction times, $F(3, 71) = 7.38$, $p. < .01$ for correction times, and $F(3, 65) = 2.91$, $p. < .05$ for the number of hints. See Figure 8 for an illustration of these differences.

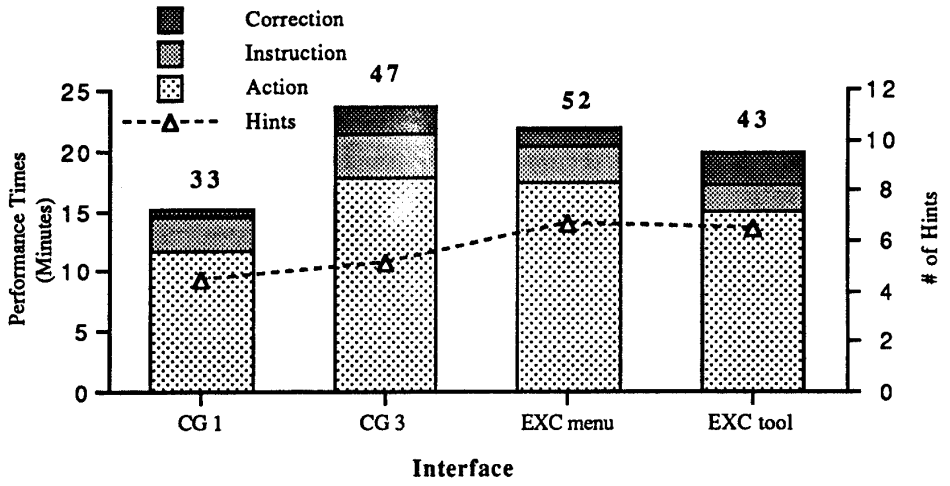


Figure 8. Performance Measures by Interface Condition during Task 1.

However, the four interfaces required different numbers of low-level actions, so that CGI for example, only required 33 steps but EXC menu required 47 steps for the whole task (see Figure 8). To control for the effect of the number of steps the overall performance times for each sub task were divided by the number of steps required for that sub task. These average times per action step were then multiplied by a constant and added across all subtasks. When the performance times were corrected for the number of action steps in this way, the differences between systems disappeared, $F(3, 69) = 1.45$, $p. > .10$ for overall completion times, $F(3, 71) = 1.66$, $p. > .10$ for action times, and $F(3, 65) = 1.58$, $p. > .10$ for the number of hints (see Figure 9). However, there were differences for instruction and correction times associated with different systems ($F(3, 69) = 5.45$, $p. < .01$ for instruction, and $F(3, 71) = 2.92$, $p. < .05$ for correction times). Gross performance time differences between interfaces can be explained to a large degree by the number of action steps that needed to be performed. The subjects seemed to have adopted different instruction reading strategies depending on which system they were using. Also, the different interfaces interacted with the experimental setup in different ways, so that more intervention was required in some cases (correction times). Action times therefore seem to be the most meaningful unit for interpretation of interface search times.

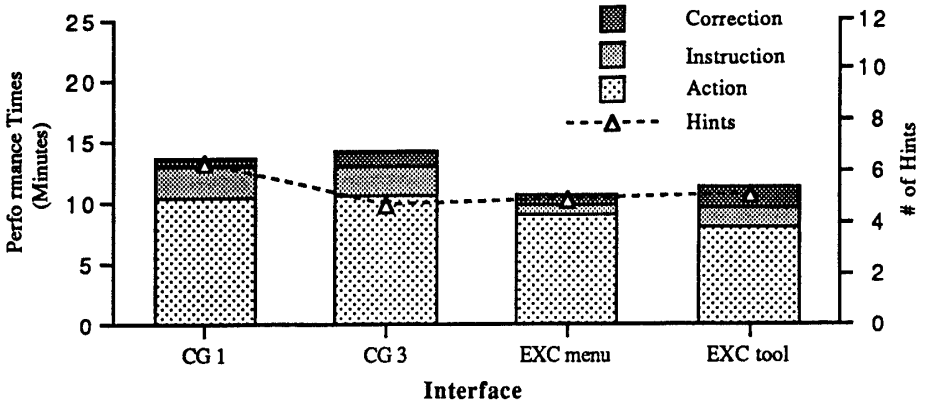


Figure 9. Corrected Performance Measures by Interface Condition during Task 1.

Effects of Delay

Finally, we investigated the effect of the delay on performance during the second task. Five mixed two-factorial ANOVA's (trial, repeated; condition, between-subjects) with a covariate controlling for Macintosh experience were performed. See Figure 10 for an illustration of the group means underlying these analyses.

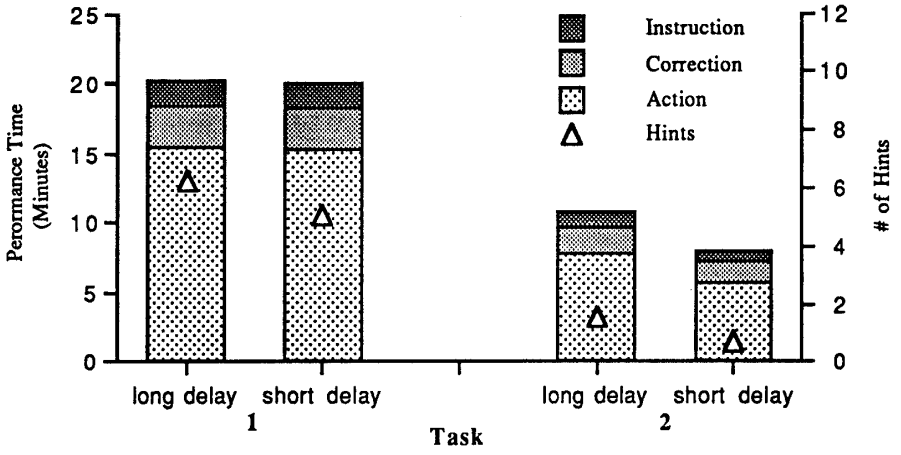


Figure 10. Performance Measures by Task and Delay Condition.

The interaction between trial and condition tests whether changes in the performance measures due to learning (trial) are dependent on delay condition. If there is a performance effect due to delay this interaction should be significant. In our analyses we found significant effects for this interaction for overall completion times ($F(1, 59) = 4.64$, $p. < .05$), for action times ($F(1, 62) = 6.33$, $p. < .05$), however not for instruction and

correction times ($F(1, 60) = 1.80$, $p. > .10$, and $F(1, 61) = .55$, $p. > .10$, respectively). There was also no significant interaction between trial and delay condition for the number of hints ($F(1, 54) = .15$, $p. > .10$). However this interaction may have been hidden by the considerable difference in hints received during the first trial (see Figure 10). If the variances of the number of hints of the two delay conditions on day two are compared in a simple ANOVA, this effect becomes significant ($F(1, 64) = 6.34$, $p. < .05$).

These results indicate that forgetting may play a role even for performance with display-based systems. If users have not used a new system for a longer time period, they need a longer time period to perform the same tasks, and they may have forgotten some methods completely, so that they require renewed external help. However, the observed forgetting effects are relatively small when compared to the large savings between the first and second trial and their significance could be debated on practical grounds. In a previous study that investigated performance on a transfer task, we had found that forgetting may play a role only in complex interactions, but not in simple ones (Franzke & Rieman, 1993). The analyses reported below investigate whether this result can be isolated from transfer effects in a simple learning design, and try to relate the difficulties in exploration and the observed forgetting effects to several design parameters.

Detailed Analyses and Results

If the current task-artifact analysis is to inform further designs of display-based applications, we need to refine our level of analysis. For any type of design recommendations we need to know which types of interactions were difficult to discover during the exploration phase (first task), and which interactions were responsible for the forgetting effects indicated in the global results.

Analysis by Subgoals

To answer these questions, the analysis was first taken to the level of subgoals. A previous study had indicated that some of the difficulties for retention may stem from the 'create graph' subgoal rather than from any of the graph modification subgoals (Franzke & Rieman, 1993). That study however, did not allow for detailed analyses of the interactions in the graph modification tasks.

For this level of analysis we will only report results for the action times and the number of hints. Action time measures the subjects' search time most cleanly. The number of hints provides important information about the explorability of an interaction, because it can be interpreted as an indicator for situations in which search of the display

alone would be unsuccessful and would require the use of other sources of information (manual, help-line, other users, etc.).

Two three-factorial mixed ANOVA's (trial, repeated; subgoal, repeated; condition, between) were performed on the corrected action time data; the means for this analysis are illustrated in Figure 11. We were interested in the interaction between trial and condition, which tests for delay effects across all goals. Furthermore we were interested in the triple interaction between trial, condition and subgoal, which tests whether differences in delay performance are dependent on particular subgoals. There was a significant interaction between trial and condition ($F(1, 65) = 5.75$, $p. < .05$), indicating the existence of overall delay effects. However, the triple interaction with subgoal was not significant ($F(9, 585) = .69$, $p. > .10$). There was a main effect for subgoal and an interaction of subgoal with trial, however, ($F(9, 585) = 55.77$, $p. < .01$, and $F(9, 585) = 24.06$, $p. <.01$, respectively). Overall action times and performance improvement due to practice vary with subgoal.

The same analysis was performed for the corrected number of hints. Again, there were significant effects of subgoal and the subgoal and trial interaction ($F (9, 513) = 27.46$, $p. < .01$, and $F(9, 513) = 17.10$, $p. < .01$, respectively), but no interaction between trial and condition or the

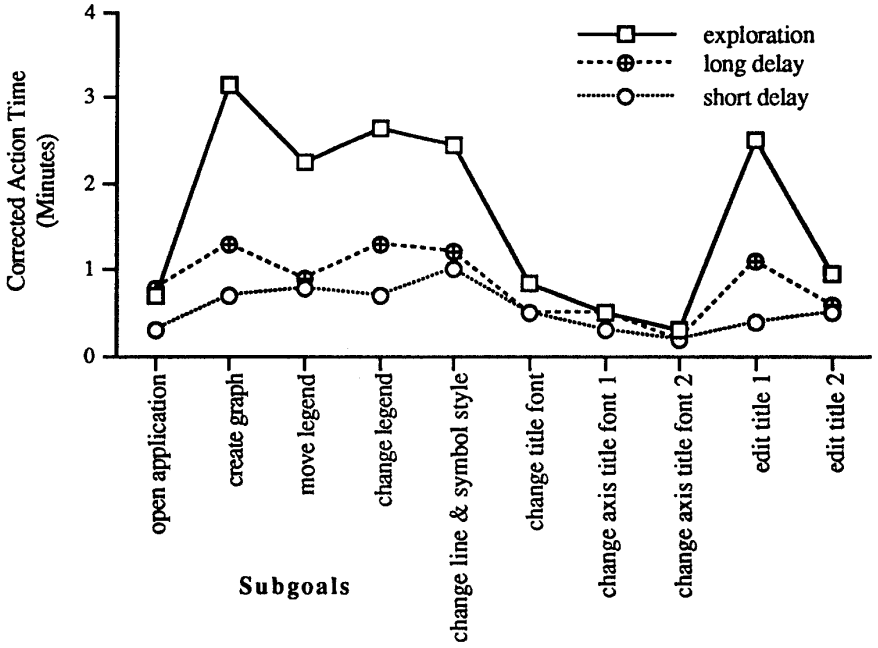


Figure 11. Corrected Action Times by Subgoal, Trial and Delay Condition.

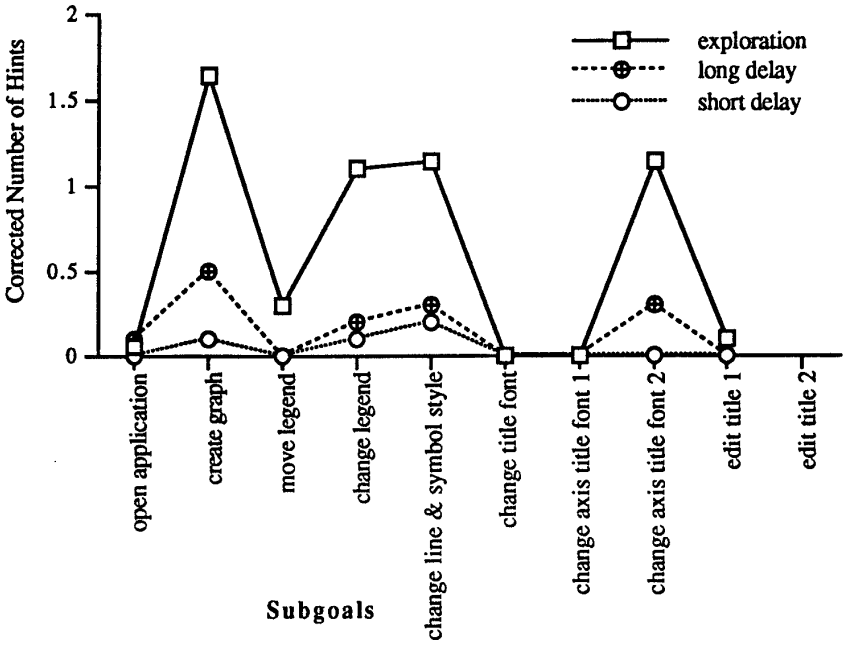


Figure 12 Corrected Number of Hints by Subgoal, Trial and Delay Condition.
triple interaction of trial, subgoal and condition (see Figure 12).

Since the triple interactions of goal, trial and delay condition did not reach significant levels, we also performed simple comparisons between the delay conditions for each of the subgoals, for corrected action times and the corrected number of hints during trial one and two. The results of these analyses are summarized in Table 3.

For the corrected action times and the corrected number of hints, there were no differences due to the delay condition during the first trial, as expected (the exploration task, where both groups received the same treatment). However, for the second trial, there were significant differences due to the delay manipulation for the corrected action times, in five cases. Three of these subgoals, create graph, change legend, and edit title introduced new methods. None of our subjects had ever used a graphing application, so *creating* of a graph established a new task and method. Similarly, subjects had to learn, in each interface, how to access text objects and how to change their style during the *change legend* subgoal. This method could then be used in subgoals six to eight, and there were no delay effects for these subgoals, with the exception of a small, but significant difference for subgoal seven. Similarly, subjects had to learn a new method for *editing* the content of text objects in subgoal nine, and this method could then be transferred to subgoal 10. There are two further exceptions to this rule: First, there was a strong delay effect for subgoal 1, where subjects had to open the data file and application by either

double-clicking the data icon, or by first double-clicking the application, and then the data icon. Even though all of our subjects new how to open files by double-clicking, several did not know that double-clicking a data-file also opens the application that it was created in. Therefore this subgoal introduced new knowledge to some of the subjects. The other exception is the subgoal *changing line and symbol style*, where performance on the second task was relatively poor for both conditions.

Table 3.
Results of simple ANOVA's comparing the corrected action times of two delay conditions during trial 1 and 2 (**significant with p. < .01, and *significant with p. < .05, ns = not significant, - = not available).

		Corrected Action Times		Corrected Number of Hints	
Subgoal		Trial 1	Trial 2	Trial 1	Trial 2
(1)	open application	ns	**	-	-
(2)	create graph	ns	**	ns	*
(3)	move legend	ns	ns	ns	-
(4)	change legend	ns	**	ns	ns
(5)	change line & symbol	ns	ns	ns	ns
(6)	change title font	ns	ns	ns	-
(7)	change axis title font 1	ns	*	-	-
(8)	change axis title font 2	ns	ns	-	-
(9)	edit title 1	ns	**	ns	*
(10)	edit title 2	ns	ns	ns	-

For the corrected number of hints, there were fewer observations, so that a comparison was not possible in many cases. Where they were possible they suggested the same interpretation as for the action times. Delay effects exist, when new methods had to be learned (subgoals two and nine), and these were also the situations in which many hints had to be given during the exploration trial (see Figure 12).

Analysis by Action Steps

To determine exploration and retention difficulties further, the analysis was finally taken down to the level of individual action steps. Figure 13 provides an example for action time variation on this level of detail, along the dimension of serial position, where each action was performed in succession.

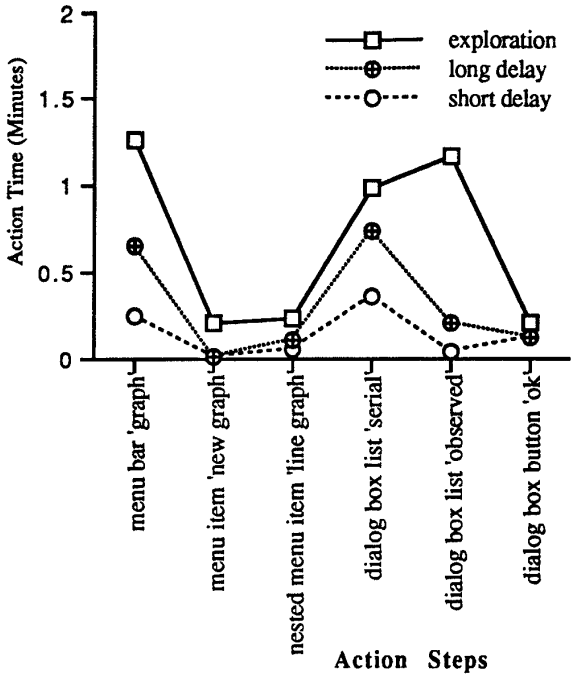


Figure 13. Action Times Associated with Single Action Steps for Graph Creation with CGIII.

Interpretation of specific action times for each action, subgoal and interfaces can be interesting to understand difficulties for particular situations, however it is difficult to see how this type of analysis could be generalized for design recommendations. Therefore, in the action step analysis, we decided to cut across the dimensions of interface and goal, to assess the overall effects of the three design dimensions introduced in the introductory chapter (type of basic-level interaction, semantic quality of the label, and number of objects on display).

In the remainder of this chapter, the coding scheme for this analysis will be detailed first. Then two sets of analyses will be reported: The first set concerns the effect of the design parameters on exploration times and on experienced use of display-based systems (overall action times from trial 2). The second set of analyses focuses on the influence of the design parameters on performance at delay.

Coding Scheme. For each action step (e.g. 'click on menu-bar item 'graph'') three variables were encoded: (1) What *type of basic-level action* was performed (e.g.: menu bar selection, button click, move operation, tool selection, etc.). (2) The *semantic quality of the label.* was determined as the semantic difference between the presumed representation of the task goal by the subject and the label linked to an interface object. We assumed that

subjects represented their active goal in terms of the task description that we provided to them. For example, if the assumed goal was to 'create a graph', then a menu item with the label 'graph' would have been defined as a semantic difference of 0, a menu item with the label 'chart' as a difference of 1 (for synonym), the label 'drawing tools' would have been assigned a value of 2 (semantically related, but inference required), and the label 'file' would have received a value of 3 (no direct semantic link, connection has to be learned). (3) We counted the number of *objects competing for attention.* Here, each object in the relevant object group was counted, as well as the number of competing object groups (e.g. for a menu selection: 6 competing menu items + 1 for the menu bar + 1 for the tool bar + 1 for the graph objects = 9). The protocols of forty randomly selected subjects were coded, for five from each experimental condition. See Appendix F for an example code.

Exploration Performance and Experienced Performance. The first set of analyses focuses on overall action time performance during trial one (exploration) and trial two (experienced performance).

One of the difficulties of using data collected in a 'naturalistic setting', rather than from an experimentally controlled design, is the possibility that some of the variables of interest may be confounded. To determine the degree to which the design parameters (type, semantic difference, and number of objects) may have been confounded, Pearson Correlation Coefficients were determined for all combinations of parameters (for the 'type' parameter, the action time means for each type were used in the calculations). The correlation coefficients were $r = .67$ for semantic difference and type, $r = .22$ for semantic difference and location, and $r = .20$ for location and type, indicating that semantic quality of the label and type of base-level interaction tended to covary quite strongly. The remaining combinations covaried only to a smaller degree.

Being warned about the problem of covariation between the design parameters, the parameters were used in a set of 'simple' regressions, where each modeled the effects of single design parameters. The first set of models determined their influence on overall action times. These models included the means of each subjects' performance as predictor and the parameters of interest were included as interactions with this subject parameter. A second set of models included the trial parameter and interactions with the trial parameter to test whether the influence of the design parameters varied with trial. Finally, two models tested triple interactions between the two sets of design parameters that were not strongly correlated, and trial. The statistical results from these three sets of analyses are included in Table 4. The interactions between the design parameters and the trials, as well as the triple interactions of pairs of design parameters and trial are illustrated in Figures 14 - 18.

Table 4.
Results from Regression Analyses of Action Times associated with Single Actions.

Predictor Variables	F-Values and Probabilities	R^2
Semantic Difference	F(1, 3404) = 482.87, p. <. 01	0.122
Number of Objects	F(1, 3404) = 60.95, p. < .01	0.02
Type of Interaction	F(16, 3420) = 31.25, p. < .01	0.125
Semantic Difference*Trial	F(1, 3404) = 112.34, p. < .01	0.03
Number of Objects*Trial	F(1, 3404) = 26.19, p. < .01	0.01
Type of Interaction*Trial	F(16, 3420) = 9.32, p. < .01	0.04
Sem*Obj*Trial	F(1, 3404) = 31.37, p. < .01	0.001
Obj*Type*Trial	F(11, 3404) = 5.83, p. < .01	0.02

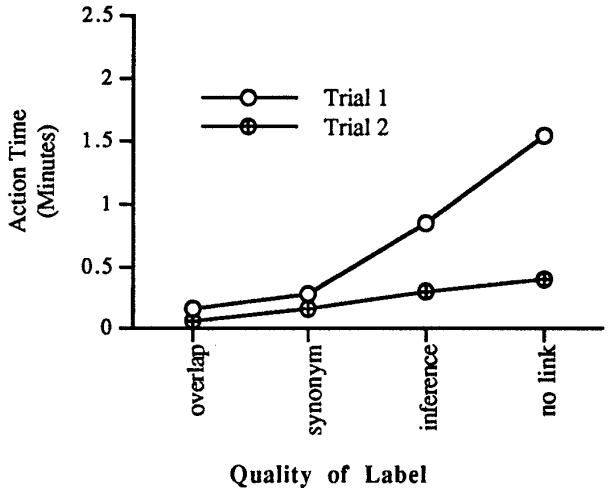


Figure 14. Action Times by Semantic Difference of Labels and Trial.

