



Computers and learning
by Lyndon Cary Marshall

A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Education
Montana State University
© Copyright by Lyndon Cary Marshall (1995)

Abstract:

Although computer systems have been part of our lives for more than three decades, most individuals are still looking for ways of integrating computers into their life. Many who work with computers frequently question why people react to computers as they do. Why do some like computers, learn them effectively, and do an excellent job of working with them, while others never seem to like computers or work well with them? Learning styles may be an answer. Therefore, the purpose of this study was to investigate the nature of the relationship between learning style and preferences that were held for different types of computer software.

The design for this research was a case study which involved the collection of both quantitative and qualitative data. The primary method for collecting data involved a series of interviews of computer users and observations of these individuals while using different computer software packages. Particular attention was paid to what programs were liked, why particular programs were preferred, and how individuals worked with different computer programs. Additionally, Kolb's Learning-Style Instrument was used to help classify individuals into different categories for the learning of new concepts.

The results of this study helped to confirm the mixed results that researchers have found regarding the effects of learning style on the way in which individuals work. Learning style was not a significant predictor of preferences for different kinds of user software. However, learning style was a significant predictor of how people talked about their work with computer systems. This result is important to both computing teachers and professionals in the computing field. Both have a need to work with the computer novice. This identification of individuals who are more inclined to talk about computers can help in the peer learning process.

Other significant findings from this study highlighted possible cultural issues that can affect the way in which individuals work with computers. Furthermore, gender based differences can affect individual perceptions of the computer in the workplace. These are issues that can affect the adult educator who has to teach technology.

COMPUTERS AND LEARNING

by

Lyndon Cary Marshall

A thesis submitted in partial fulfillment
of the requirements for the degree

of

Doctor of Education

MONTANA STATE UNIVERSITY
Bozeman, Montana

January 1995

D378
M3569

ii

APPROVAL

of a thesis submitted by

Lyndon Cary Marshall

This thesis has been read by each member of the graduate committee and has been found to be satisfactory regarding content, English usage, format, citations, bibliographic style, and consistency, and is ready for submission to the College of Graduate Studies.

1-12-95
Date

Gary J. Conte
Chairperson, Graduate Committee

Approved for the Major Department

1/12/95
Date

June Mell
Head, Major Department

Approved for the College of Graduate Studies

1/31/95
Date

R. Brown
Graduate Dean

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a doctoral degree at Montana State University, I agree that the Library shall make it available to borrowers under rules of the Library. I further agree that copying of this thesis is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for extensive copying or reproduction of this thesis should be referred to University Microfilms International, 300 North Zeeb Road, Ann Arbor, Michigan 48106, to whom I have granted "the exclusive right to reproduce and distribute my dissertation for sale in and from microform or electronic format, along with the right to reproduce and distribute my abstract in any format in whole or in part."

Signature

Sydney Cary Marshall

Date

January 15, 1995

ACKNOWLEDGEMENTS.

Many people are due thanks as a result of my doctoral studies. The first who should be acknowledged are Alexandra, my wife; Jessica and Gwydion, my children; and my mother, Isabel Marshall, who all put up with long hours of my absence, both mental and physical. Secondly, I would like to thank my committee, particularly Dr. Gary Conti, for the advice and guidance necessary to move me through this process. Next, I would thank my co-workers, Joe Schopfer and Jim Croft, who tolerated my preoccupation with good humor. Finally, I would thank Susan and Una, Librarians at the College of Great Falls, who managed to get every peculiar piece of information that I requested from them.

TABLE OF CONTENTS

	Page
LIST OF FIGURES	ix
ABSTRACT	x
1. INTRODUCTION	1
Background	1
Problem	11
Purpose	12
Significance of Study	12
Researcher Background	14
Research Questions	15
Definition of Terms	16
Limitations	19
Delimitations	20
Assumptions	20
2. REVIEW OF LITERATURE	21
Learning Styles	21
Hierarchy of Learning Styles	25
Kolb and Learning Style	27
Learning Style and Intellectual Ability	31
Learning Style and Disciplinary Area	32
Criticisms of Learning Style	33
Other Aspects of Learning Style	35
Divided Learning Style	36
Learning Strategy	39
Learning Style and Cognitive Style	40
Computers and Cognition	41
A Brief History of Computing	41
The Psychology of Computer Programming	44
Computer Semiotics	47
Learning Style and Computers	49
Human Factors in Computing	53
Computers, Communication, and Culture	54