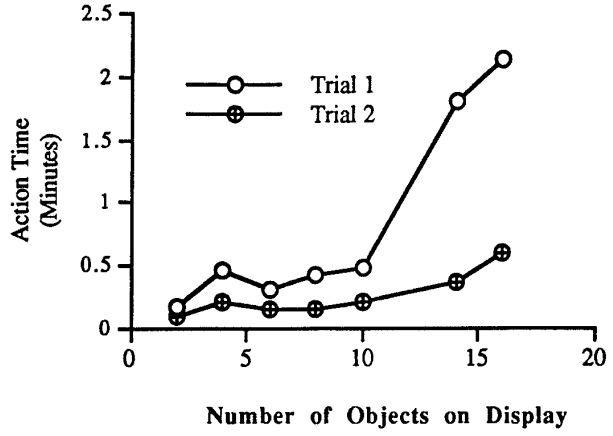


Figure 15. Action Times by Number of Objects on Display and Trial.

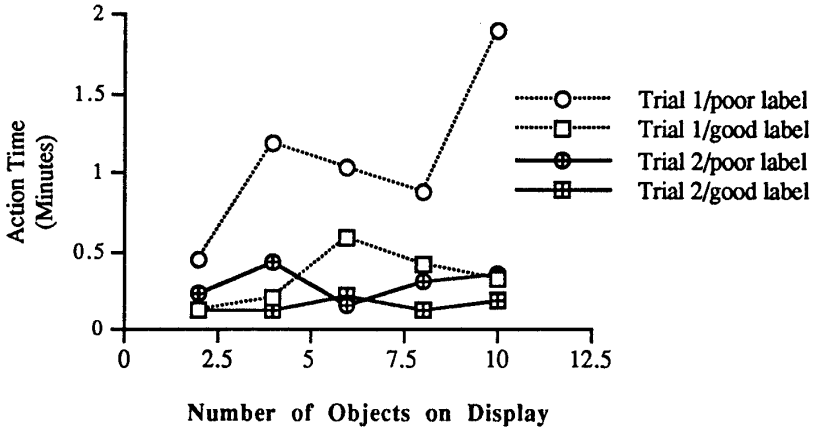


Figure 16. Action Times by number of Objects on Display, Semantic Difference and Trial.

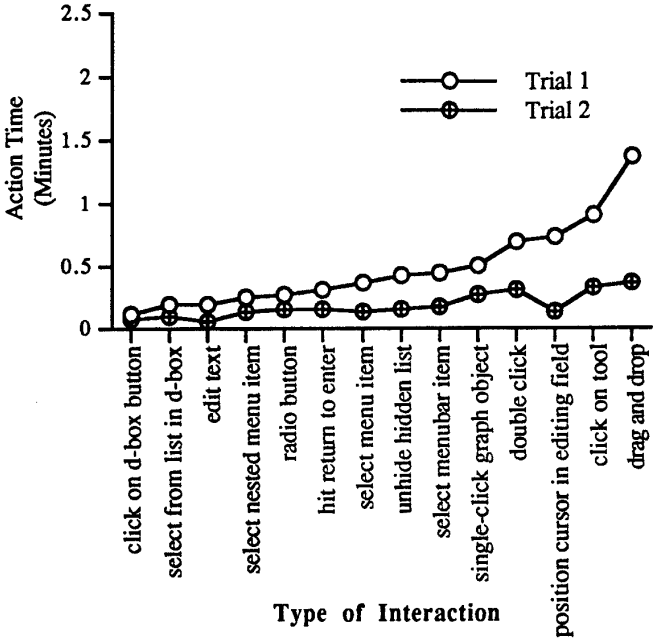


Figure 17. Action Times by Type of Interaction and Trial.

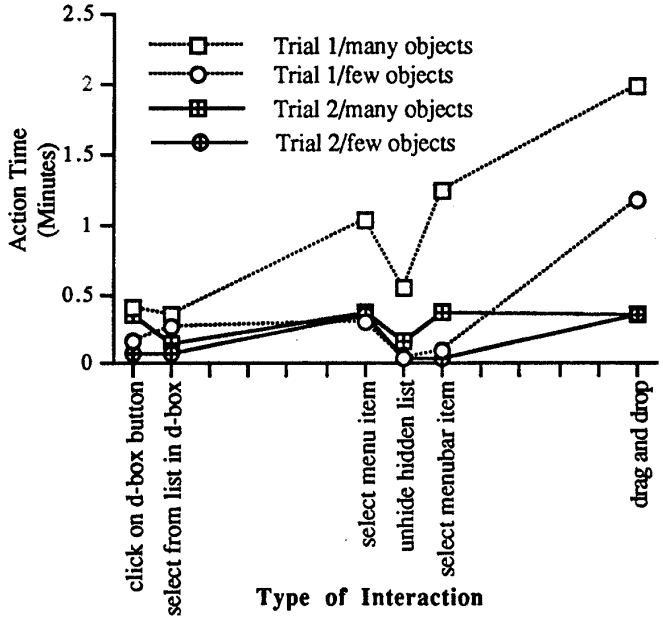


Figure 18. Action Times by Type of Interaction, Number of Objects and Trial.

These analyses demonstrate that the three coded design parameters indeed had the predicted effects on subjects' action time performance. Overall, there was an effect of semantic difference, so that action times were longer with a larger semantic difference between the labeling of an object and the subjects' representation of the goal. Similarly, the more objects were displayed at any given time, the longer the action times to access the correct object. However, in both cases these main effects interacted with trial, so that action times during the exploration trial were more affected than action times during the second trial (see Figures 14 and 15). The number of objects to search and the difficulties involved in evaluating potential operators both affect search time during exploration. During trial two, however, where subjects have access to knowledge of successful interactions, the number of objects to search and the goodness of the semantic match do not matter as much. Additionally, there is a three-way interaction between these two design parameters (number of objects and semantic difference) and trial, so that search time during trial one due to semantic difference is elevated additionally if there are many objects to search (Figure 16). If the semantic differences between labels and goals are small, there is no effect of the number of objects on display, and in fact, the action times during trial one do not seem to be very different from the action times during trial two.

There was also a significant effect of the type of interactions on action times that varied with trial. During trial one (exploration) some interactions are especially difficult to discover; Figure 17 displays the types of interactions in order of difficulty. Some of the more difficult interactions were dragging and dropping of items, clicking on tool icons, or double- and single-clicking on objects. On the low end of the action time spectrum are selections of buttons, selections of menu items and of lists in dialog boxes. None of the more difficult interaction are labeled in any meaningful way. All of the easier interactions however, are associated with labels. It seems that in searching a new application for methods, subjects strongly orient themselves towards reading of labels, rather than blindly trying direct manipulation operations on objects or icons. The order of difficulty of these interaction types also explains the strong correlation between semantic distance and interaction type that was reported in the beginning of this section.

Finally, we observed a triple interaction of type of interaction, the number of objects on display, and trial. If there are many objects to search, action times during trial one will be inflated especially for interactions that are difficult to discover. Figure 18 displays this effect for several interaction types. Unfortunately the combinatorial set of all action types with all numbers of display objects was not complete in our sample, another hazard of naturalistic observation.

To summarize these results: We found that the three coded design parameters,

semantic difference, number of objects on display, and type of interaction, had effects on action times. Action times during the exploration trial were affected more strongly than times during later performance (trial two). Once subjects had acquired a new action they seemed to be able to use this knowledge immediately in further interactions. Additionally, there are interactions between some of the design parameters. The pattern of these interactions suggests that subjects first search interface objects with labels, and only consider unlabeled choices (e.g. iconic tools) later. Second, well-labeled objects will be considered and found faster, and the number of objects in the display does not seem to affect search times for such situations. If the relevant object is labeled poorly, more items will be considered in depth, so that the number of objects to search will elevate the action times even further.

Delay Performance. In the analyses by subgoals we noted that performance on subgoals where new methods had to be discovered was more prone to performance decrease due to the delay manipulation. The following analyses investigate these effects on a lower level of detail. Only the actions from subgoals (3) create graph, (4) change legend, and (9) edit title 1, from trial two were included in these analyses.

Again, Pearson Correlation Coefficients between the three design parameters were calculated to estimate the degree with which they were confounded with each other. Here, the correlation coefficient for semantic difference and type of interaction were r = .66, for semantic difference and number of objects r = .62, and for type of interaction and number of objects r = .55, considerably higher than for the complete sample of actions. For this reason, tests for interactions between the design parameters were not possible, and separate regressions for the three design parameters (including interactions between the subject control variable, the condition variable and the variable of interest) will be reported. Table 5 summarizes the results from these analyses.

Table 5. Regression of Delay Differences on Design Parameters

Predictor Variables	F-Values and Probabilities	R^2
Semantic Difference*Condition	F(1, 567) = 62.94, p. <. 01	.17
Number of Objects*Condition	F(1, 567) = 53.40, p. < .01	.14
Type of Interaction*Condition	F(16, 572) = 5.35, p. < .01	.20

Again, we found strong effects of all three parameters on the differences between the two delay conditions. Figures 19 to 21 illustrate these effects. The same factors that lead to poor performance during search also lead to poorer performance after a week delay between the exploration and second session. In no case was this performance close to the long search times during exploration, however.

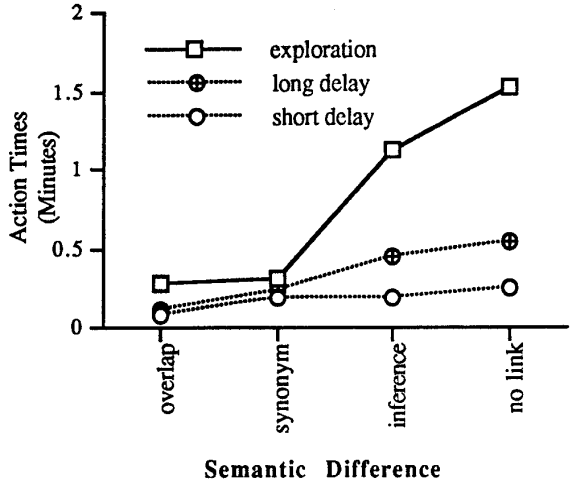


Figure 19. Action Time Differences Due to Delay by Semantic Difference (Exploration Times are Provided as Reference).

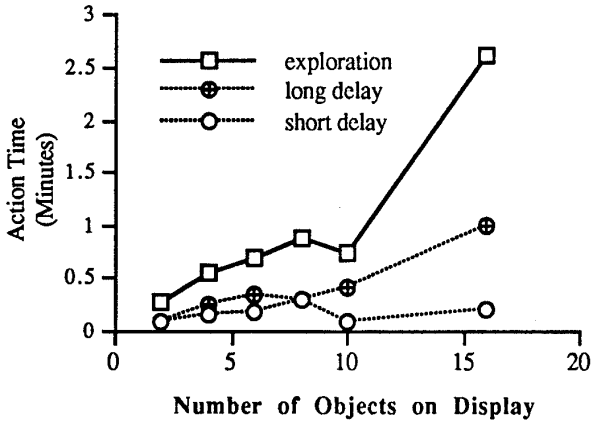


Figure 20. Action Time Differences Due to Delay by Number of Objects on Display (Exploration Times are Provided as Reference).

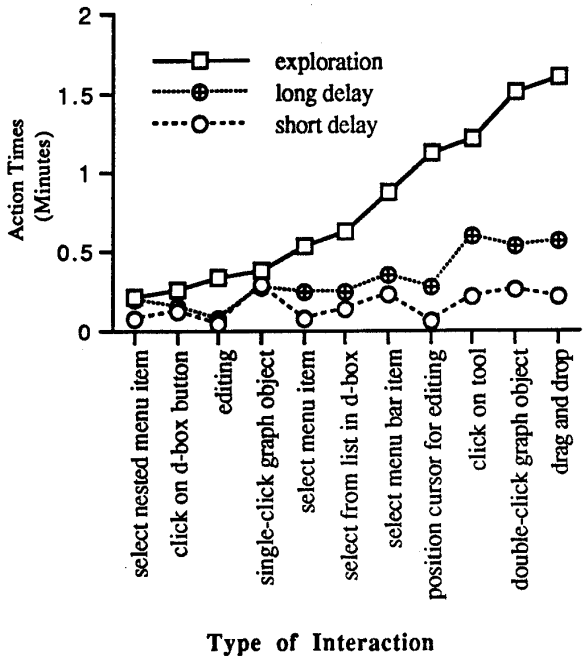


Figure 21. Action Time Differences Due to Delay by Type of Interaction (Exploration Times are Provided as Reference).

Table 6 . Regression of the Design Parameters on the Proportion of Hints.

Predictor Variables	F-Values and Probabilities	R^2
Data Set 1 (Performance on Task 1 and 2, all subgoals)		
Semantic Difference	F(1, 3421) = 232.66, p. < .01	.06
Number of Objects	F(1, 3405) = 45.21, p. < .01	.01
Type of Interaction	F(17, 3421) = 9.79, p. < .01	.05
Semantic Difference*Trial	F(1, 3421) = 48.54, p. < .01	.02
Number of Objects*Trial	F(1, 3405) = 10.35, p. < .01	.001
Type of Interaction*Trial	F(17, 3421) = 3.30, p. < .01	.02
SemDiff*Objects*Trial	F(1, 3405) = 13.84, p. < .01	.001
Type*Objects*Trial	F(11, 3405) = 3.91. p. < .01	.01
Data Set 2 (Performance on Task 2, Subgoals 3, 4, and 9)		
Semantic Difference*Condition	F(1, 573) = 44.12, p. <. 01	.13
Number of Objects*Condition	F(1, 568) = 32.80, p. < .01	.10
Type of Interaction*Condition	F(27, 573) = 3.51, p. < .01	.15

One final step in the analysis of the individual actions looked at the number of hints provided during exploration and in the two delay conditions. Recall that subjects were provided with a hint whenever they had not made progress towards the next correct action step in more than two minutes. For each action step, there is therefore a record of the proportion of subjects that had to be given a hint, an indication of when subjects would otherwise consult manuals or other sources of additional information. The same two overall sets of analyses as for the time data were conducted. The first set is done on the whole data set and investigates overall effects of the design parameters on the proportion of hints during the two trials. The second set of analyses focuses on trial 2 and the influence of the parameters on delay performance. Results from both sets of analyses are provided in Table 6.

These analyses on the proportion of hints replicate the findings from the action times perfectly. Long action times during exploration are representative of the difficulty of the search for the correct action. For difficult searches, hints were needed more often than when the correct action could be discovered easily. The design parameters therefore affect the number of hints provided in exactly the same way as the action times. Figures 22 to 24 illustrate these effects for the delay and the exploration trials.

Figure 22 replicates the finding from the action time analyses that difficulty of search is related to the goodness of the semantic match between the task goal and the label in the display. The more difficult the evaluation of a label, the more hints were provided to help subjects make the right selection.

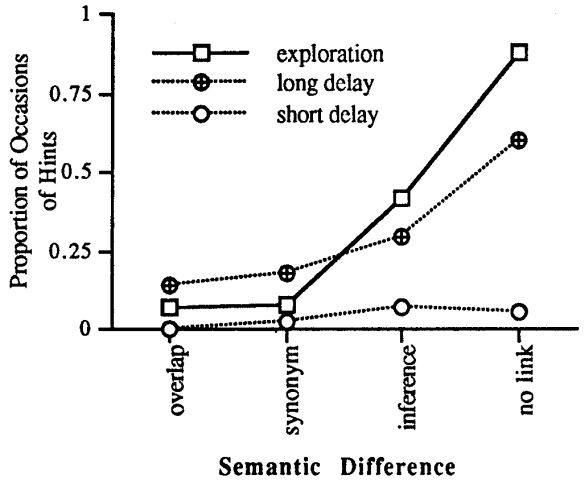


Figure 22. Proportion of Hints by Trial, Delay Condition and Semantic Difference.

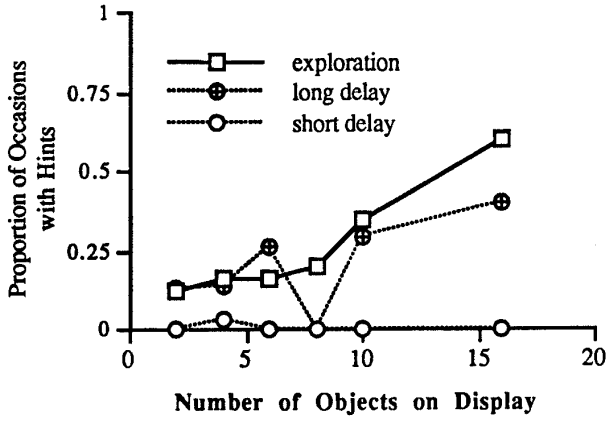


Figure 23. Proportion of Hints by Trial, Delay Condition and Number of Objects on Display.

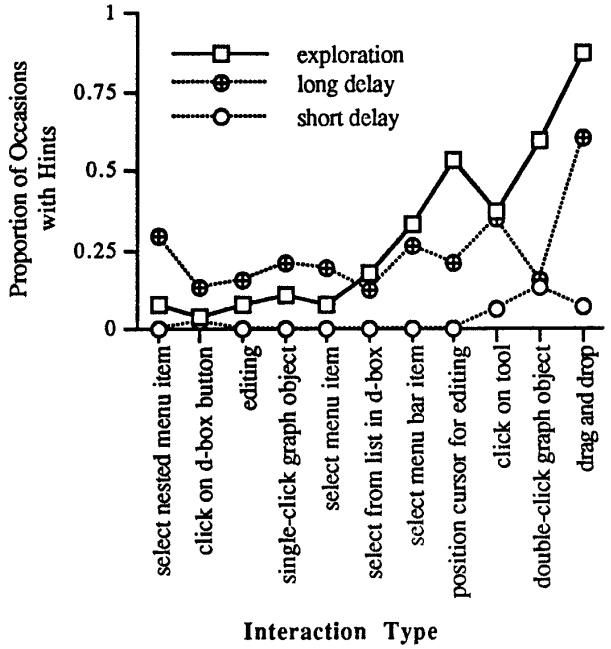


Figure 24. Proportion of Hints by Trial, Delay Condition and Interaction Type.

Figure 23 documents the finding for the number of objects on the display. When only few objects were displayed, fewer subjects needed hint than when many objects were on display. Furthermore, Figure 24 shows that interactions without label guidance (drag and drop, double-clicking, and tool selections) required more hints than actions that were guided by labels. The interactions reported in Table 6 show that as for the action times.

the effects of the design parameters interacted. If a semantic match to an object in the display was difficult (because no label was available, or the given label was inappropriate), the number of objects in the display increased the probability that a hint had to be provided.

For the differences between the delay conditions, the same effects apply. The greater the semantic difference, and the more objects to search, the greater the probability that subjects will also need a hint during the long delay trial. Unfortunately, the design of the experiment does not rule out a simple "generation effect" explanation of this result: Whenever a hint had to be given during exploration, it was also more likely that subjects needed a hint after a long delay interval. It is possible that subjects encoded self-generated methods with more cognitive effort than other-generated methods (e.g. Slamecka & Graf, 1978), and therefore were able to retain them better. This interpretation of the delay effect could be investigated more closely with a study that controlled for the provision of hints.

Summary

First, we found an effect of breadth of computer experience on task performance, so that subjects who had exposure to a variety of Macintosh systems tended to be faster during exploration and later trials, even though the number of hints they received was comparable to that given to less experienced subjects. It was also shown that the number of action steps necessary for performing the experimental tasks greatly influenced overall performance measures. The more action steps were necessary in completing the tasks, the longer the performance times and the more hints needed to be provided. Differences between systems were largely due to this variable, and once this effect was corrected for, the performance times for the four tested interfaces were comparable.

Second, and most prominently, strong learning effects were established for all measurements and at all levels of analysis. Our subjects were able to reduce their overall completion times by almost 60 percent from the first to the second trial, and a further improvement of 25 percent was observed between the second and third trial. Second, overall delay effects were demonstrated such that performing a second graphing task after a short (five minute) delay lead to significantly faster completion times than performance after a relatively long (one week) delay. However, overall this difference was relatively small (approximately 2.5 minutes) compared to the large difference between the exploration trial and second trial performance (approximately 11 minutes).

Third, we were able to show that exploration and delay performance varied by subgoal, and the results suggest that longer exploration times and poorer delay

performance are both associated with subgoals where new methods have to be learned. Analysis of action times and numbers of hints provided at the level of individual action steps showed that the type of interaction, the semantic match between label and goal, and the number of objects on the display all are reliable predictors of action times and the numbers of hints provided. Furthermore, all three design parameters interacted with trial, so that they influenced performance most clearly during the exploration trial. An analysis of the interactions between the parameters, where possible, suggested that the number of objects to search impairs exploration performance only when a simple semantic match between the label and the goal is not possible.

Finally, we found that the three design parameters also influence performance at a week-long delay. After a week delay, performance is worse (action times are longer, and more hints have to be provided) when interaction objects are not or poorly labeled, and when many objects have to be searched. However, it could be possible that these retention difficulties are mediated by a Generation Effect, where actions that were provided by outside help at are encoded less efficiently than actions that were self-discovered.

CHAPTER VI

DISCUSSION

How the Findings Inform Theory

Interface-Literacy Enables Exploratory Learning. First, the overall success with which our subjects could make use of a completely new system during the exploration trial support our knowledge analysis. Recall, that subjects needed only approximately 19 minutes and about six hints to complete the first graphing trial successfully. We assumed that the user population tapped in this experiment would already know basic Macintosh interface conventions and would therefore be literate in the language of menu selections, button presses, and direct manipulations of display-objects. We claimed that learning these basic interface conventions (basic-level action knowledge and background knowledge) is mediated by a phase of skill acquisition and had been confounded with learning of application-goal mappings in other studies (e.g. Carroll & Mazur, 1986, and Whiteside, et al., 1985). To determine whether exploratory learning of display-based interfaces was an option for discretionary, interface-literate users, these two phases of acquisition had to be separated. Our experiment achieved this goal by observing only users who had at least six months of experience, or knew at least one other Macintosh application. Our data show that exploratory learning was possible for our experimental subjects, as they accomplished a reasonably complex task (creating a graph and modifying several graph objects) in an appropriate time and could discover the large majority (90 percent) of the necessary action steps on their own. However, without provision of some external help none of the subjects, not even the more experienced ones, would have succeeded.

Additionally, we found an indication of a direct effect of Macintosh experience on task performance. When subjects were divided into a low-knowledge and a high-experience group according to the number of previously used Macintosh applications, high-knowledge subjects performed better overall, and there was a trend indicating that the differences were larger during the exploration trial. These differences could be due to several factors, namely better psychomotor mouse control, faster decision times, and better search strategies. If psychomotoric aspects would have played a great role in this difference, one would expect that it would have been measured better by the 'years of

Macintosh use' parameter then by the number of applications used. However, this parameter did not explain a significant amount of variance in the performance measurement. Therefore this component will be given less weight in the explanation of differences between the two knowledge groups. We argue that both faster overall decision times (possibly due to more general basic-action knowledge) and better search strategies may have helped the experienced subjects to perform better, since they were faster overall, but also had an extra advantage during exploration. To separate the effects of these components cleanly, experiments that control the factors independently would need to be conducted.

Label-Following as Search Heuristic. The detailed-level analyses from our experiment provide clear evidence that label-following is a main search heuristic during task-oriented exploration of a new interface. First, exploration times associated with different basic-level actions differed significantly. Our analyses showed that basic level-actions that are performed on labeled objects (dialog-box interactions, menu choices, etc.), were overall easier to discover than basic-level actions that are performed on unlabeled objects (direct-manipulation interactions, such as tool selections, single- and double-clicking of graph objects, etc.). This suggests that subjects may have a label-first strategy, and only consider unlabeled action choices in a secondary step. Second, actions associated with good labels were discovered faster than actions associated with poor labels. Third, when the labels were good, subjects discovered the relevant actions quickly, no matter how many objects had to be searched on the display. When the labels were poor, times to discovery and number of hints provided were larger overall, and were additionally inflated by the number of objects that had to be searched. This result suggests that subjects will first do a quick scan of display objects, looking for a good match between their goal and the labels offered on the display. If a good match exists, subjects will select it and repeat the same evaluation process for the next set of action possibilities. This observed behavior is compatible with a *scan-search* strategy described in Newell & Simon (1972, p. 826):

"A number of the most effective heuristic problem solving programs for computers make use of a *scan-search* strategy that is heavily dependent on backup capabilities. The essential ideas of the strategy are these ... : (1) enough information is kept about prior nodes so that any knowledge state previously visited can be recovered; (2) an evaluation of its promise is made of each knowledge-state from which search has not yet been undertaken; (3) each successive burst of search is made from the knowledge-state that is most promising among those available at the moment, and the new nodes created are then evaluated. ... Since the scheme would place heavy demands upon memory for backup information, its absence from the repertory of human problem solving programs that we encountered is a consequence of the structure of the human IPS (information processing system)."

One might add that its absence may have been a condition of the types of human problem solving that were observed in Newell and Simon's studies, namely problem solving in the context of cryptarithmetic, logic proofs, and chess. In our case of exploratory search of display-based systems, plenty of backup information was provided on the display. If a menu is dropped from the menu-bar, its location and the highlighting of the root-menu label provide redundant backup information. Even if a dialog box is opened, some backup information will be accessible (see the examples provided in Appendix C). Backup information is therefore readily accessible to the human IPS in this context, and the scan-search strategy can be used efficiently. However, this strategy breaks when the heuristic to quickly evaluate 'promising operators', in this case, label-following, breaks. In effect, the display functions as a type of *external memory* (EM). Simon and Newell (1972, pp. 800-803) claim that:

"An IPS with only a STM and LTM ... will behave very differently in a problem solving situation from an IPS, ..., that is also provided with external memories - paper, say, or a chessboard. ... A problem solving program cannot be specified independently of its EM any more than it can be specified independently of its internal STM and LTM. Hence, one can predict the problem solving program of an IPS only after characterizing the EM's available to it."

Early problem solving work (e.g., Atwood & Polson, 1976), concentrated on problem solving in situations where problem spaces and therefore backup information had to be represented primarily internally, in STM or LTM. In our situation, the display of the interface presents backup information and some of the previous selection history to users. The display therefore represents a dynamic *external memory*. Careful evaluation of each action is not necessary, because backup information is provided and most previous choices can be undone easily. In this situation, a human IPS with dynamic external memory, the scan-search strategy can be used efficiently. Because of the type of information displayed, a semantic match between the label and goal can be considered quickly to evaluate the promise of a problem state (or display-object). On the other hand, extensive planning of action sequences in the context of a new application is also not possible. A new user of a system simply doesn't know what problem states the selection of an operator will expand into. A quick forward search, guided by semantic similarity matches between the label and the goal, is therefore an efficient way to acquire that information.

If the label-following heuristic fails, and a good label-match does not 'pop-out' quickly, subjects will drop into a more detailed search, where each object on the display is considered more deeply. This will also lead to abandonment of a simple scan-search strategy. One alternative strategy observed in the context of our graphing tasks is the

'deepening attention' strategy, identified by Rieman (1994). Here it was observed that subjects would spend increasingly more time 'thinking' about individual menu items, when a simple label match could not be accomplished. Note that this search is one of memory, and not of problem states on the display, and seemingly directed at fixing the breakdown in the simple label-following heuristic.

In summary, we found evidence for a heuristic for display-based interaction, termed label-following, in which semantically promising (good) labels will be explored first, at each level in the interaction sequence. This heuristic is used by a strategy that drives search forward along a gradient of 'promising' operators, as long as such a promising operator can be identified quickly. If a search has ended without success, users can easily back up to higher levels in the search tree and consider alternative choices. This strategy is dependent on information about backup options, which are provided richly in the interface of display-based computer systems. In our case, the scan-search strategy surfaced in interaction of new users with a dynamic computer display (as external memory), but it can be expected that an IPS would make use of similar search strategies, whenever the problem solving environment can be used as dynamic external memory in a comparable way.

Experienced Performance and Learning. Our data analyses showed little evidence of influences of the type of basic-level action, semantic quality of the label, and number of objects on performance in a second trial, if the second trial was performed after a short delay (approximately five minutes). After only a single exploratory sequence of accomplishing a new complex task in a new application, subjects performed a second task without any signs of search behavior. Selection times for single actions after a brief delay are similar to selection times for 'good label' matches during the exploration trial. It seems that elaborative information that links 'poor labels' to the goal is readily available from LTM in this situation, so that the fast display-scan can suggest the correct action immediately. Slower exploratory search of alternative display objects can therefore be avoided. The current experiment and analyses do not allow any inferences about the nature of these elaborative memory traces. We suspect that they are episodic traces of previous interactions that are triggered by the rich contextual cues provided on the display. Previous work in other problem solving domains suggested that experts retain rich episodic traces of prior problem solving episodes (i.e., Chase & Simon, 1973; Egan & Schwartz, 1979, and Chiesi, Spilich, & Voss, 1979). Previous work in our domain suggests (Mayes, et al., 1988, Payne, 1991) that retrieval is extremely context dependent and deteriorates when no display-cues are provided. These studies mainly looked at recall of display-states by experienced software users. Our study adds to these results by

showing that build-up of new application-goal knowledge is very fast and efficient. Interface-literate users of a new graphing application showed no signs of forgetting (and repeated search behavior) after only a single exploratory episode with the application, if this episode had happened recently.

Forgetting after Long Retention Delays. Our experimental design also allowed us to map performance after longer time-delays (one week). Our analyses showed here that for subgoals where new methods needed to be established, performance showed renewed signs of the type of search behavior that had been observed during the exploration trial. However, even after a one-week delay between the exploration trial and the second trial, delay impairment was relatively minor compared to the long search times of the exploratory trial. Unfortunately it is difficult to determine the exact cause for the resurfacing of search behavior after a week of delay. There are two general classes of explanations for this effect. One explanation assumes that the forgetting happens uniformly for all cases of interactions. In this case the differences between the two delay conditions would be due to the fact that subjects simply have to search the same environment as during exploration, and therefore produce a similar behavior. On the other hand, it could be that retrieval failure is more likely in some cases than in others. Here, the forgetting effect may be attributed to a simple generation effect, where other-provided actions are encoded less deeply than self-provided actions (Slamecka & Graf, 1978). Or, as argued in the introduction and elsewhere (Kitajima & Polson, 1994), it could be that selection of poorly labeled action choices is dependent on a greater number of elaborative links and therefore more prone to retrieval failure. Unfortunately, with the current data analysis these three possible causes of the performance decreases at delay can not be distinguished. However, a more detailed analysis of the interactions and verbal protocols of the second trial performance is planned. It is possible that this will provide evidence for the type of memory traces that guide experienced performance, and helps to identify the exact cause of the renewed search behavior.

To summarize the delay findings: The action-level analyses show evidence of forgetting, such that interactions that were more difficult to discover are also more easily forgotten after a relatively long time delay (one week). Unfortunately, at this stage in the research, we can not determine the exact cause for this effect, or clearly separate it from repeated search behavior.

How the Findings Inform Designers

How can the current findings be used in the context of designing systems for use? We believe that the results bear on design and evaluation of display-based systems by offering direct empirical evidence for heuristic and theoretical evaluation criteria found in the literature (e.g. Nielsen & Molich, 1990, and Polson, et al., 1992). Furthermore, the results from this study bear on training issues.

Design Issues. Designing a new display-based application for use in today's work environments frequently involves design for users that are interface-literate and use display-based systems discretionary to support other tasks. Previous research, reviewed in the introduction, shows that users prefer to learn new systems by doing (Carroll, 1990, Rieman, 1994), and that they can be relatively successful with this strategy when given a concrete problem to solve (Charney & Reder, 1986) and when prevented from committing major mistakes (Carroll & Carrithers, 1984). One form of preventing major mistakes with a new interface is to provide new users with a simpler, down scaled version of a new software, the so-called training-wheels approach (Carroll & Carrithers, 1984; Franzke & Rieman, 1993). A more efficient way of handling this problem would be to design new systems in a way that the user stays on the direct solution path without any additional training implements. This would mean that the interface itself has to guide the exploratory behavior of the users so that it allows for optimal utilization of 'naturally occurring' search heuristics.

In this research we found evidence that subjects use a label-following heuristic to guide their search through a new interface. We also identified design parameters that complicate search based on this heuristic:

- Unlabeled direct manipulation techniques such as double-clicking on objects, tool selections, and mouse dragging for various purposes are difficult to discover. Subjects seem to investigate actions on labeled objects first (menus, labeled buttons, lists of options), and actions on unlabeled objects (icons, graph-objects, etc.) later.
- Good labels, which means labels that link the user's conceptualization of the task goal to an action, are essential in guiding the user towards the correct action choice (Polson, et al., 1992). For example, if a user thinks of 'drawing a graph', then any label directly linked to this representation, such as 'draw', 'graph', 'chart', will get his or her attention, but hiding the correct option under labels such as 'file' or 'new' will lead the user astray.
- If no label is provided (direct manipulation) or the label is a poor match to

the user's language or conceptualization, providing many other choices that are accessible at the same time makes success at finding the right choice especially unlikely. If a situation like this cannot be prevented, users will be forced to use external help (manuals, other users, etc.) to discover the correct action step.

- If an action or action sequence was difficult to discover in the first place, it also becomes a likely candidate for forgetting. In other words, designing an interface that will be easy to explore during first use also provides an interface that will be easy to handle for intermittent users.
- Finally, interfaces that require short action sequences for the completion of task goals will require less time during learning and use than systems that require long and baroque action sequences (Card, Moran, & Newell, 1983).

These findings could be added to other guidelines or evaluation techniques for computer interfaces. The Cognitive Walkthrough (Polson, et al., 1992), for example, provides a method to step through a set of benchmark tasks with an interface and check at each step for the goodness of the label match, the relation between the action necessary (for example a button press) and the instruction to that action provided on screen (for example a highlighted button), and the physical difficulty of that action. As background for this evaluation, the method provides heuristics to estimate the probability that users will have problems with the interface action. The results from the current study could provide clearer guidelines to estimate what types of actions will be difficult to find (the direct manipulation techniques identified), and what types of labels will present difficulties (synonyms will be sufficient, but not labels that require long inference chains on the part of the user). They also suggest that the evaluator needs to take the number of objects into account that would have to be searched if a simple-label following strategy fails. Probability of 'user-trouble' can be expressed as the combination of these design parameters, and a visual guide that expresses probability of 'trouble' as an effect of these factors could be provided to designers and evaluators (see Appendix G as a suggestion).

Training Issues. If a new application is developed for an organization, decisions about a proper introduction to users must be made. In the early days of office systems, new systems had to be introduced by training classes, because the systems were opaque to exploratory use and the great majority of users were not computer or interface literate. Today, many users already use other applications on the same platform and are therefore quite comfortable with the general class of display-based systems. Our results show that

interface-literate users can be expected to learn a system by exploration, given that they have a clear task goal, and given that some help is provided. It is unlikely that users will need a full-blown tutorial to step them through the functionality of a new system. However, help in form of manuals, on-line help-systems, or expert-users, should be available for the discovery of potentially troublesome steps.

Our data also open an interesting issue about learning with external help. We observed that subjects had greater trouble remembering interactions that they did not discover on their own. Unfortunately we were not able to decide whether this was because difficult interactions are difficult to remember, or because they were encoded differently when help was provided. It would be interesting to follow up on this issue to gain some further insight into the benefits and drawbacks of using external help versus self-explanation in the learning of a new system.

Methodological Issues

In this experiment, we opted for an empirical approach that avoids some of the pitfalls of controlled experimental design. We used real software, rather than limited functionality programs, constructed to satisfy experimental design constrains. Additionally we investigated performance in the context of a realistically complex task, rather than inducing an experimentally contrived task. On the other hand, we were interested in gaining theoretically interesting and generalizable results, which often have to be given up in such 'environmentally correct' settings.

This methodological conflict has a long tradition in the cognitive sciences, as well as in Human-Computer Interaction. It concerns the tradeoff between methodological and theoretical rigor of controlled experimentation and the questionable applicability of results gained in contrived environments to cognitive functions in the service of complex goals and complex task environments. (Simon, 1967; Miller, Polson, & Kintsch, 1984; Suchman, 1987; Carroll, 1990; Norman, 1993).

With respect to research in the area of HCI these concerns have been expressed recently by the participants of the Kittle House workshop (Carroll, 1991). One central problem pointed out by these authors is that methods based on formal experimental design may force dichotomies that are unlikely to be found in the design space of real systems (e.g., Barnard, 1991) and that experiments based on such artificial design decisions may also bias the results to favor a particular feature or interaction style. We have provided an example for such a study in the introduction (Bensabat & Todd, 1993). In that study clean factorial control was exerted over the dimensions of text vs. icon and menu-use vs. direct

manipulation. However, the interfaces derived to satisfy the experimental combinatorics only supported a very small set of tasks, and simply did not represent realistic design choices for interface design. Therefore, 'general' results from such studies can not easily applied to realistic settings.

Additionally, experimental tasks often do not represent realistic interface interactions. For example, complex search in service of vaguely defined task goals is often reduced to a visual search task in the context of clearly defined, experimentally induced task goals. In the section reviewing the empirical literature on menu and icon design, we discussed several of these studies (e.g., Perlmann, 1984, Norman, 1990, and Card, 1982, Byrne, 1993). Interesting and important characteristics of use of display-based systems, however, almost always involves search in the service of global and fuzzy task goals (see similar criticisms by Barnard, 1991, Carroll, 1991, and Halgren & Cooke, 1993).

Finally, observations collected in a contextually deprived laboratory situation often do not capture variations of everyday use, such as social aspects, multitasking or performance-reducing variables of the environment such as noise level or time pressure (Monk, Nardi, Gilbert, Mantei, & McCarthy, 1993).

On the other hand, global comparisons between systems on the basis of overall user performance during a realistic task often do not inform about which particular features of a design supported or hindered task completion. Comparing design A with design B may inform a design team about which general design to pursue, but rarely produces principled design rules that could be applied to the design of other applications (e.g. Carroll, 1990). In our literature review the studies by Carroll & Mazur (1986) and Whiteside et al. (1985) can count as examples. Even though they clearly pointed at the problem of the learnability of the Lisa (as it was provided to the user at the time), and two 'iconic systems', some of the important issues raised are confounded with others. In the Lisa study, for example, the difficulty of acquiring the Macintosh interaction style ('base-level actions' in our terminology) is confounded with the poorly designed manuals and training materials provided with the Lisa. Studies like this can raise interesting issues, however, that could be followed up in more detailed research.

Studies conducted in rich contexts using holistic analysis thus may produce anecdotal evidence for or against the use of particular interface features. However, often such claims are critically dependent on their particular context of use, and it is difficult to see how they generalize to other contexts (Payne, 1991). Furthermore, it may be difficult to find enough instances of a particular phenomenon of interest in the field if no attempt of behavior induction has been made (e.g. Rieman, 1993).

This constitutes the contradiction that formal experimental methods may produce

strong and general results that may not necessarily be applicable to any real-world situation; whereas less formal observational studies may produce results that are applicable to a very small set of design decisions, without being generalizable.

The current study was an attempt to bridge this discrepancy by combining methods of both approaches. New users of four commercial graphing systems were observed accomplishing a series of graphing tasks. Even though the tasks were experimentally induced, an effort was made to represent a realistic first use of a graphing system. Furthermore, the task goals were induced in as indirect and 'fuzzy' a manner as possible (with the use of examples), while still insuring comparability of interactions between subjects (sequencing of the tasks). The experiment was also designed to produce a long and a short retention interval to induce behavior modification due to this variable. The subjects' performance was analyzed at both a global and a very detailed level. Users' performance in the service of complex goals was dissected to a detailed level of single button-clicks and menu-selections. At this detailed level, interactions could be analyzed with respect to design dimensions that were embodied in the actual applications, rather than in artificially induced experimental designs. This approach was aimed at generality by insuring applicability of the results. What, then are the gains and payoffs that came with this type of empirical approach?

Applicability

The tasks observed in this experiment were at a level of complexity of a task that a new user of a graphics application would be likely to attempt. Users created a new graph and attempted a series of modifications constrained only by a list of tasks to accomplish and an order in which to accomplish them. While we did not rely on observations of 'naturally occurring' events, neither did we introduce an artificially reduced experimental task, in which all but variations on one critical dimension were controlled, or where subjects were submitted to a rigid training regime. Instead subjects were given freedom to explore the interface at their pace and using their own strategies. Secondly, while we artificially created a situational variable (the delay conditions) that would have been difficult to observe in a more natural environment, we relied on the design space of the chosen interfaces to provide the parameters for analysis. The latter choice prevented the introduction of artificially introduced dichotomies that would not likely represent actual design decisions.

On the other hand, our experiment did control the sequence of tasks to be accomplished, and probably introduced some experimental side-effects that may not represent the use of a new system in a realistic setting, for example, the knowledge that

one's interactions are captured on video, and the verbalization of thoughts. One could also argue that the experimenter-provided hints may have introduced a form of context-sensitive help that is not quite available yet with existing systems. While we generally agree that our experimental situation differed from say, the use of the same system in a noisy office environment, where the user is under time pressure and does not have access to external help, we believe that the experimental results can be extrapolated to make predictions about such cases. In the office situation, for example, it can be expected, that the difficulties that were encountered by our experimental subjects will be magnified, and it will be likely that the interaction sequence totally breaks in situations where external help had to be provided.

Generalizability

On the other hand, one could question the generalizability of the results. First, users were investigated in the context of particular tasks, and in a particular application domain. Do the results generalize to, say, the use of drawing tools or desk-top publishing systems? We believe they do, for the following reasons: First, the graphing systems and tasks used here represent a broad range of interactions, such as some direct manipulation actions, and a broad range of menu-based and dialog-box-based interactions. The same combination of interaction types can be found in most other display-based office systems. Unless there is reason to expect that a particular tasks (creating a graph versus modification of text for example) interacts with general search behavior so that the label-following heuristic is modified, we would expect that the label-following heuristic is general. We have some evidence to support this argument: In statistical analyses reported elsewhere (Franzke, 1994) we found that the inclusion of 'goals' as additional parameter in multiple regressions models, did not explain more variance in the action times over and above what had been explained by design parameters. We therefore have reason to believe that the design parameters and not the tasks themselves were responsible for the differences found. However, it may be that work in a domain that calls for less verbal representations of goal states than graphing or editing (for example drawing, or graphic design) would make use of the label-following strategy less likely and therefore produce different types of search behavior. In such spatially represented domains direct-manipulation interactions may be favored over interactions with labeled menu choices or buttons.

Second, since the parameters for the analysis were taken from the commercial designs used in the study, rather than from an evenly constructed space of logically possible combinations, we had to deal with the fact that some parameters were confounded, and not all combinations along their dimensions could be observed. This

makes conclusions from the analysis less strong than if they had been obtained in more ideal conditions. To remedy this problem, more controlled studies could be run, and results from those studies in combination with the results reported here could be used as 'converging evidence' that the results are not an artifact of the experimental situation.