TABLE OF CONTENTS--Continued

	Page
3. METHODOLOGY	58
Descriptive Case Studies	58
Procedures	60
Kolb's Framework	62
Interview Questions	67
Kolb Learning-Style Inventory	70
Methods of Data Recording	72
4. FINDINGS	74
Learning Style Assessment	74
Interview and Observation Results	78
Learning Strategies	78
Strategies Unique to Learning Computers	79
Strategies for Learning Different Computer Systems	81
Manuals	82
Technical People and Manuals	84
Improvement of Manuals and Documentation	84
Gender Differences in Learning Strategy with Computer Systems	86
Problem Solving Approaches	87
Teaching Strategies	95
Give Me Only What I Need to Get the Job Done	95
The Big Picture	96
Preferences for Different Types of Software	98
Software Used by Participants	98
Software Preferences	99
Software Analytics	99
Software Features to Be Changed	102
Command Line Interfaces	103
Graphical User Interfaces	106
Menus	110
Why Individuals Dislike Software and Its Features	111
Mode of Interaction	113
Expectations	115
Advantages	116
Disadvantages	117

TABLE OF CONTENTS--Continued

	Page
5. SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS	120
Summary	120
Conclusions	122
Conclusions for Teaching and Learning	122
Conclusions for Computers	123
Discussion	124
Teaching Strategies	124
Assessing Student Learning Preferences .	125
Approaching the Subject	126
Problem Solving Approaches to Computers .	126
Flaws in Computer Manuals	128
New Software Architectures	128
Use of Intelligent Agents to Improve	
Software	129
Appropriate Use of Existing Computer	
Technologies	133
Approaches to Systems Development	136
Methods and Tools for Systems Analysis .	136
Computers in the Corporate Culture . . .	140
Recommendations for Further Research	145
The Future	147
REFERENCES CITED	149

LIST OF TABLES

Table		Page
1.	Study participants in each learning style category	78
2.	Study participants in each category of computer use and problem solving	89

LIST OF FIGURES

Figure		Page
1.	Kolb's Experiential Learning Model	28
2.	Kolb's Learning Styles	63

ABSTRACT

Although computer systems have been part of our lives for more than three decades, most individuals are still looking for ways of integrating computers into their life. Many who work with computers frequently question why people react to computers as they do. Why do some like computers, learn them effectively, and do an excellent job of working with them, while others never seem to like computers or work well with them? Learning styles may be an answer. Therefore, the purpose of this study was to investigate the nature of the relationship between learning style and preferences that were held for different types of computer software.

The design for this research was a case study which involved the collection of both quantitative and qualitative data. The primary method for collecting data involved a series of interviews of computer users and observations of these individuals while using different computer software packages. Particular attention was paid to what programs were liked, why particular programs were preferred, and how individuals worked with different computer programs. Additionally, Kolb's Learning-Style Instrument was used to help classify individuals into different categories for the learning of new concepts.

The results of this study helped to confirm the mixed results that researchers have found regarding the effects of learning style on the way in which individuals work. Learning style was not a significant predictor of preferences for different kinds of user software. However, learning style was a significant predictor of how people talked about their work with computer systems. This result is important to both computing teachers and professionals in the computing field. Both have a need to work with the computer novice. This identification of individuals who are more inclined to talk about computers can help in the peer learning process.

Other significant findings from this study highlighted possible cultural issues that can affect the way in which individuals work with computers. Furthermore, gender based differences can affect individual perceptions of the computer in the workplace. These are issues that can affect the adult educator who has to teach technology.

CHAPTER 1

INTRODUCTION

Background

The last few years have witnessed the advent of the information society. This information society is manifested in many different ways. Computers and communications technologies, the driving force behind the development of the information society, are becoming a pervasive part of our daily lives. For example, certain aspects of computer technology have become such an integral part of our lives that we expect and demand the benefits of computer systems. People have come to expect the following services in the everyday workings of their lives: dealing with computer-based automatic teller machines on a 24-hour basis, having groceries checked-out more rapidly and more accurately through the use of computer bar-code readers, obtaining up-to-the-minute weather projections because of sophisticated weather projection computers at the National Weather Service, and tracking monetary information generated by computers at the different financial centers around the world.

The expectation of the benefits from computer systems is all the more pervasive in the work place. This can be readily seen from the fact that at present more than 60% of workers are considered to be individuals whose primary purpose is the creation, manipulation, retrieval, and storage of information (Whitten, Bentley, & Barlow, 1989, p. 33). These individuals are commonly described as knowledge workers. Computer systems are very much an integral part of the work that is expected from knowledge workers. Entry and retrieval of customer information, use of a word processor to type in memos and reports, and the use of computers to track products sold by a company are all common activities for individuals in the work place.

Given the increasing importance of computers in both one's daily and professional lives, it is not surprising to see that increasing levels of private and governmental resources are being devoted to the development of computer software. The level economic commitment to the development of computer systems is reflected by the \$70 billion worldwide that was spent on computer hardware and software in 1990 alone (Cringely, 1992, p. 4).