Usability

Would this general approach be usable in the investigation of other phenomena within HCI? We can easily imagine the use of this method for the investigation of other design dimensions, such as the use of color, the location of action objects on the display, size parameters, shape parameters, etc. Furthermore, we believe that the approach could be useful to determine the impact of design decisions in different types of systems, for example parameters associated with voice-dialog systems (such as message length, branching, numbers of options). However, there are certain limitations: First, the artifacts/interfaces investigated should be chosen to span a large design space. The results gained may be relative to that chosen design space and therefore conservative. Recommendations from such studies are not likely to suggest novel designs or introduce new methods, but will evaluate the usability of methods and designs already in use. A related second point is that recommendations gained from such studies will be *late* with respect to the development of new technology. They would be based on the accessibility of several artifacts within one general design space that support reasonably complex tasks. This state of affairs is usually reached when technology has already ripened to some degree. Finally, studies of this kinds are extremely cost intensive. Analyzing a reasonable large corpus of behavior at a low level of analysis comes at a high time investment. However, new interface technology may make the use of video-protocols and transcriptions obsolete, by allowing for the collection of interpretable data from hierarchical user histories (Kosbie, 1994).

Conclusions

The current study presents new insights into the nature of display-based interaction, namely that learning by task-oriented exploration is possible for interface-literate users, both in terms of the initial learning phase, as well as the use of this knowledge in later trials. We showed that knowledge established in only one exploration trial can be used to prevent search behavior in a second trial, if there is no long time delay between these two episodes. We showed that forgetting plays a relatively small role in the use of display-based systems, and pointed to some of its possible causes. Furthermore, we presented

evidence for the use of a search heuristic in the context of display-based systems, called label-following (Engelbeck, 1986, and Polson, 1988). We showed that this heuristic may be used in a search strategy by which the search space is scanned for a promising operator and search continues along the gradient of most promising operators. We identified three design flaws, the use of unlabeled direct manipulation interactions, use of poor labels, and provision of a large set of possible actions, that impair the use of the label-following strategy, and therefore complicate discovery of the correct actions. We suggested possibilities for application of the results in guidelines for evaluation of display-based interfaces. Finally, we discussed the choice of the current methodology in the context of the claims for generalizability and applicability of experimental design, and evaluated some of the benefits and tradeoffs of the chosen methodology chosen.

BIBLIOGRAPHY

Anderson, J.R. (1987). Skill acquisition: compilation of weak-method problem solutions. Psychological Review, 94, 192-210.

Apperley, M.D., & Field, G.E. (1985). A comparative evaluation of menu-based interactive human-domputer dialogue techniques. Proceedings of INTERACT '84, Elsevier: North-Holland, 323-328.

Atwood, M.E., & Polson, P.G. (1976). A process model for water jug problems. Cognitive Psychology, 8, 191-216.

Ballas, J.A., Heitmeyer, C., & Pérez, M. A. (1992). Evaluating two aspects of direct manipulation in cockpits. Proc. ACM Human Factors in Computing Systems CHI'92, Monterey, CA, ACM: 127-134.

Barnard, P. (1991). Bridging between basic theories and the artifacts of HCI. In Carroll (Ed.): Designing Interaction. Psychology at the Human-Computer Interface. Cambridge, MA: Cambridge University Press, 103-127.

Bensabat, I. & Todd, P. (1993). An experimental investigation of interface design alternatives: icon vs. text and direct manipulation vs. menus. International Journal of Man-Machine Studies, 38, 127-134.

Benson, C.R., Govnindaraj, T., Mitchell, C.M., & Krosner, S.P. (1989). Effectiveness of direct meanipulation interaction in the supervisory control of FMS part movement. Proceedings IEEE International Conference on Systems, Man and Cybernetics. Cambridge, MA, 947-952.

Byrne, M.D. (1993). Using icons to find documents: Simplicity is critical. Proceedings of InterCHI '93 Conference in Human Factors in Computing Systems, Amsterdam, Holland: ACM, 446-453.

Card, S., Moran, T., and Newell, A. (1983). The Psychology of Human-Computer Interaction. Hillsdale, New Jersey: Lawrence Erlbaum.

Card, S.K. (1982). User perceptual mechanisms in the search of computer command menus. Proceedings of Human Factors in Computer Systems, Gaithersburg, MD, SIGCHI, 190-196.

Carroll, J.M. (1988). Modularity and naturalness in cognitive science. Metaphor and Symbolic Activity, 3, 61-86.

Carroll, J.M. (1990). Infinite detail and emulation in an ontologically minimized HCI. In Proceedings CHI '90, Seattle: ACM, 321-327.

Carroll, J.M. (1990). The Nurnberg Funnel: Designing Minimalist Instruction for Practical Computer Skill. The MIT Press: Cambridge, MA.

Carroll, J.M. (1991). Introduction: The Kittle House Manifesto. In J.M. Carroll (Ed.): Designing Interaction: Psychology at the Human-Computer Interface. Cambridge, MA: Cambridge University Press, 1-16.

Carroll, J.M., & Carrithers, C. (1994). Blocking learner errors in a training wheels system. Human Factors, 26, 377-389.

Carroll, J.M., & Mazur, S.A. (1986). LisaLearning. IEEE Computer, 91, 35-49.

Carroll, J.M., & Rosson, M.B. (1987). The paradox of the active user. In J.M. Carroll, (ed.), Interfacing thought: Cognitive Aspects of Human-Computer Interaction. The MIT Press: Cambridge: MA.

Charney, D. & Reder, L. (1986). Designing interactive tutorials for computer users. Human-Computer Interaction, 2, 297-317.

Charney, D., Reder, L., & Kusbit, G. (1990). goal setting and procedure selection in acquiring computer skills: a comparison of tutorials, problem solving, and learner exploration. Cognition and Instruction, 7, 323-342.

Chase, W.G. & Simon, H.A. (1973). Perception in chess. Cognitive Psychology, 5, 55-81.

Chiesi, H.L., Spilich, G.J., & Voss, J.F. (1979). Acquisition of domain-related information in relation to high and low domain knowledge. Journal of Verbal Learning and Verbal Behavior, 18, 257-273.

Eberleh, E., Korfmacher, W., & Streitz, N.A. (1992). Thinking or acting? Mental workload and subjective preferences for a command code and a direct manipulation interaction style. International Journal of Human-Computer Interaction, 4, 105-122.

Egan, D.E. & Schwartz, B.J. (1979). Chunking in recall of symbolic drawings. Memory and Cognition, 7, 149-158.

Engelbeck, G. (1986). Exceptions to Generalizations: Implications for Formal Models of Human-Computer Interaction. Unpublished Masters Thesis: University of Colorado, Boulder CO.

Ericsson, K. A. & Simon, H.A. (1984). Protocol Analysis: Verbal Reports as Data. Cambridge, Mass.: MIT Press.

Field, G.E., & Apperley, M.D. (1990). Context and selective retreat in hierarchical menu structures. Behaviour and Information Technology, 9, 133-146.

Fischer, G. (1991). Supporting learning on demand with design environments. Proceedings of the International Conference of the Learning Sciences, Evanston, IL, 165-172.

Franzke, M. & Rieman, J. (1993). Natural Training Wheels: Learning and Transfer between two Versions of a Computer Application. In T. Grechenig & Tscheligi (Eds.). Lecture Notes in Computer Science: Human Computer Interaction. Vienna Conference VCHCI. Berlin, FRG: Springer-Verlag.

Franzke, M. (1994, in preparation). Task-artifact analysis of display-based interaction: Gaining general results from complex task scenarios. To be submitted to CHI '95 Conference in Computing Systems.

Franzke, M. (in press). Transfer kognitiver Fertigkeiten. In: Kintsch, W. & Hoffman, J., (Eds.). Enzyklopädie der Psychologie, Band C/II/8 - Lernen und Üben. Göttingen, FRG: Hogrefe.

Frese, M., Schulte-Göcking, H., & Altmann, A. (1987). Lernprozesse in Abhängigkeit von der Trainingsmethode, von Personenmerkmalen und von der Benutzeroberfläche. In W. Schönpflug, M. Wittstock (Eds.): Software Egonomie '87, Berlin: Teubner, 377-386.

Halgren, S.L. & Cooke, N.J. (1993). Towards ecological validity in menu research. International Journal for Man-Machine Studies, 39, 51-70.

Howes, A. & Payne, S. (1990). Display-based competence: toward user models for menu-driven interfaces. International Journal of Man-Machine Studies, 33, 637-655.

Howes, A. & Young, R.M. (1993). A learning and performance model of display-based task action mapping. Manuscript under review.

Howes, A. (1993). Recognition-based problem solving. Proceedings of the 15th Meeting of the Cognitive Science Society, Boulder, Colorado, 551-556.

Howes, A. (1994). A model for the acquisition of menu knowledge by exploration. In Proceedings of CHI '94, Boston, MA: ACM, 445-451.

Hutchins, E.L., Hollan, J.D., & Norman, D.A. (1986). Direct manipulation interfaces, in D.A. Norman & S.W. Draper (eds.), User Centered System Design: New Perspectives on Human-Computer Interaction. Hillsdale, N.J.: Erlbaum, 87-124.

Kacmar, C.J. & Carey, J.M. (1991). Assessing the usability of icons in user interfaces. Behaviour & Information Technology, 10, 443-457.

Kieras, D.E., and Polson, P.G. (1985). An approach to the formal analysis of user complexity. International Journal of Man-Machine Studies, 22, 365-394.

Kintsch, W. (1988). The role of knowledge in discourse comprehension: A construction-integration model. Psychological Review, 95, 163-182.

Kitajima, M. & Polson, P. (1992). A computational model of use of a graphical user interface. In Proceedings of CHI '92, Monterey, CA: ACM, 241-249.

Kitajima, M. & Polson, P.G. (1994a). A comprehension-based model of correct performance and errors in skilled, display-based human-computer interaction. Technical Report # 94-02, Institute of Cognitive Science, University of Colorado.

Kitajima, M. & Polson, P.G. (1994b). A model-based analysis of errors in display-based HCI. Conference Companion CHI '94, Boston, MA, 301-302.

Kosbie, D. (1994). Hierarchical events in graphical user interfaces. Presentation given at CHI '94 doctoral consortium, Boston, MA.

Kurtenbach, G.P., & Buxton, W.A.S. (1993). The limits of expert performance using hierarchic marking menus. Proceedings of InterCHI '93 Conference in Human Factors in Computing Systems, Amsterdam, Holland, 482-487.

Kurtenbach, G.P., Sellen, A.J., & Buxton, W.A.S. (1993). An Empirical evaluation of some articulatory and cognitive aspects of marking menus. Human-Computer Interaction, 8, 1-23.

Landsdale, M.W., Simpson, M., & Stroud, T.R.M. (1990). A comparison of words and icons as external memory aids in an information retrieval task. Behaviour & Information Technology, 9, 111-131.

Lee, E., McGregor, J. (1985). Minimizing user search time in menu retrieval systems. Human Factors, 27, 157-162.

MacLeod, M., & Tillson, P. (1990). Pull-down, holddown, or staydown? A theoretical and empirical comparison of three menu designs. Proceedings of INTERACT '90, Elsevier, North-Holland, 429-434.

Mannes, S.M. & Kintsch, W. (1991). Planning routine computing tasks: Understanding what to do. Cognitive Science, 15, 305-342.

Mayes, J.T., Draper, S.W., McGregor, A.M., & Oatley, K. (1988). Information flow in a user interface: the effect of experience and context on the recall of MacWrite screens. In Jones, D.M, Einder, R. (Eds): People and Computers IV, Cambridge UK: Cambridge University Press, 191-220.

McDonald, J.E., Stone, J.D., & Liebelt, L.S. (1983). Searching for items in menus: the effects of organization and type of target. Proceedings of the Human Factors Society 27th Annual Meetings, Santa Monica, CA: Human Factors Society, 834-837.

Merkin, M. (1983). In love with lisa. Creative Computing, 23, 12-27.

Miller, D.P. (1981) The depth/breadth tradeoff in hierachical computer menus. Proceedings of the Human Factos Society 25th Annual Meeting, Santa Monica, CA: Human Factors Society, 296-300.

Miller, J.R., Polson, P.G., & Kintsch, W. (1984). Problems of methods in cognitive science. In Kintsch, Miller, Polson (Eds.): Methods and Tactics in Cognitive Science. Hillsdale, NJ: Lawrence Erlbaum, 1-18.

Mills, Z. & Prime, M. (1990). Are all menus the same? - An empirical study. Proceedings of INTERACT '90, Elsevier, North-Holland, 423-427.

Monk, A., Nardi, B., Gilbert, N., Mantei, M., & McCarthy, J. (1993). Mixing oil with water: Ethnography vs experimental psychology in the study of computer-mediated communication (Panel). In Proceedings INTERCHI '93, Amsterdam: ACM, 3-6.

Newell, A. & Simon, H.A. (1972). Human Problem Solving. Englewood Cliffs, N.J.: Prentice-Hall.

Newell, A. (1980). Reasoning, problem-solving, and decision processes: the problem space as fundamental catergory. In R. Nickerson, (ed.), Attention and Performance VIII. Hilldale, N.J.: Erlbaum Associates.

Nielsen, J. (1990). Miniatures versus icons as visual cache for videotex browsing. Behaviour & Information Technology, 9, 441-449.

Nielsen, J., & Molich, R. (1990). Heuristic Evaluation of user interfaces. In Proceedings CHI '90, Seattle, WA: ACM, 249-256.

Norman, D.A. (1993). Cognition in the head and in the world: An introduction to the special issue on situated action. Cognitive Science, 17, 1-6.

Norman, K.L. (1991). The Psychology of Menu-Selection. Designing Cognitive Control at the Human/Computer Interface. Norwood, NJ: Ablex.

O'Hara, K.P., & Payne, S. (1994). Cost of operations affects planfulness of problem-solving. Manuscript under review.

Paap, K.R., & Schulte-Hofstrand, R.J. (1986). The optimal number of menu options per panel. Human Factors, 28, 377-385.

Parkinson, S.R., Hill, M.D., Sisson, N., & Viera, C. (1988). Effects of breadth, depth, and number of respnses on computer menu search performance. International Journal of Man-Machine Studies, 28, 683-692.

Payne, S. & Green, T.R.G. (1986). Task-action grammar: a model of the mental representation of task languages. Human-Computer Interaction, 2, 93-133.

Payne, S. (1991). Display-based action at the user iterface. International Journal of Man-Machine Studies, 35, 275-289.

Payne, S. (1991). Interface problems and interface resources. In Carroll (Ed.): Designing Interaction. Psychology at the Human-Computer Interface. Cambridge, MA: Cambridge University Press, 128-153.

Payne, S., Squibb, H., & Howes, A. (1990). The nature of device models: the yoked state space and some experiments with test editors. Human Computer Interaction, 5, 415-444.

Pennington, N., Nicolich, R. and Rahm, J. (in press). Transfer of Training between cognitive subskills: Is knowledge really use specific? Cognitive Psychology.

Perlman, G. (1984). Making the right choices with menus. Proceedings of INTERACT '84, Elsevier, North-Holland, 317-321.

Polson, P. G., Lewis, C., Rieman, J., & Wharton, C. (1992). Cognitive walkthroughs: a method for theory-based evaluation of user interfaces. International Journal of Man-Machine Studies, 36, 741-773.

Polson, P., Lewis, C., Rieman, J., & Wharton, C. (1992). Cognitive walkthroughs: A method for theory-based evaluation of interfaces. International Journal for Man-Machine Studies, 36, 741-733.

Polson, P.G., 1988. The consequences of consistent and inconsistent interfaces. In R. Guindon (Ed.), Cognitive science and its application for human computer interaction. Hilldale, NJ: Lawrence Erlbaum Associates.

Rieman, J.F. (1993). The diary study: A workplace-oriented tool to guide laboratory studies. Proceedings of InterCHI ' 93 Conference on Human Factors in Computer Systems. New York: Association for Computing Machinery, 321-326.

Rieman, J.F. (1994). Learning Strategies and Exploratory Behavior of Interactive Computer Users. Ph.D. Dissertation, Dept. of Computer Science, University of Colorado, Dept. of Computer Science, Tech. Report CU-CS-723-94.

Rieman, J.F., Lewis, C., Young, R.M., & Polson, P. (1994). "Why is a raven like a writing desk?" Lessons in interface consistency and analogical reasoning from two cognitive architectures. Proceedings of CHI '94 Conference on Human Factors in Computer Systems. New York, Association for Computing Machinery, 438-444 .

Roberts, T. L. & Moran, T. P. (1982). The evaluation of computer text editors: methodology and empirical results. ACM Transactions on Office Information Systems, 1, 254-271.

Rosson, M.B. (1986). Classifying users. A hard look at some controversial issues. In Proceedings CHI ' 86 Proceedings, Boston, MA: ACM, 85-86.

Santhanam, R., & Wiedenbeck, S. (1993). Neither novice nor expert: the discretionary user of software. International Journal of Man-Machine Studies, 38, 201-229.

Shneiderman, B. & Margono, S. (1987). A study of file manipulations by novices using commands vs direct manipulation. Proceedings of the 26th Annunal Technical Symposium of the Washington, DC Chapter of the ACM, 154-159.

Shneiderman, B. (1982). The future of interactive systems and the emergence of direct manipulation. Behaviour and Information Technology, 1, 237-256.

Shneiderman, B. (1983). Direct manipulation: a step beyond programming languages. IEEE Computer, 57-69.

Simon, H.A. (1967). The Sciences of the Artifical. Cambridge, MA: MIT Press.

Singley, M. K., & Anderson, J.R. (1985). The transfer of text-editing skill. International Journal of Man-Machine Studies, 22, 403-423.

Slamecka, N.J., & Graf, P. (1978). The generation effect: delineation of a phenomenon. Journal of Experimental Psychology: Learning, Memory, & Cognition, 14, 223-239.

Smelcer, J.B., & Walker, N. (1993). Transfer of knowledge across computer command menus. International Journal of Human-Computer Interaction, 5, 147-165.

Streitz, N.A., Lieser, A., & Wolters, A. (1989). The combined effects of metaphor worlds and dialogue modes in human-computer-interaction. In F. Klix, N.A. Streitz, Y.Waern, & H. Wandke (Eds.), Man-Computer Interaction Research, MACINTER II, Elsevier, North Holland, 75-88.

Suchman, L.A. (1987). Plans and Situated Action: The problem of Human-Machine Communication. New York: Cambridge University Press.

Svendsen, G.B. (1991). The influence of interface style on problem solving. International Journal of Man-Machine Studies, 35, 379-397.

Van Hoe, R., Poupeye, K., Vandierendonck, A., & DeSoete, G. (1990). Some effect of menu characteristics and user personality on performance with menu-driven interfaces. Behaviour & Information Technology, 9, 17-29.

Walker, N., Smelcer, J.B., & Nilsen, E. (1991). Optimizing speed and accuracy of menu selection: A comparison of wlking and pull-down menus. International Journal of Man-Machine Studies, 35, 871-890.

Whiteside, J., Jones, S. Levy, P.S., & Wixon, D. (1985). User performance with command, menu, and iconic, interfaces. In Proceedings CHI '85, New York: ACM, 185-191.

Young, R.M., Howes, A., & Whittington, J. (1990). A knowledge analysis of interactivity. Proceedings of Interact '90, Cambridge, England, 881-885.

Ziegler, J.E. & Fähnrich, K.-P. (1988). Direct Manipulation. In M. Helander (ed.): Handbook of Human-Computer Interaction. Elsevier Science Publishers, North-Holland.

Ziegler, J.E., Vossen, P., & Hoppe, H.U. (1986). On using production systems for cognitive task analysis and prediciton of transfer of cognitive skill. Proceedings of the Third European Conference on Cognitive Ergonomics, Paris.

APPENDIX A

QUESTIONNAIRE

Macintosh Experiment Questionnaire

Please indicate your:

Age: _____

Sex: _____

Educational level: _____

Please answer the following Questions as accurately as possible:

1. For how long have you been using Macintosh computers?

Less than a year, but over several months	_____
a year	_____
two years	_____
three years	_____
four years	_____
more than four years	_____

2. Which applications have you been using during that time, and how much?

	not at all	< 50 hrs	> 50 < 100 hrs	> 100 hrs
Microsoft Word	_____	_____	_____	_____
Microsoft Excel	_____	_____	_____	_____
Cricket Graph	_____	_____	_____	_____
Mac Draw	_____	_____	_____	_____
Super Paint	_____	_____	_____	_____

Other?

_____

_____

3. For how long have you been using PC's, IBM computers, or clones?

Not at all	_____
Less than a year, but over several months	_____
a year	_____
two years	_____
three years	_____
four years	_____
more than four years	_____

4. If you have used PC's, which applications have you been using during that time, and how much?

	not at all	< 50 hrs	> 50 < 100 hrs	> 100 hrs
Microsoft Word	_____	_____	_____	_____
Microsoft Excel	_____	_____	_____	_____
Lotus	_____	_____	_____	_____
Word Perfect	_____	_____	_____	_____
DBase	_____	_____	_____	_____
Windows?	_____	_____	_____	_____

Others?

_____

_____

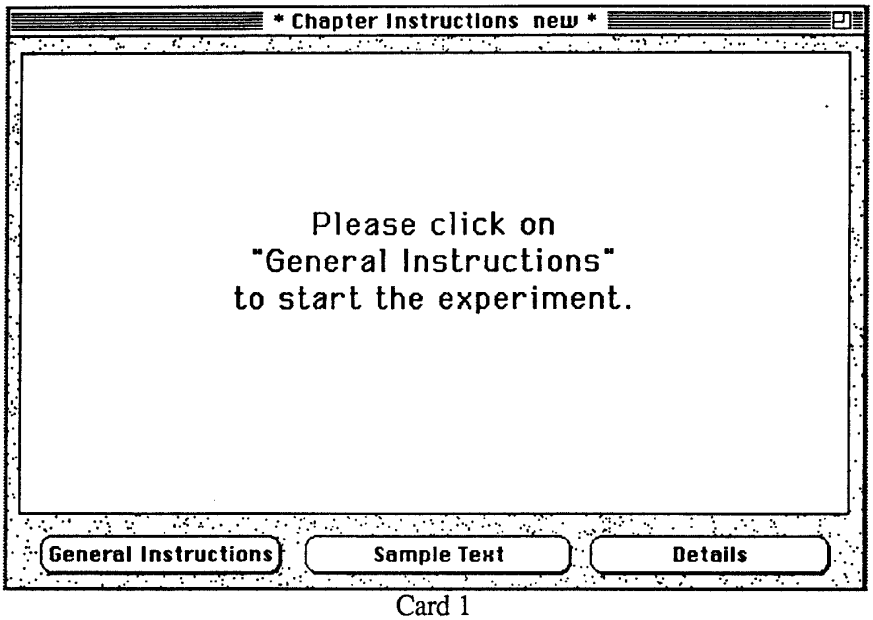
5. Estimate the number of graphs that you have drawn by hand in your life:

none	_____
less than 10	_____
less than 50	_____
less than 100	_____
more than 100	_____

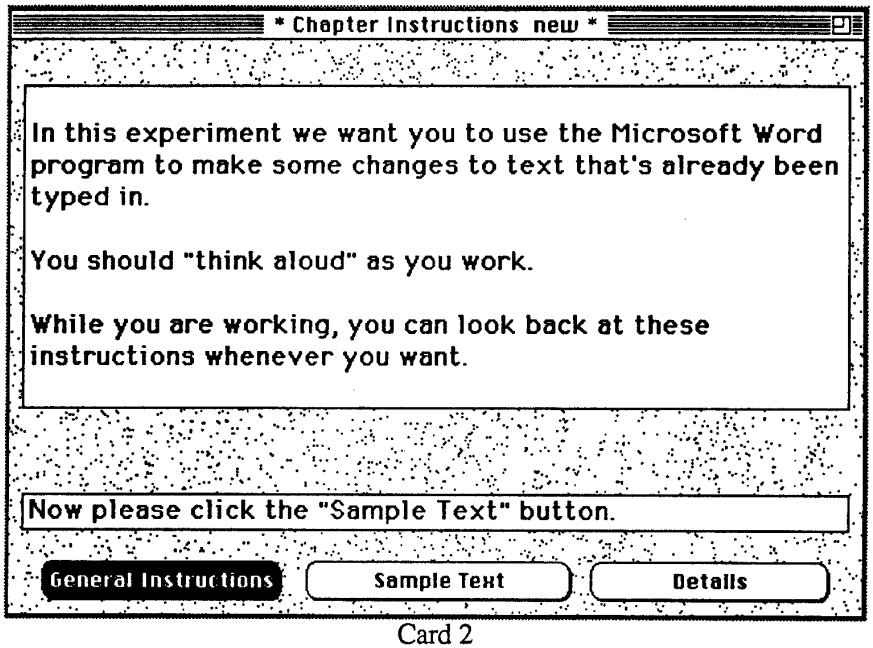
APPENDIX B

EXAMPLES FOR INSTRUCTION MATERIALS

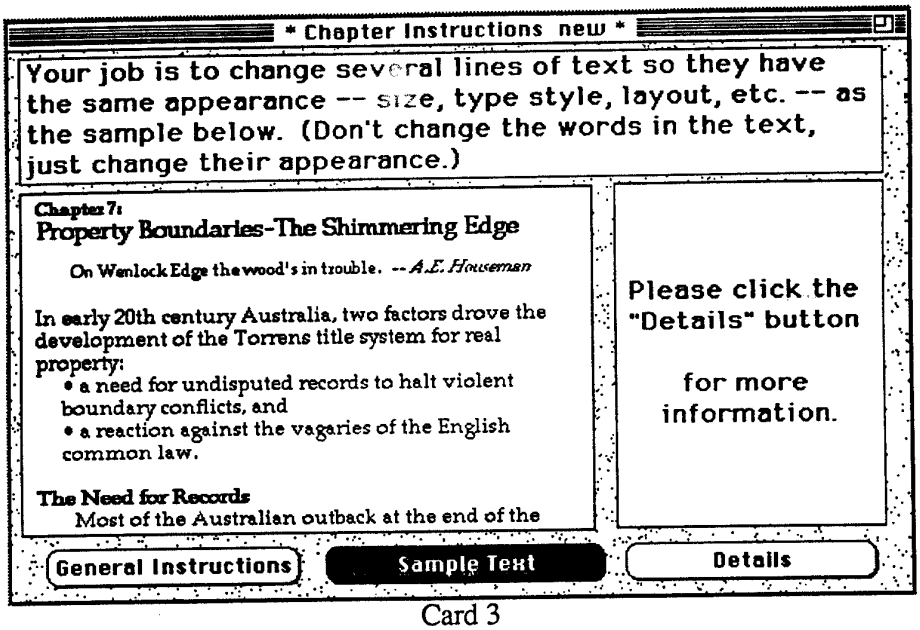
1. Instructions for editing warm-up tasks
1.1 Cards 1 - 4, containing general instructions



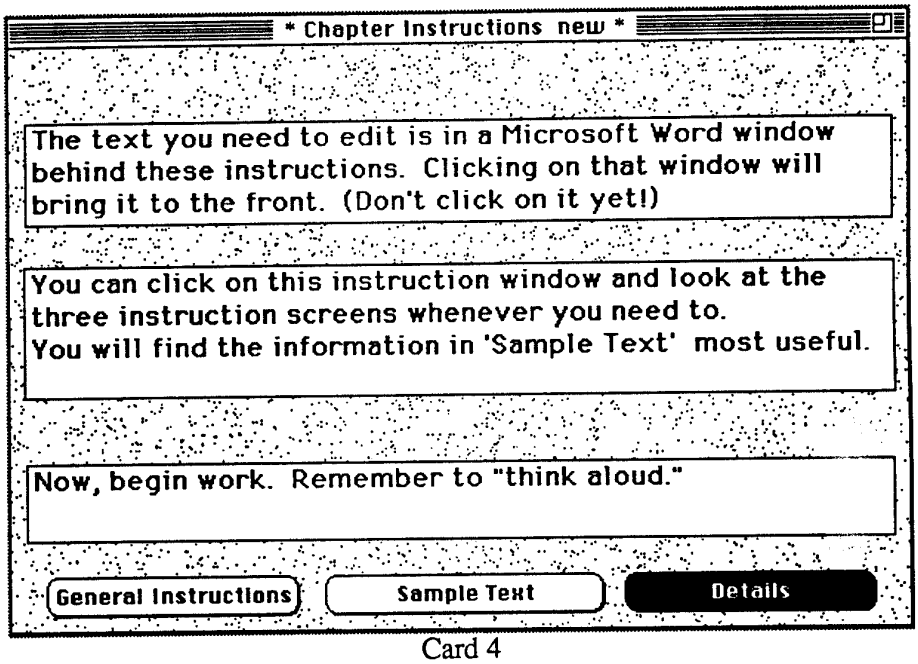
Card 1



Card 2

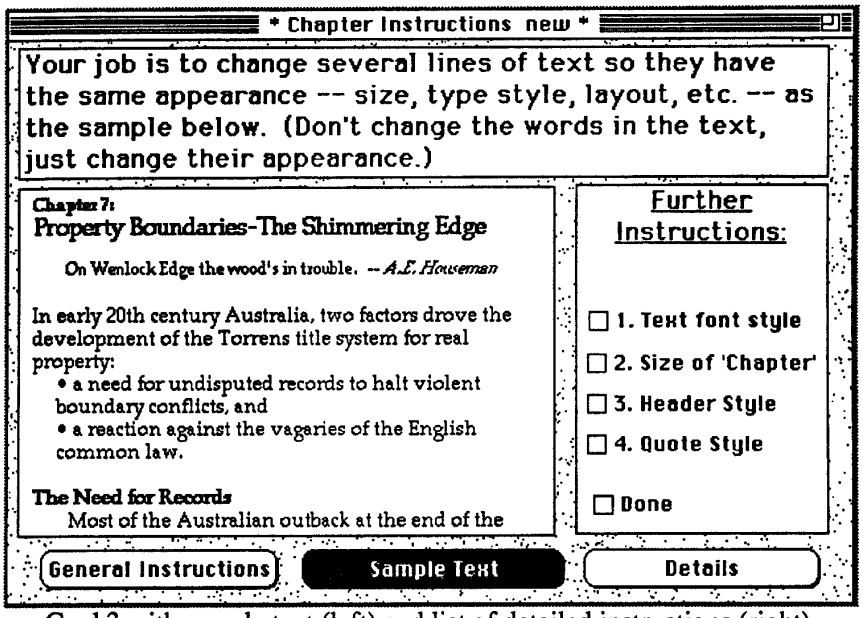

* Chapter Instructions new *
Your job is to change several lines of text so they have the same appearance -- size, type style, layout, etc. -- as the sample below. (Don't change the words in the text, just change their appearance.)
Chapter 7:
Property Boundaries-The Shimmering Edge
On Wenlock Edge the wood's in trouble. -- A.E. Housman
In early 20th century Australia, two factors drove the development of the Torrens title system for real property:
• a need for undisputed records to halt violent boundary conflicts, and
• a reaction against the vagaries of the English common law.
The Need for Records
Most of the Australian outback at the end of the
Please click the "Details" button for more information.
General Instructions
Sample Text
Details

Card 3

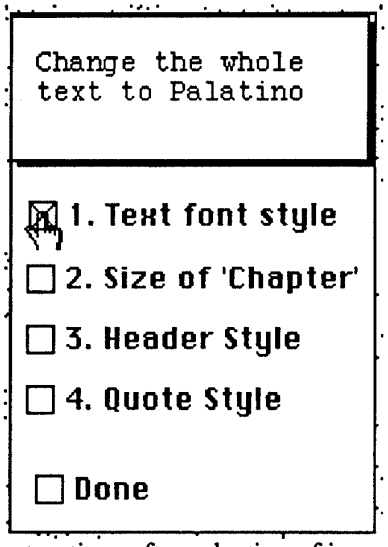

* Chapter Instructions new *
The text you need to edit is in a Microsoft Word window behind these instructions. Clicking on that window will bring it to the front. (Don't click on it yet!)
You can click on this instruction window and look at the three instruction screens whenever you need to.
You will find the information in 'Sample Text' most useful.
Now, begin work. Remember to "think aloud."
General Instructions
Sample Text
Details

Card 4

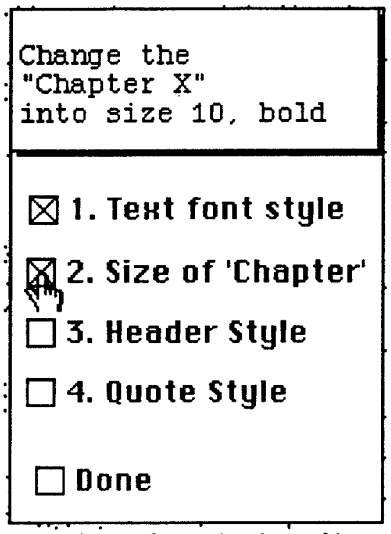
1.2 Details of Card 3, containing specific editing instructions



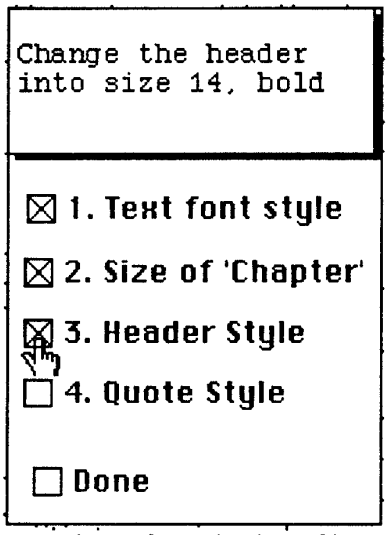
Card 3 with sample text (left) and list of detailed instructions (right).



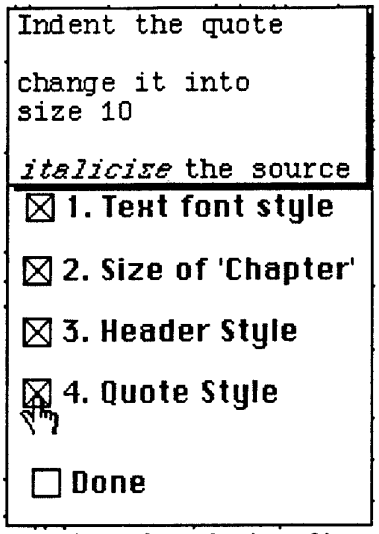
List of instructions after selection of instruction 1.



List of instructions after selection of instruction 2.

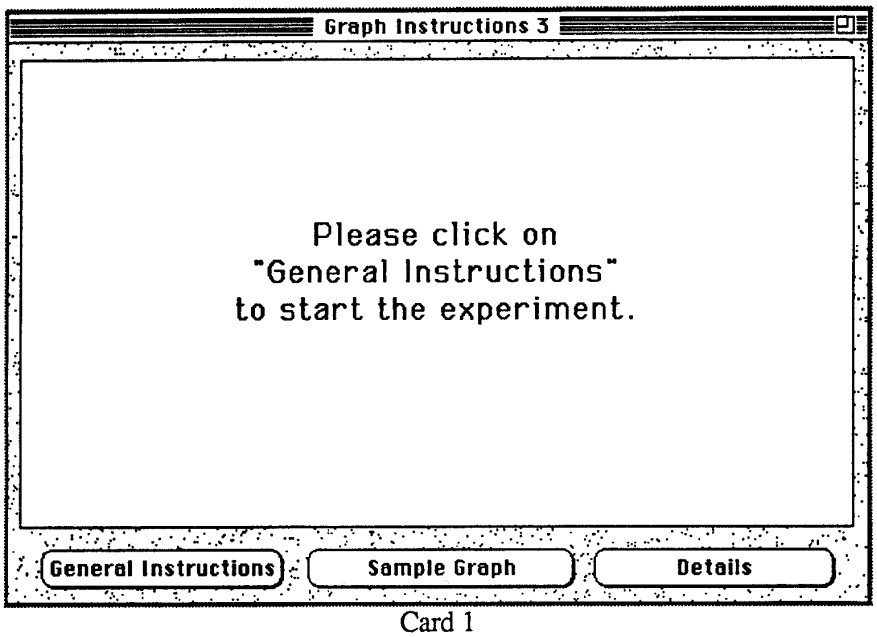


List of instructions after selection of instruction 3.

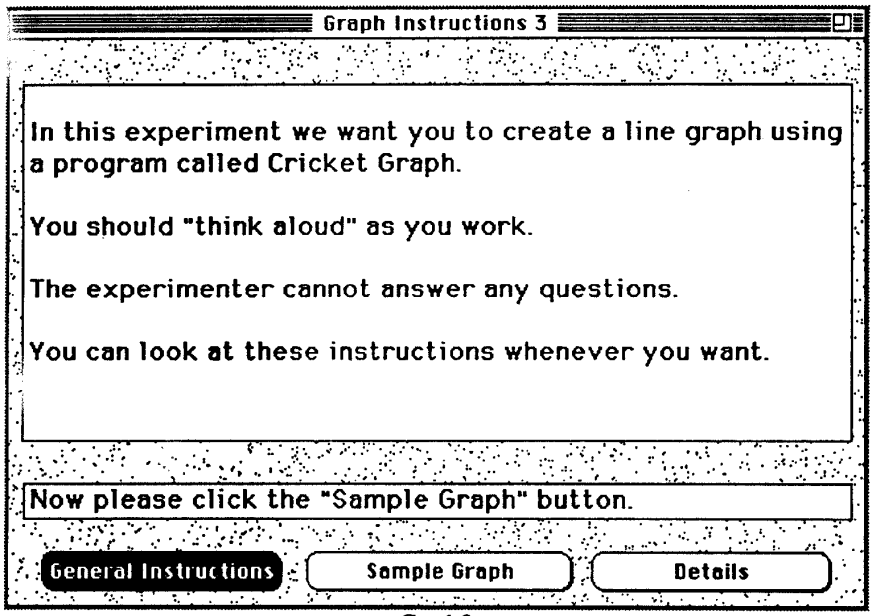


List of instructions after selection of instruction 4.

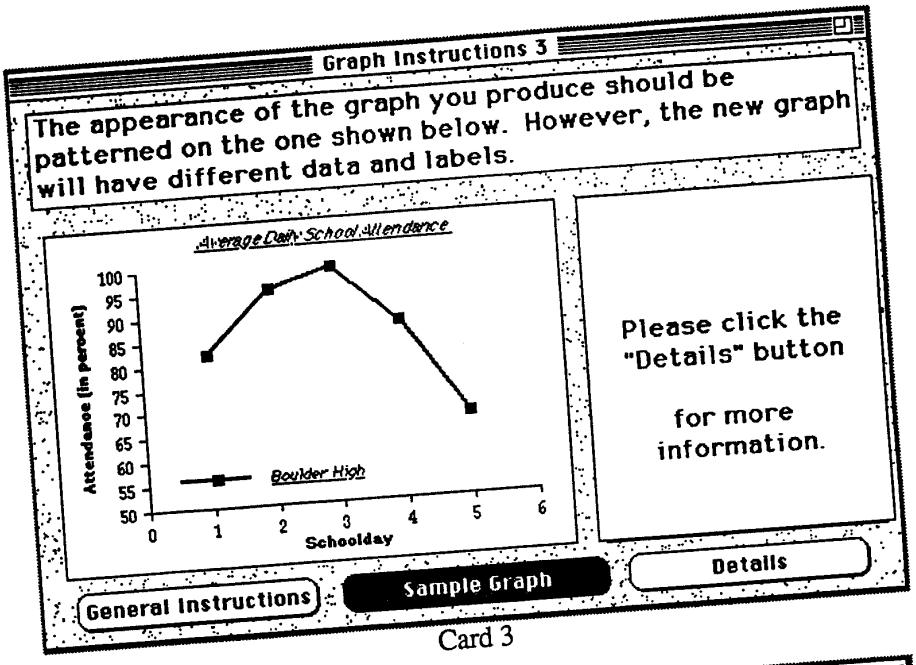
2. Instructions for experimental graphing tasks
2.1 Cards 1 - 4, containing general instructions



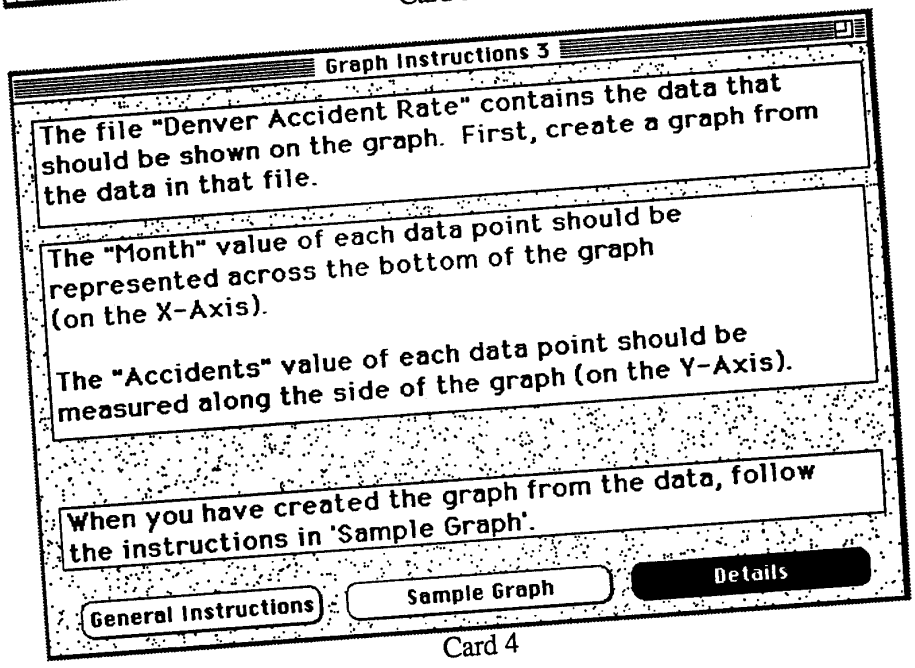
Card 1



Card 2

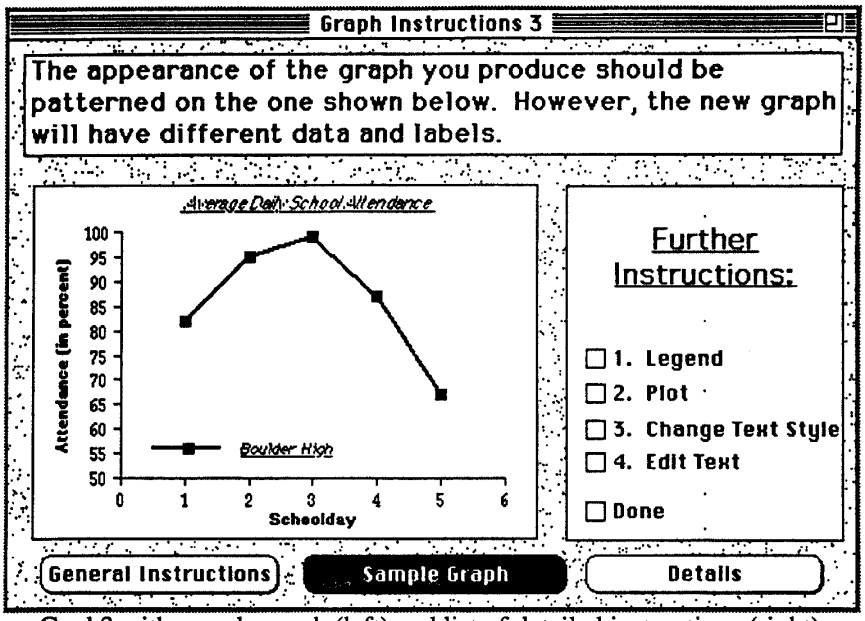

Graph Instructions 3
The appearance of the graph you produce should be patterned on the one shown below. However, the new graph will have different data and labels.
Average Daily School Attendance
Attendance (in percent)
100
95
90
85
80
75
70
65
60
55
50
0
1
2
3
4
5
6
Schoolday
Boulder High
Please click the "Details" button
for more information.
General Instructions
Sample Graph
Details

Card 3

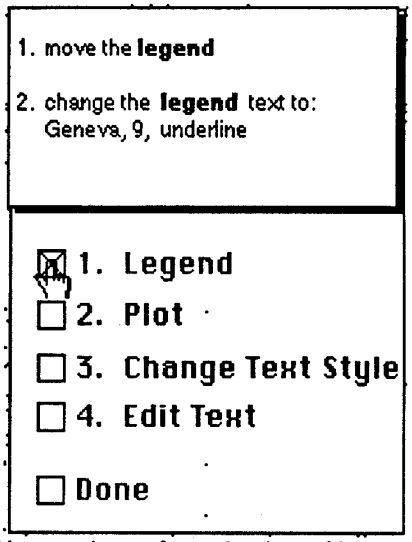

Graph Instructions 3
The file "Denver Accident Rate" contains the data that should be shown on the graph. First, create a graph from the data in that file.
The "Month" value of each data point should be represented across the bottom of the graph (on the X-Axis).
The "Accidents" value of each data point should be measured along the side of the graph (on the Y-Axis).
When you have created the graph from the data, follow the instructions in 'Sample Graph'.
General Instructions
Sample Graph
Details

Card 4

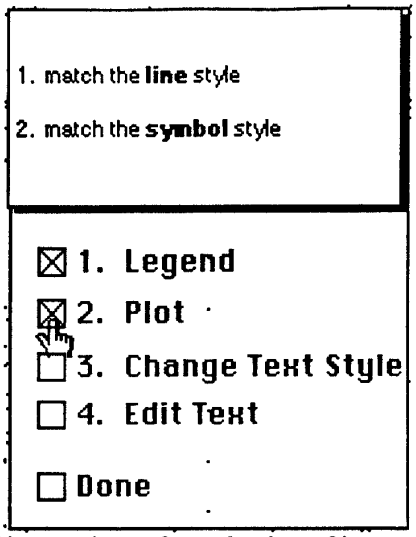
2.2 Details of Card 3, containing specific graph modification instructions



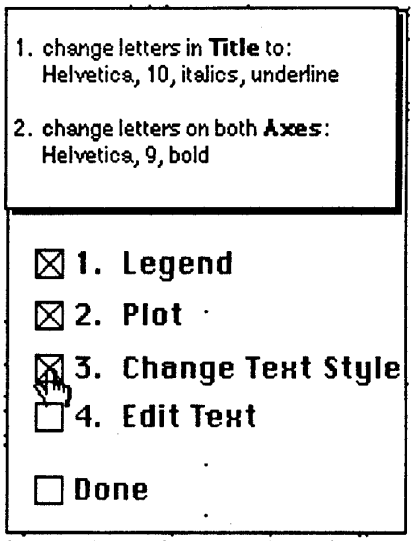
Card 3 with sample graph (left) and list of detailed instructions (right).