However, with increasing resources comes increasing demands for performance in the computer industry. This is particularly true with software. Purchasers of software are increasingly impatient with software that does not work right and with software that may work right but that is so

inconvenient to use that it does not matter if it works right. There is substantial reason for this impatience. In a study of a series of software projects developed for just one government department, \$6.8 million was spent for the programming of software. Of the software that was developed from these contracts, only approximately \$100,000 of it was used as delivered. The vast bulk of the software, \$3.2 million worth, was never delivered; \$1.3 million worth of software had to be abandoned or reworked; and \$.2 million worth of programs were used after substantial changes (Ince, 1988, p. 4).

Many explanations are offered for this poor performance in the software development industry. In some instances, developers simply do not have a complete understanding of the problem that they are trying to solve with their programs. In other instances, poor software development and testing methods were employed. However, in a large number of instances, software does not do the job because it works in a way that is either inconvenient or unappealing for the people who are supposed to work with it (Ince, 1988, pp. 1-19).

A major component of the effectiveness of software is represented by the nature of the user interface or shell. This controls the way in which users interact with computers. Since the nature of the interactions that people have with computers will be based on the nature of

the user interface, the success of computer software is frequently based on user preferences for different types of software interfaces.

The importance of user interface preference in the acceptance of computer systems is highlighted by a number of pieces of evidence. First and most significant is the development of entire journals which are devoted to understanding the nature of user preferences for software and which are devoted to the development of more effective computer software interfaces. The most prominent of these journals is SIGCHI Bulletin (for Special Interest Group for Computers and Human Interaction) from the Association for Computing Machinery.

A second piece of evidence which highlights the importance of user acceptance of software as a factor in the success of computer information systems comes from Merle P. Martin's text, The Analysis and Design of Business Systems (1991). In this book, Martin states:

Note that designers have no control over the end-user component. Yet if designers are aware of factors describing the end-user, they can design systems to compensate for negative aspects and enhance positive aspects . . . It is no longer appropriate to force the end-user to learn difficult hardware and software systems. (p. 237)

Furthermore, in this same section of text, Martin states that psychological factors such as the way in which information is processed by the individual and the individual's psychology or attitude can be very important

factors in the acceptance of software systems (pp. 215-218).

Therefore, given the scope of the investment that many organizations make in computer hardware, software, and training, it is critical to have an understanding about the reactions to computer systems of those individuals who will be using them heavily. Understanding the underlying reasons for an individual's reactions will help to insure that expensive information systems are effectively installed and utilized. This will help to avoid the situation, often cited in systems analysis and design texts, of computer systems not being used for any apparent reason (Ince, 1988, p. 4)

The topic of user preferences is also important because it relates to the teaching of the subject of computers. Since people tend to have a preference for those things with which they work the most effectively, the issue of cognitive style (which seeks to explain how different individuals input and process information differently) is tied to the issue of user software preferences. This is a consequence of the fact that factors such as cognitive style can affect the effectiveness with which individuals can master the subject (Barba, 1990; Foreman, 1990; Sharma, 1987). This should affect teachers of computers in academic settings as well as in industrial settings (where it is particularly

important to teach adults how to use new software as part of the systems development process).

However, despite the work that has been done to investigate the relationships among cognitive style, the ability to learn computers, and the preference that one has for different types of user interfaces, little work has been done to try to understand the underlying reasons that different individuals with different cognitive styles prefer one type of user interface from another. Since the interface is the tool by which users interact with computer systems, this is a significant omission.

Much work has been done in the field of computers to try to relate cognitive style to the ways in which individuals work with computers (Chu & Spires, 1991; Clariana & Smith, 1988; Cline, 1991; Ellis, 1989). The general issue of cognitive style in the design of information has received much attention. Currently, it is argued that system design must be adjusted to the cognitive styles of the individuals for which the system is being constructed (Cline, 1991, p.1). This is a major method for insuring the development of truly effective information systems. Additionally, tools such as the Embedded Figures Test and the Meyers-Briggs Type Indicator can be used by systems analysts to assess the cognitive style of computer users. By basing the design of a system on cognitive factors discerned in the user community, the overall

functionality of a computer system can be improved. Furthermore, designers can be more certain that the system more effectively meets the cognitive needs of users. Enhancing the functionality of systems can be based on features such as the availability of different computer interaction formats so that users can choose among them and by developing systems with the ability to customize themselves to the nature of the users who are interacting with them. (Cline, 1991)

Other similar arguments have also been offered. These arguments basically state that human cognitive processing must be considered in the development of information systems. This approach to the design and development of computer systems is sometimes referred to as "Cognitive Engineering" (Norman, 1987, chap. 1). In any event, most arguments that are made regarding the effects of cognitive processing on the systems design process suggest that to insure a truly effective software system designers should constrain themselves to design programs that meet the psychological needs of users (Fischer & Lemke, 1988).