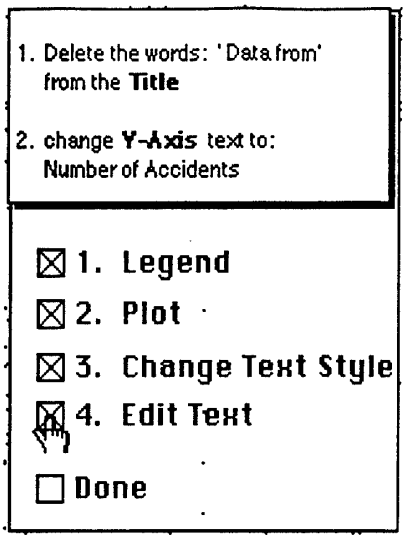
List of instructions after selection of instruction 1.



List of instructions after selection of instruction 2.



List of instructions after selection of instruction 3.

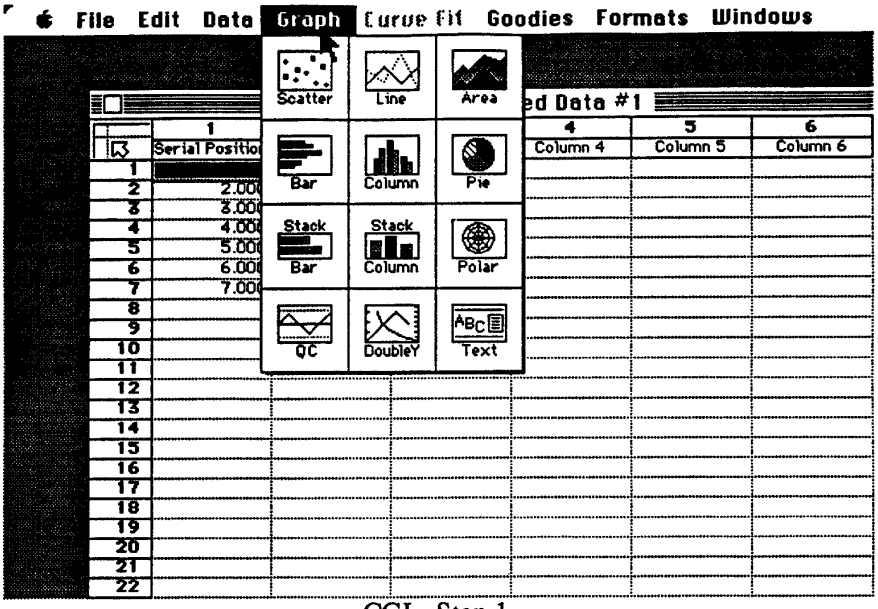


List of instructions after selection of instruction 4.

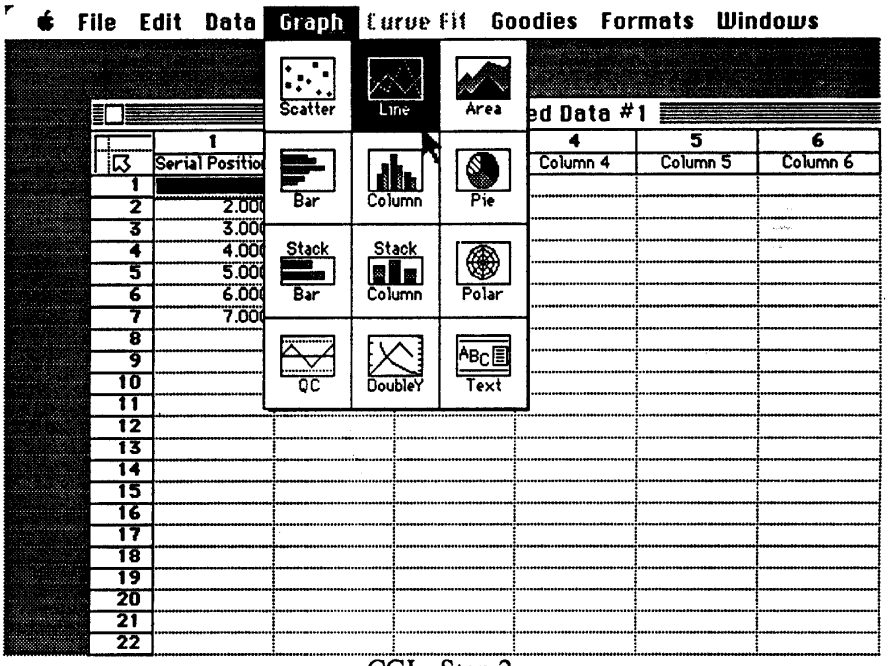
APPENDIX C

INTERACTION SEQUENCES FOR GRAPH CREATION IN THE FOUR INTERFACE CONDITIONS

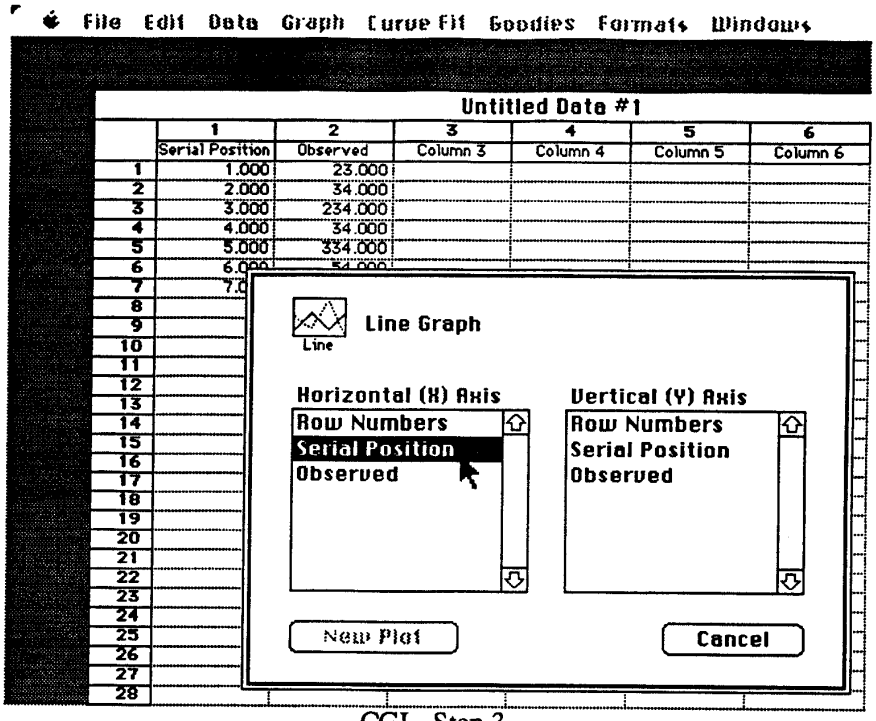
1. CG I



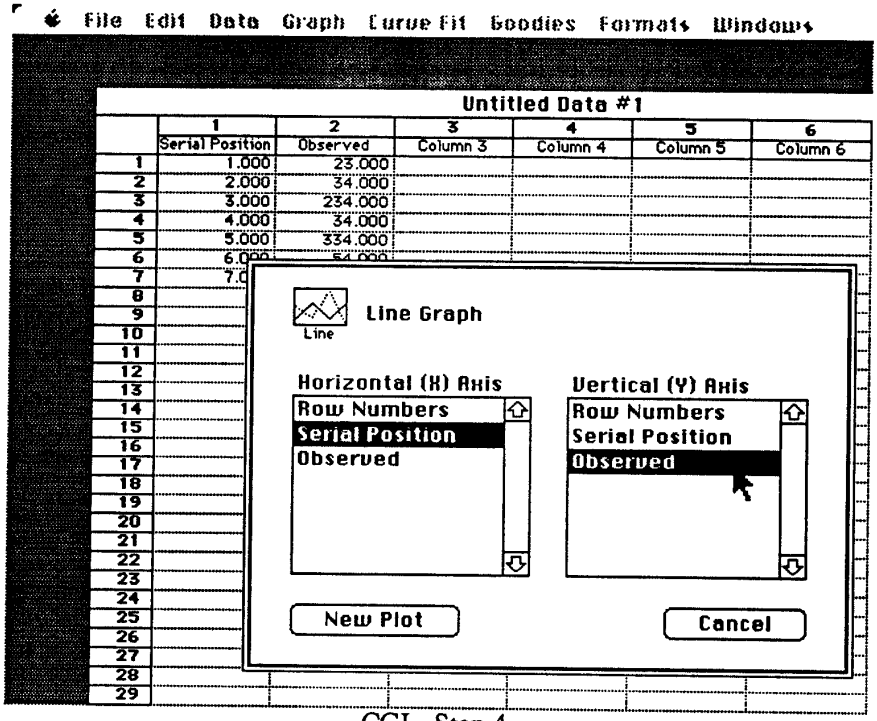
CGI - Step 1



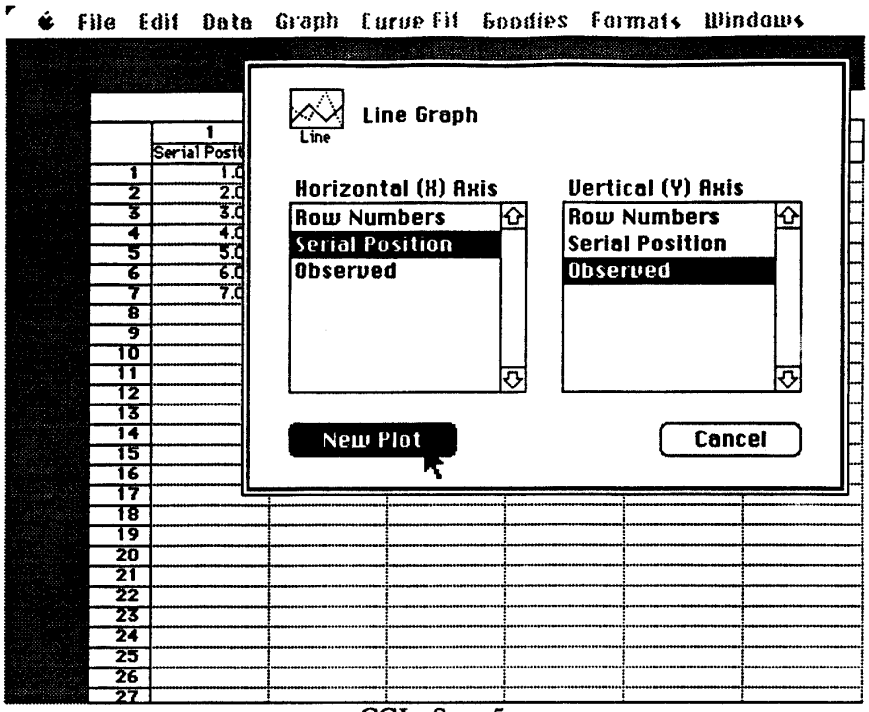
CGI - Step 2



CGI - Step 3

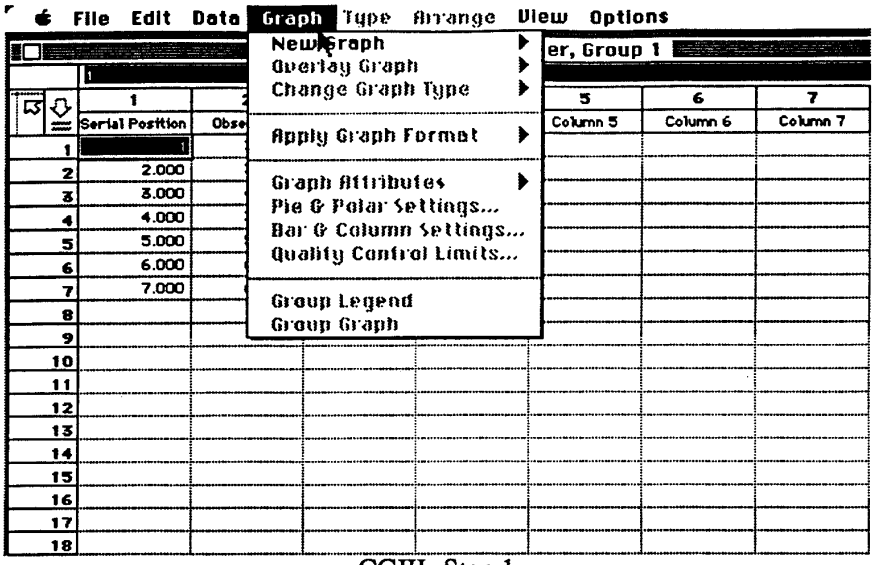


CGI - Step 4

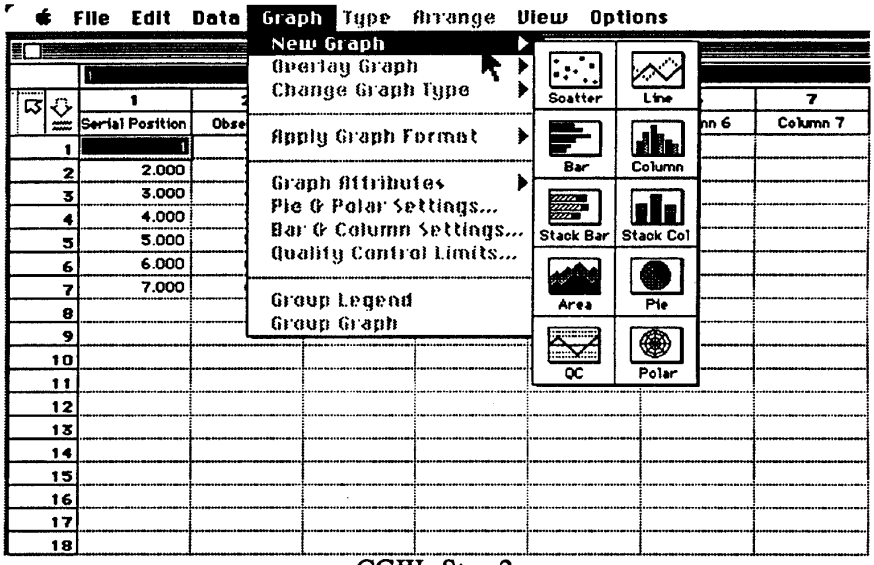

File Edit Data Graph Curve Fit Goodies Formats Windows
Line Graph
Line
Horizontal (X) Axis
Row Numbers
Serial Position
Observed
Vertical (Y) Axis
Row Numbers
Serial Position
Observed
New Plot
Cancel

CGI - Step 5

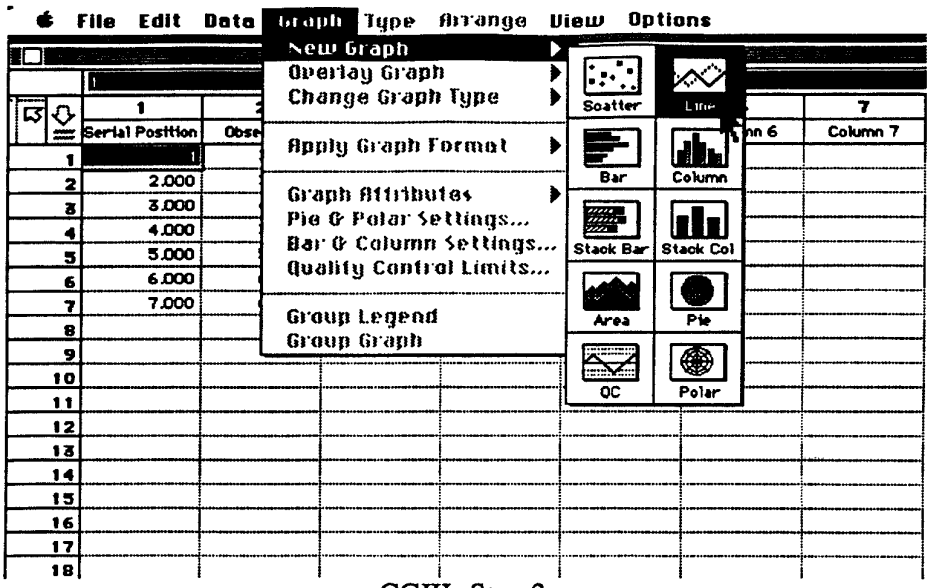
2. CG III



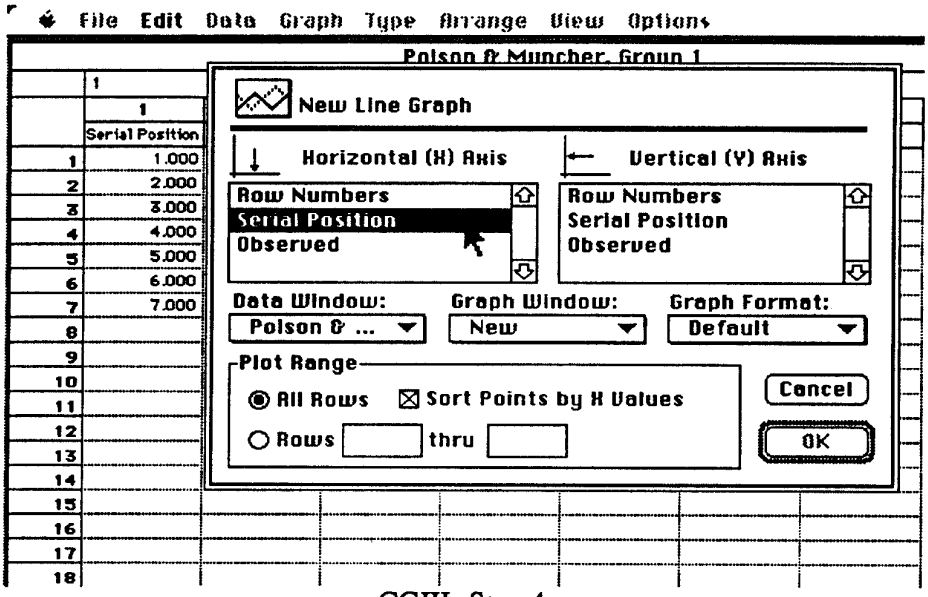
CGIII- Step 1



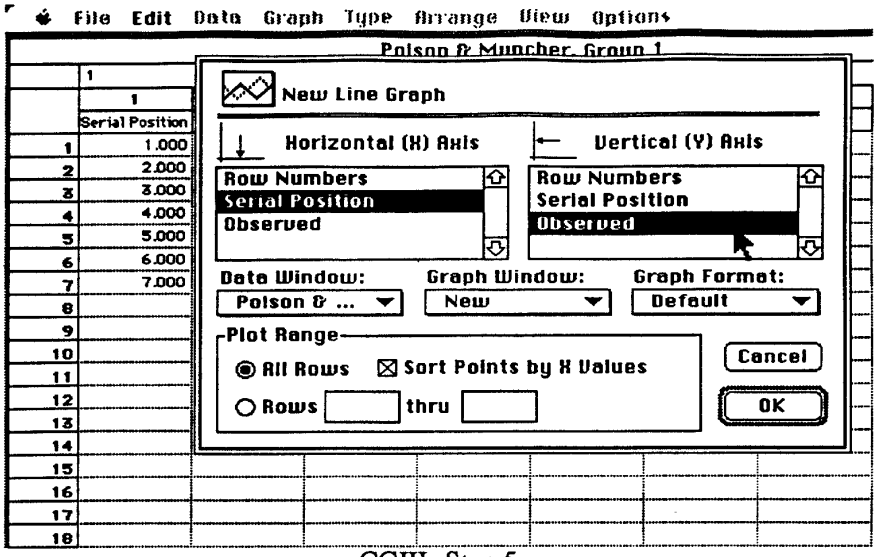
CGIII- Step 2



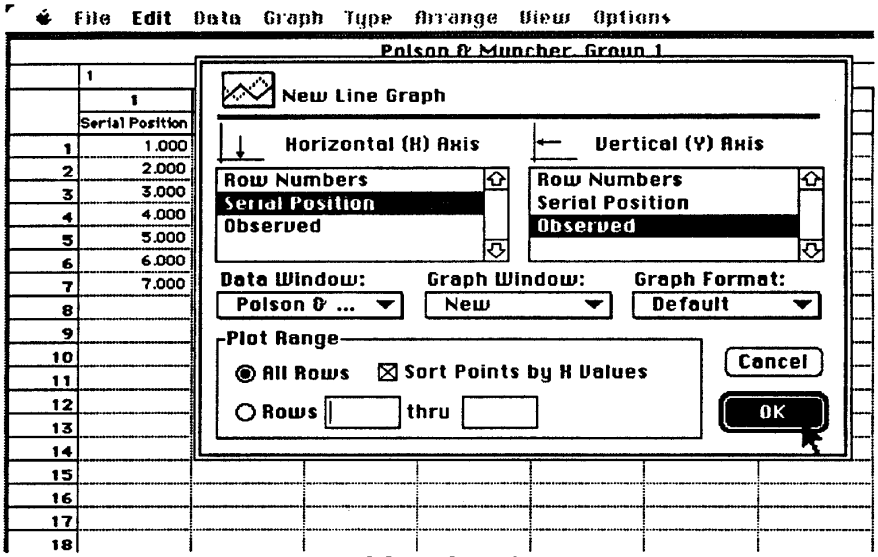
CGIII- Step 3



CGIII- Step 4

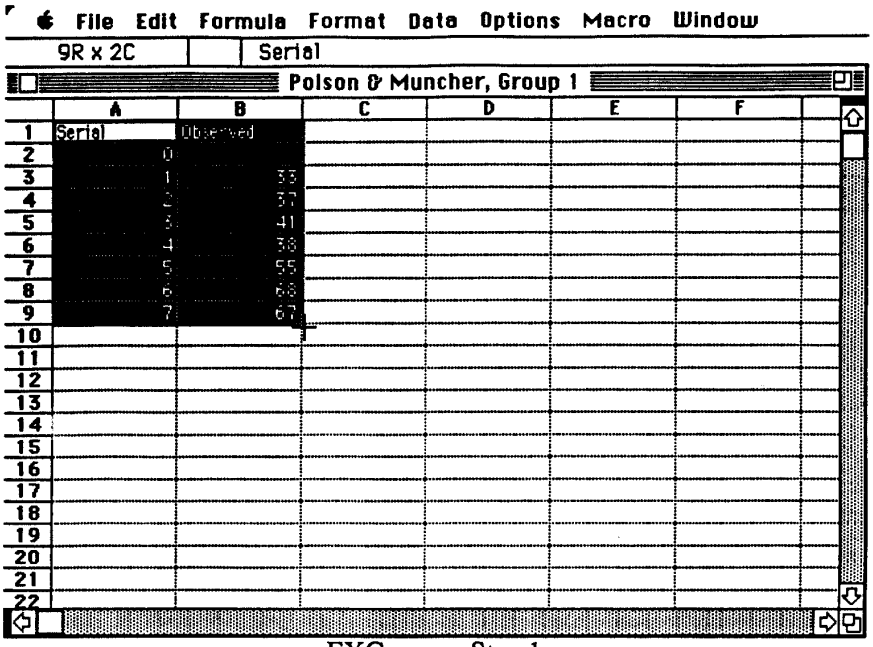


CGIII- Step 5

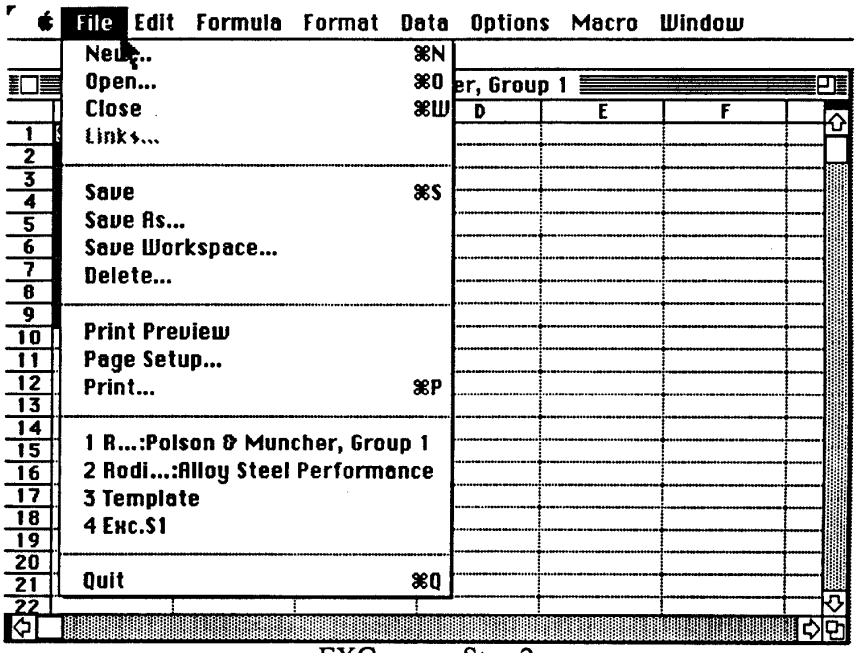


CGIII- Step 6

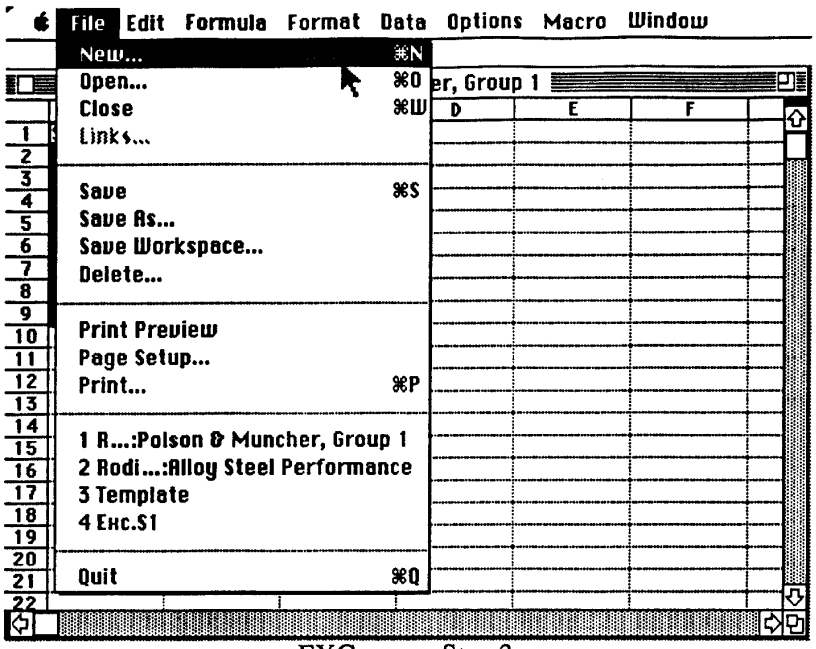
3. EXC menu



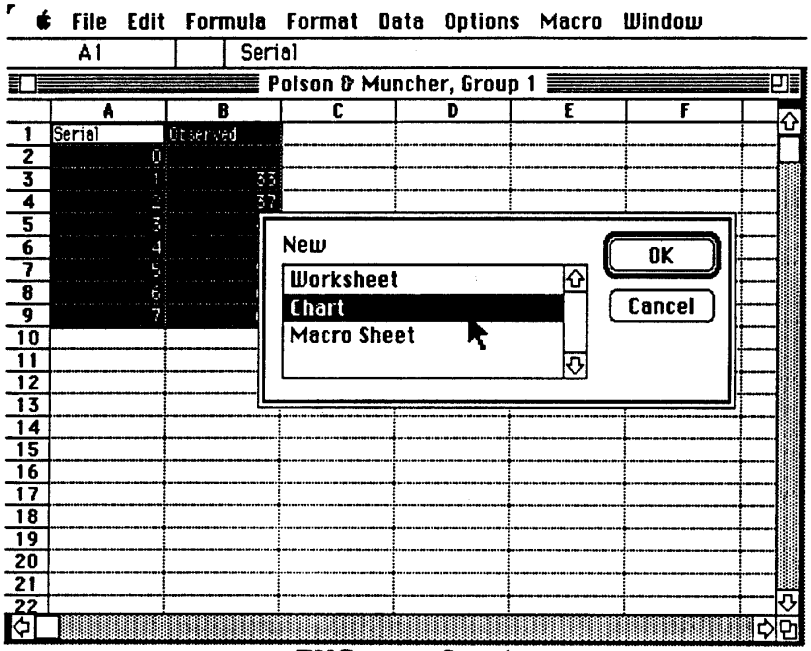
EXC menu- Step 1



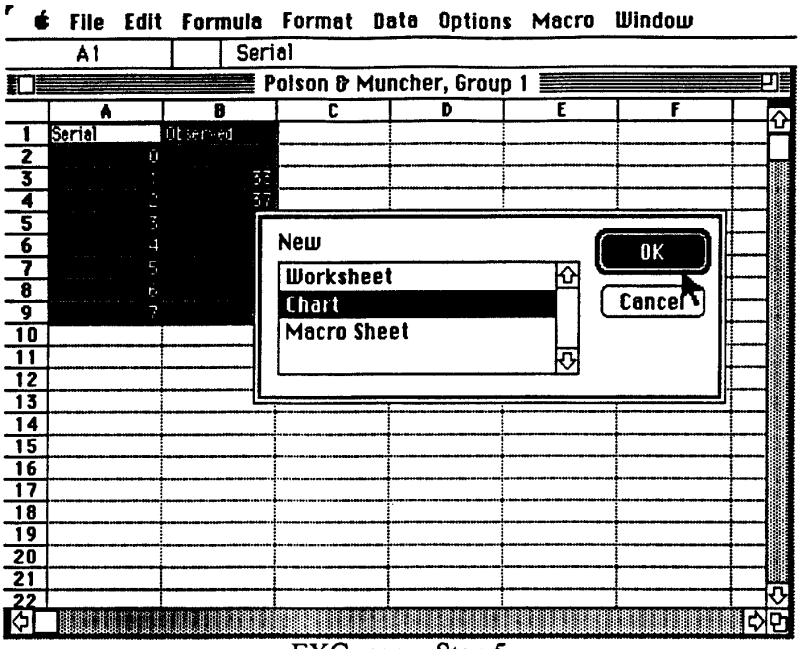
EXC menu- Step 2



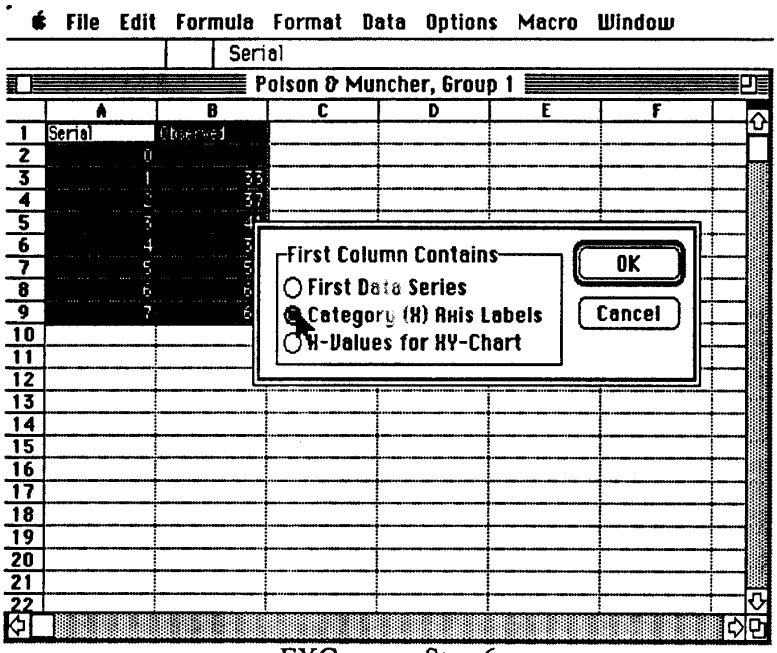
EXC menu- Step 3



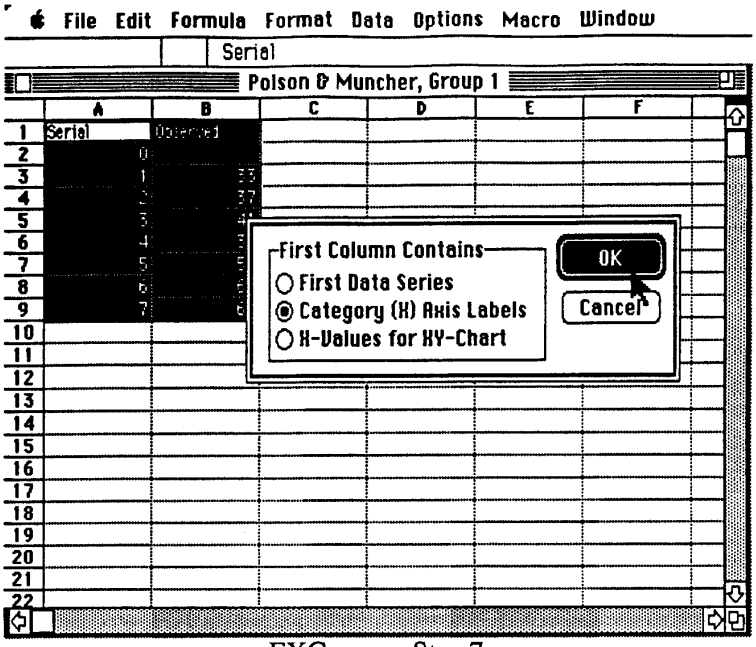
EXC menu- Step 4



EXC menu- Step 5

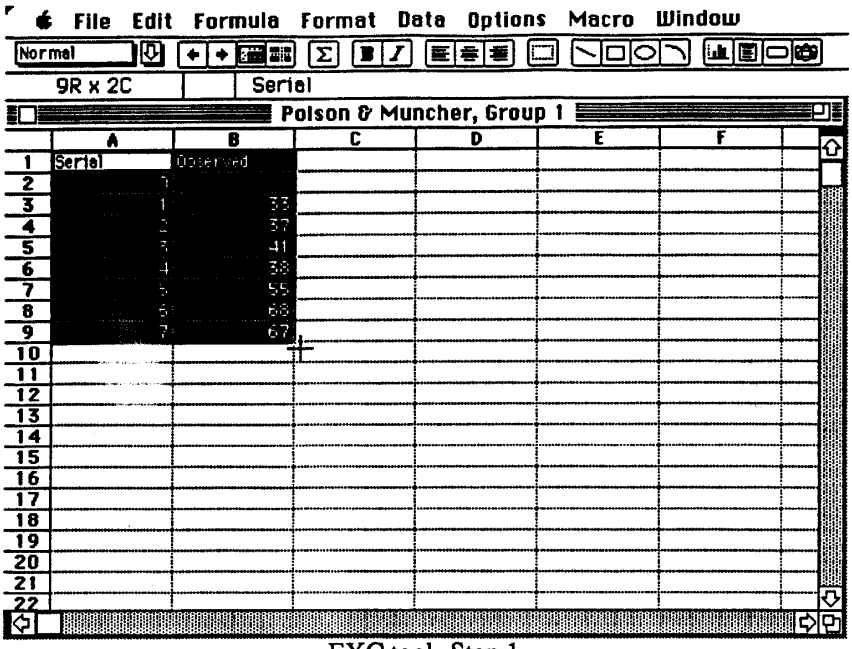


EXC menu- Step 6

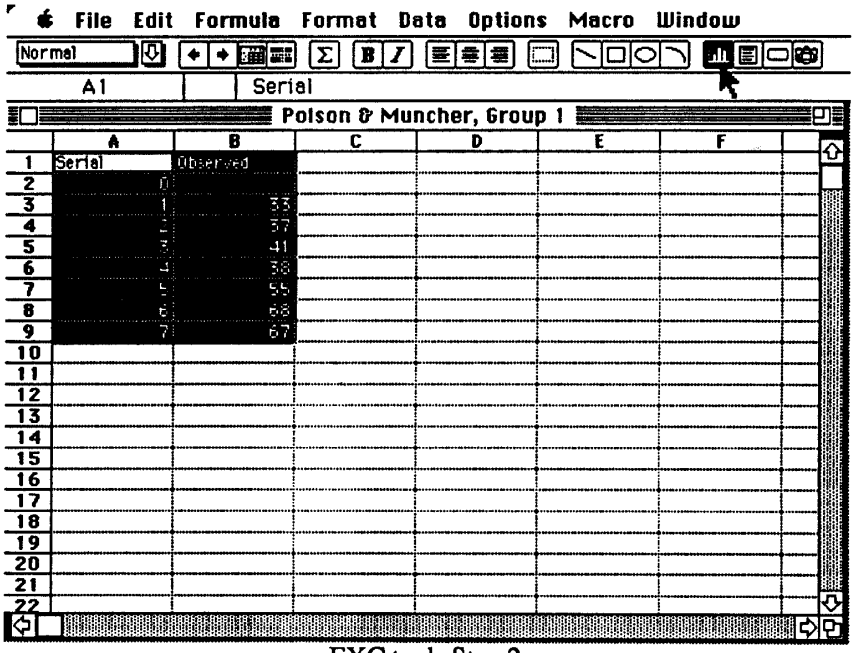


EXC menu- Step 7

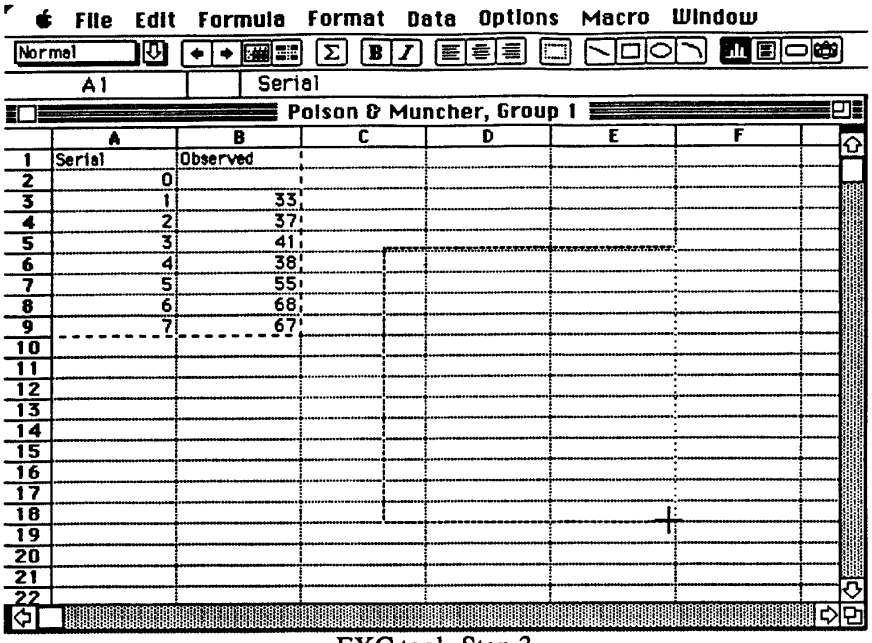
4. EXC tool



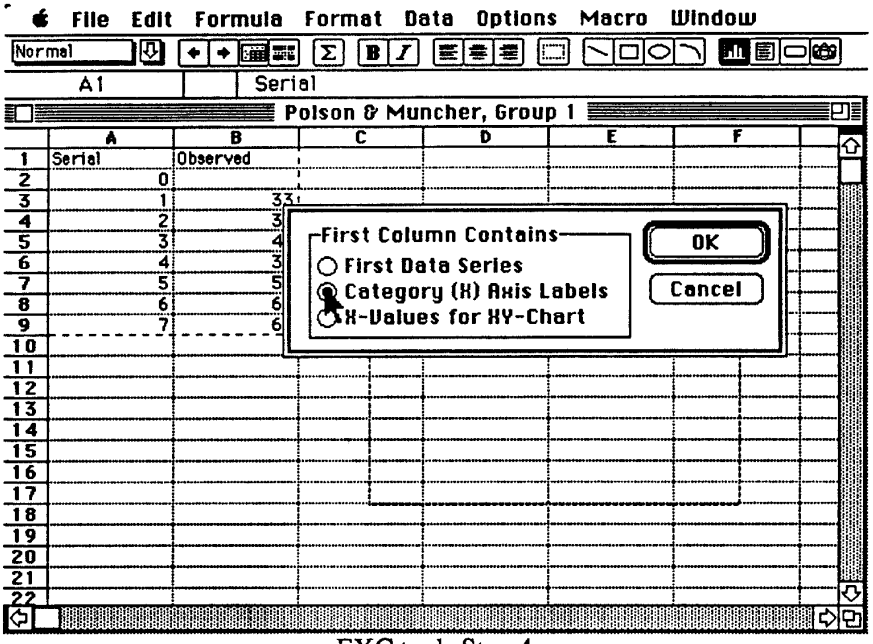
EXC tool- Step 1



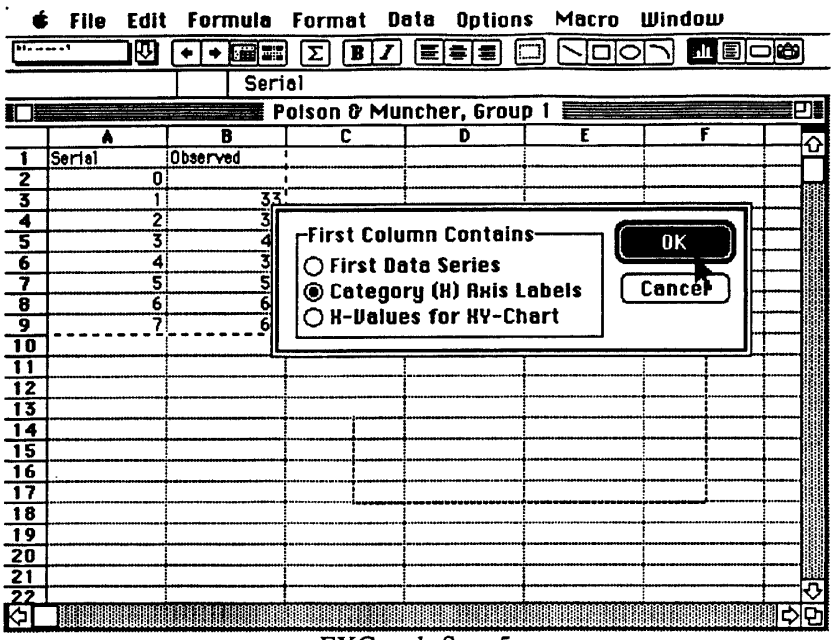
EXC tool- Step 2



EXC tool- Step 3



EXC tool- Step 4

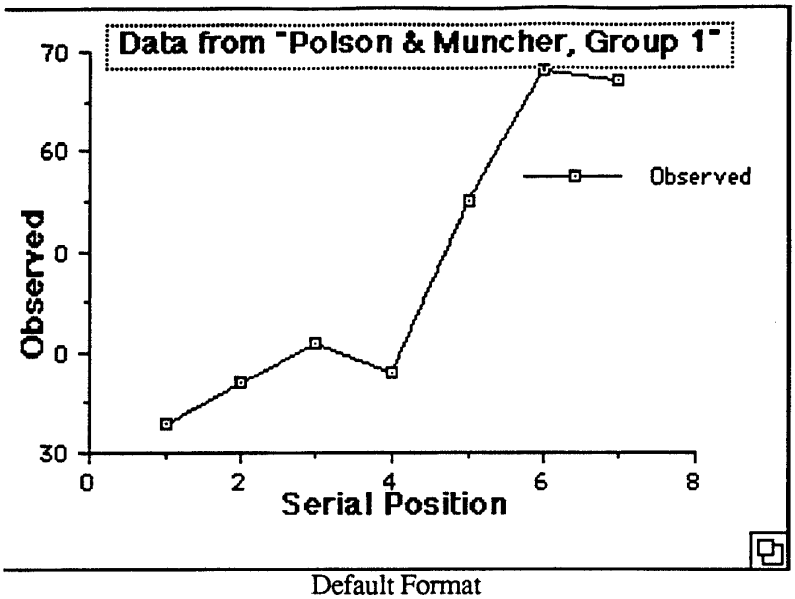


EXC tool- Step 5

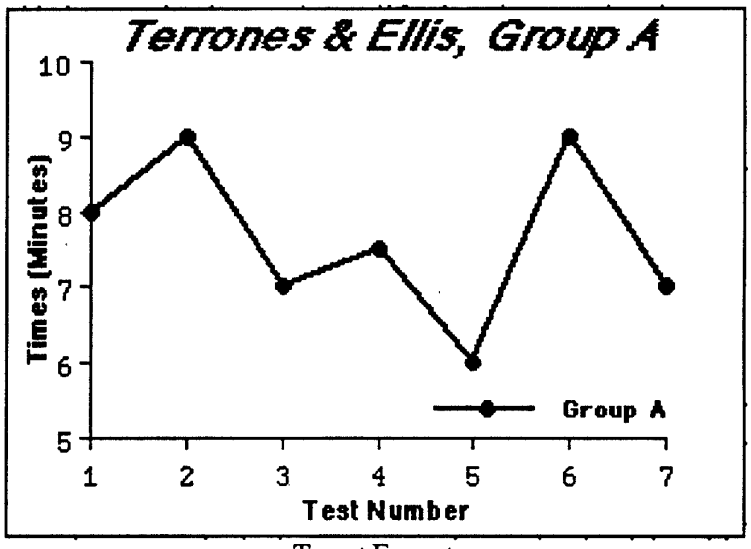
APPENDIX D

DEFAULT AND TARGET FORMATS OF GRAPHS CREATED IN THE EXPERIMENTAL TASKS

1. Default and Target Formats of the Graphs in Task 1

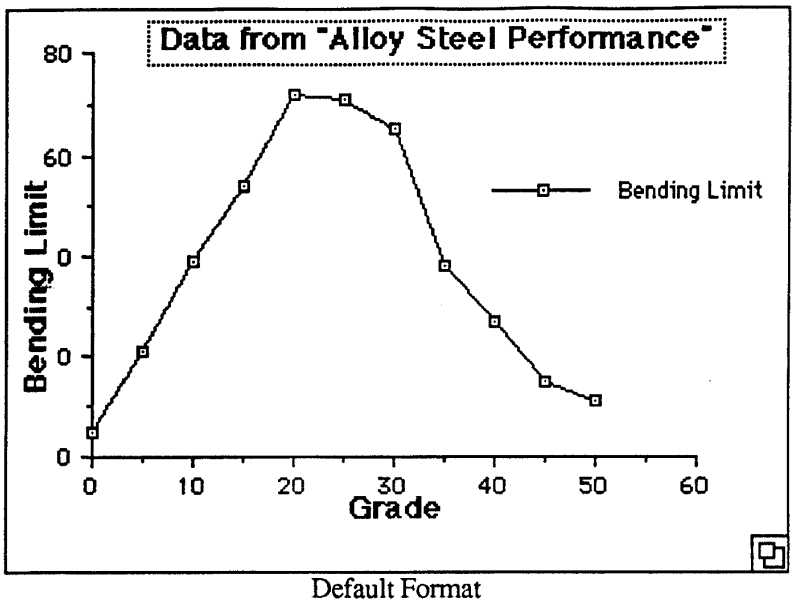


Default Format

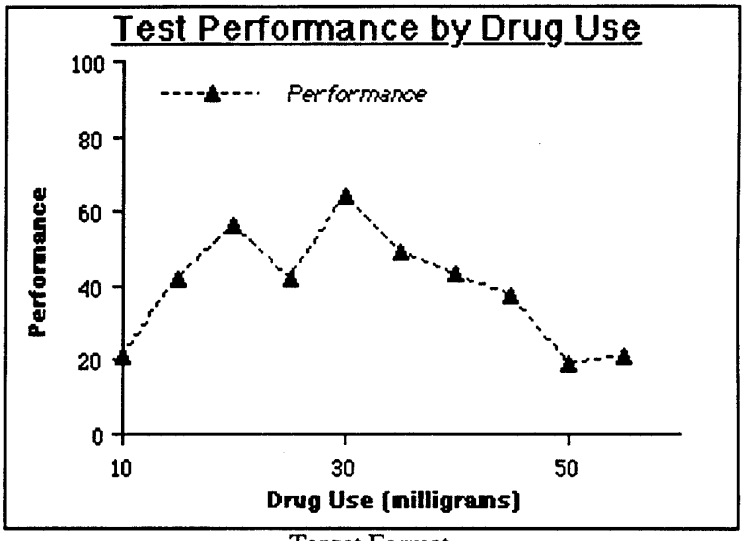


Target Format

2. Default and Target Format of the Graphs in Task 2

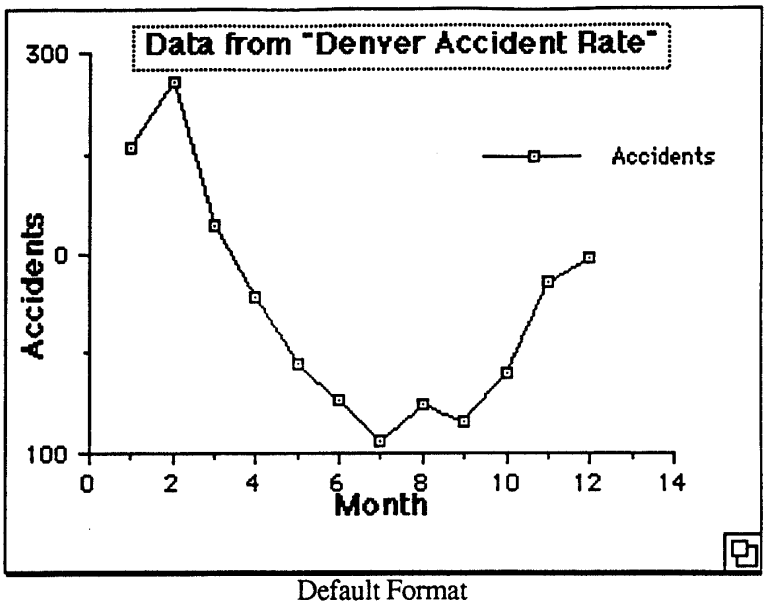


Default Format

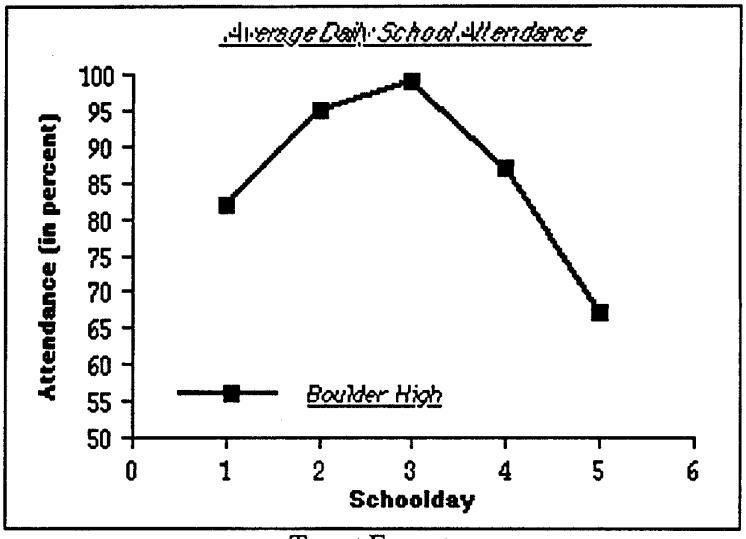


Target Format

2. Default and Target Format of the Graphs in Task 3



Default Format



Target Format

APPENDIX E

SAMPLE TRANSCRIPT

CG1 no delay, subject 45								
Task 1		time	time	instr	instr	corr	corr	comments
Task A: open Data sheet and application								
	Method 1							
1	double-click Data icon	0.02						
				0.17	0.4			
Task B: Create Graph								
	Method 1							
2	Select Graph from menu bar	0.58						
3	Select line graph	1.03		1.14	1.26	1.27	1.42	
4	chose1. axis label	1.45		1.51	1.54	1.57	2	
5	chose 2. axis label	2.04						
6	click ok	2.05						
Task C: Move Legend				2.09	2.17			
				2.31	2.38			
	Method 1							
7	Click and drag legend to new l	2.39	2.43					
Task D: Change Legend Type				2.47	2.49			
				3.04	3.08			
	Method 1			4.04	4.29			
								hint : work with the legend itself: 4.33
8	double click on legend text	4.42						hint : click a little faster in succession : 4.41
9	click on bold in d-box	4.47						
10	click ok	4.49						
Task E&F: Change Line and Symbol Style				4.51	5.02			
				6.2	6.27			
	Method 1							hint : try to work with the legend : 6.33
								hint : focus on legend : 7.05
11	double click on legend symbol	7.22						hint : be more specific about where you click
12	select symbol	7.31						
13	select line	7.33						
14	click ok	7.36						
Task G: Change Title Style				7.39	8.02			
	Method 1							
15	double click on title	8.05						
16	click on 18	8.1		8.12	8.2			
17	click on bold	8.22						
18	click ok	8.23						
Task H: Change Axis Style 1				8.25	8.29			
	Method 1							
19	double click on axis title	8.31						
20	click on 10	8.36						
21	click ok	8.4						
Task I: Change Axis Style 2								
	Method 1							
22	double click on axis title	8.42						
23	click on 10	8.47						
24	click ok	8.49						
Task J: Edit Title (delete)				8.52	9.06			
	Method 1							
25	double click on Title	9.12						
26	Position cursor in edit window	9.15	9.16					
27	delete text	9.17						

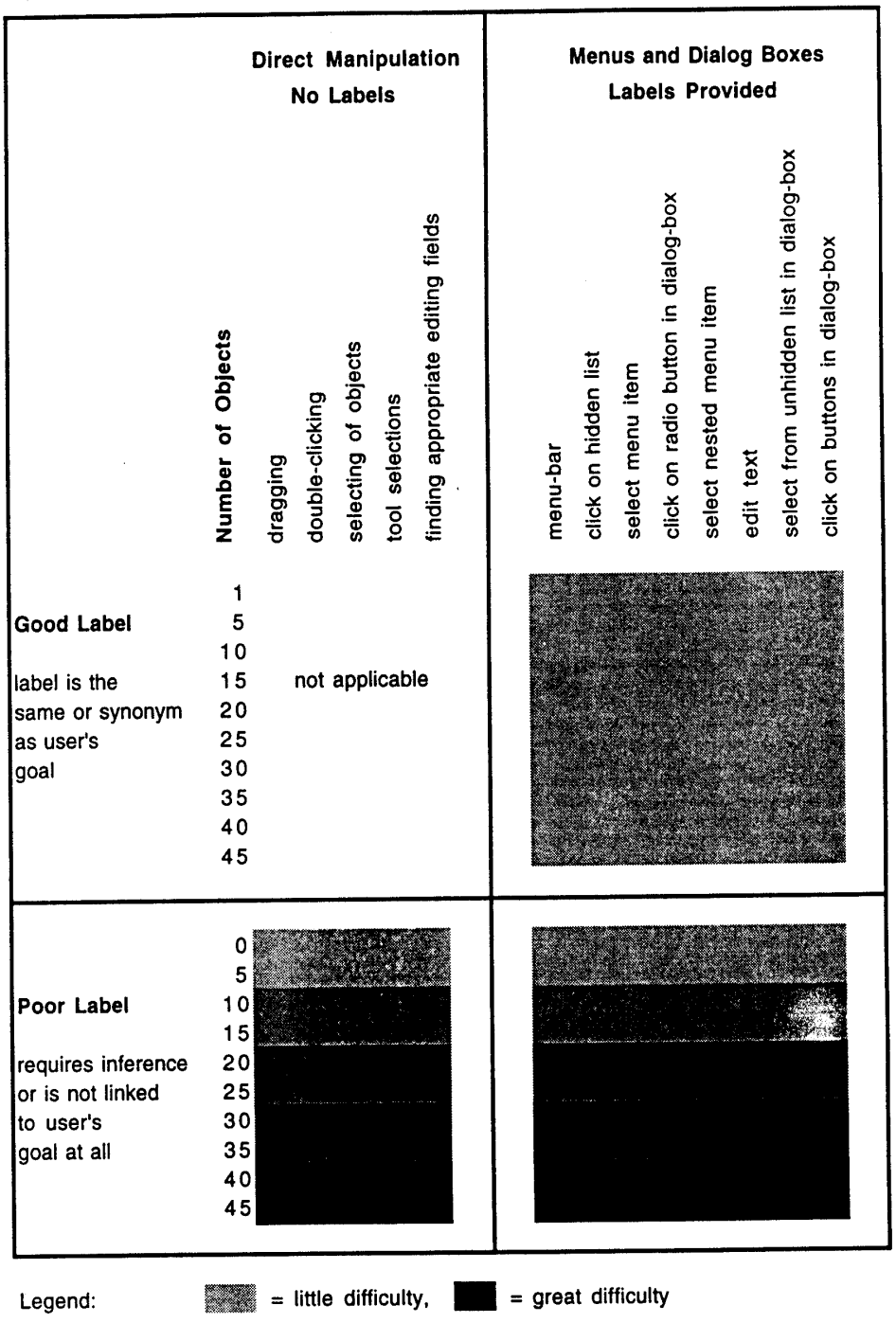
APPENDIX F

SAMLE CODE

nr	description	time	subid	interface	condition	trial	type	objects	sem diff
1	double-click Data icon	0.1	10	CG1	delay	1	dc	1	0
2	Select Graph from menu bar	0.433	10	CG1	delay	1	mb	8	0
3	Select line graph	0.133	10	CG1	delay	1	mm	12	0
4	chose1. axis label	0.083	10	CG1	delay	1	dl	4	2
5	chose 2. axis label	0.433	10	CG1	delay	1	dl	4	2
6	click ok	0.033	10	CG1	delay	1	db	4	0
7	Click and drag legend to new location	0.233	10	CG1	delay	1	dd	4	2
8	double click on legend text	2.55	10	CG1	delay	1	dc	4	2
9	click on bold in d-box	0.333	10	CG1	delay	1	dl	9	0
10	click ok	0.067	10	CG1	delay	1	db	2	0
11	double click on legend symbol	2.733	10	CG1	delay	1	dc	4	2
12	select line	0.167	10	CG1	delay	1	dl	5	0
13	select symbol	0.117	10	CG1	delay	1	dl	40	0
14	click ok	0.083	10	CG1	delay	1	db	2	0
15	double click on title	0.533	10	CG1	delay	1	dc	4	1
16	click on bold	0.117	10	CG1	delay	1	dl	9	0
17	click on 18	0.067	10	CG1	delay	1	dl	5	2
18	click ok	0.067	10	CG1	delay	1	db	2	0
19	double click on axis title	0.2	10	CG1	delay	1	dc	4	1
20	click on 10	0.083	10	CG1	delay	1	dl	5	1
21	click ok	0.233	10	CG1	delay	1	db	0	0
22	double click on axis title	0.05	10	CG1	delay	1	dc	4	1
23	click on 10	0.167	10	CG1	delay	1	dl	5	1
24	click ok	0.033	10	CG1	delay	1	db	0	0
25	double click on Title	0.617	10	CG1	delay	1	dc	4	1
26	Position cursor in edit window	0.033	10	CG1	delay	1	pc	5	1
27	delete text	0.033	10	CG1	delay	1	ed	0	0
28	click ok	0.067	10	CG1	delay	1	db	2	0
29	double click on Title	0.167	10	CG1	delay	1	dc	4	1
30	Position cursor in edit window	0.05	10	CG1	delay	1	pc	5	1
31	delete text	0.033	10	CG1	delay	1	ed	0	0
32	enter Text	0.05	10	CG1	delay	1	ed	0	0
33	click ok	0.367	10	CG1	delay	1	db	2	0
1	double-click Data icon	0.067	10	CG1	delay	2	dc	1	0
2	Select Graph from menu bar	0.217	10	CG1	delay	2	mb	8	0
3	Select line graph	0.183	10	CG1	delay	2	mm	12	0
4	chose1. axis label	0.217	10	CG1	delay	2	dl	4	2
5	chose 2. axis label	0.033	10	CG1	delay	2	dl	4	2
6	click ok	0.033	10	CG1	delay	2	db	4	0
7	Click and drag legend to new location	0.717	10	CG1	delay	2	dd	4	2
8	double click on legend text	0.267	10	CG1	delay	2	dc	4	2
9	click on bold in d-box	0.067	10	CG1	delay	2	dl	9	0
10	click ok	0.183	10	CG1	delay	2	db	2	0
11	double click on legend symbol	0.683	10	CG1	delay	2	dc	4	2
12	select line	0.1	10	CG1	delay	2	dl	5	0
13	select symbol	0.15	10	CG1	delay	2	dl	40	0
14	click ok	0.05	10	CG1	delay	2	db	2	0
15	double click on title	0.3	10	CG1	delay	2	dc	4	1
16	click on bold	0.067	10	CG1	delay	2	dl	9	0
17	click on 18	0.083	10	CG1	delay	2	dl	5	2
18	click ok	0.017	10	CG1	delay	2	db	2	0
19	double click on axis title	0.35	10	CG1	delay	2	dc	4	1
20	click on 10	0.067	10	CG1	delay	2	dl	5	1
21	click ok	0.067	10	CG1	delay	2	db	0	0
22	double click on axis title	0.05	10	CG1	delay	2	dc	4	1
23	click on 10	0.067	10	CG1	delay	2	dl	5	1
24	click ok	0.067	10	CG1	delay	2	db	0	0
25	double click on Title	0.083	10	CG1	delay	2	dc	4	1
26	Position cursor in edit window	0.033	10	CG1	delay	2	pc	5	1
27	delete text	0.033	10	CG1	delay	2	ed	0	0
28	click ok	0.1	10	CG1	delay	2	db	2	0
29	double click on Title	0.2	10	CG1	delay	2	dc	4	1
30	Position cursor in edit window	0.05	10	CG1	delay	2	pc	5	1
31	delete text	0.017	10	CG1	delay	2	ed	0	0
32	enter Text	0.05	10	CG1	delay	2	ed	0	0
33	click ok	0.233	10	CG1	delay	2	db	2	0

APPENDIX G

VISUAL GUIDE FOR ESTIMATING THE PROBABILITY OF INTERFACE 'TROUBLE'


Direct Manipulation
No Labels
Menus and Dialog Boxes
Labels Provided
Number of Objects
dragging
double-clicking
selecting of objects
tool selections
finding appropriate editing fields
menu-bar
click on hidden list
select menu item
click on radio button in dialog-box
select nested menu item
edit text
select from unhidden list in dialog-box
click on buttons in dialog-box
Good Label
label is the same or synonym as user's goal
1
5
10
15
20
25
30
35
40
45
not applicable
Poor Label
requires inference or is not linked to user's goal at all
0
5
10
15
20
25
30
35
40
45
Legend:
= little difficulty,
= great difficulty