Not only has the idea of the application of cognitive style to system design been discussed extensively in books and journals, but the issue of interface design and its relationship to cognitive processing has also been considered (Cline, 1991; Grabinger, 1989; Mandel, 1994). It has been concluded that the appearance of software

screens is not that important to the effectiveness of the software. However, the individual making this conclusion also concluded that the sequences of processes that are present in software should be amenable to the internal processes that go on in the human mind (Grabinger, 1989).

Another study concluded that a recognition of cognitive processes is important in the teaching of computer-oriented courses. For example, the shifts in learning styles that have occurred in students over a period of time when they are exposed to Computer Based Training (CBT) systems have been charted. As a result of being exposed to Computer Based Training, learners have demonstrated a distinct shift toward a preference for more concrete experimentation and for more reflective observation as a result of the CBT experience. (Clariana & Smith, 1988)

Furthermore, direct research has been done that reflects the effects of cognitive style on the ability of individuals to learn the subject. The ability to succeed in computer programming classes is often reflected in the level of impulsivity or reflectivity that is present in the individual who is learning to program. (Van Merriënboer, 1990)

Therefore, the subject of cognitive processing and its effects on the nature of human-machine interaction has received substantial academic attention. From this

attention, it can be concluded that it is important to understand the nature of mental processes that go into interacting with computers. Indeed, one researcher approached the subject from the standpoint of the individual who is interested in effectively structuring the manner in which social entities, such as companies, interact with the functionality of the computer system (Zuboff, 1988). This process of constructing effective modes of computer and social organization interaction, systems analysis and design, is increasingly devoted to an understanding of the modes of cognitive processing of the individuals who will be working with a computer information system. This blossoming interconnection is reflected in a number of ways. However, it is most effectively demonstrated by the fact that texts on systems analysis and design include chapters which are devoted to recognizing the potential effects on the design process of the varying cognitive processing styles among the people with which the analyst will be working. Examples of this trend toward the recognition of the importance of cognitive processing include Martin's (1991) Analysis and Design of Business Information Systems (chap. 7, pp. 215-238) and Principles of Information Systems Management (Ahituv & Neumann, 1990, chap. 2, pp. 35-72).

Despite the research that has been done in this area, some basic questions have not been asked. The answers to

these might be useful in understanding the basic underlying structure of the process of human-computer interaction. These questions relate to the user interfaces for software and the effectiveness of users in working with them. One such question relates to determining why an individual with a certain cognitive style prefers to work with a certain type of computer user interface.

While there are numerous reasons for needing to know why users who process information in varying ways prefer a certain user interface, there is one underlying and very compelling justification: economics. Systems will not be successful if they are not supported by the user community. Since the user interface is the primary element by which users interact with the computer system, it is critical to understand the underlying processes for liking or for disliking a particular interface.

This understanding of the underlying processes for liking or disliking user interfaces is demonstrably different from the kinds of research into the nature of user interface preferences that has been done before. Existing research has basically established that there is a relationship between interface preference and cognitive style and that designers need to provide for the availability of multiple interfaces to accommodate different human information processing styles. Therefore, instead of continuing work on establishing a relationship

between human information processing styles and interface preference, the focus of further inquiry should begin to address the following subject: why do learners of a certain style prefer a certain type of user interface?

In pursuing the question of why computer users prefer a particular type of user interface, broad categories of user interfaces will have to be identified. User interfaces fall into two primary categories: command line interfaces and graphical user interfaces (Deitel, 1990; Mandel, 1994, pp. 1-28).

Graphical User Interfaces are characterized by the use of an input-output device, called a mouse, to point to different programs that the user wishes to run. Examples of operating systems that use GUI's include Windows, OS-2 Presentation Manager, MacIntosh System 7, and X-Windows.

Command Line Interfaces are characterized by the appearance of a system prompt (for example, A> or C>) at which the user is expected to type in a syntactically correct command line to get programs to execute. Examples of Command Line Interfaces include the various versions of MS-DOS, PC-DOS, and Unix.

Problem

Two major themes are revealed in examining the relationship between cognitive style and preferences for software. First, user preferences will affect the

acceptance and the success of software systems. Secondly, cognitive style will affect the success that individuals have in working with and in learning software systems. However, the full nature of the relationship between preference for a particular type of software and cognitive processing style remains clouded. Why people prefer what they prefer remains a mystery.

Purpose

Therefore, the purpose of this study was to discover underlying reasons that individuals with different cognitive styles prefer each of the major different types of user interfaces. In this regard, this study can serve to provide additional depth of understanding of both cognitive processing and user interface preferences.

Significance of Study

Underlying information about why an individual with a certain cognitive style will prefer a certain type of user interface can prove to be truly useful to professionals in a number of areas. First, in knowing the reasons that individuals with a certain type of cognitive style prefer a user interface, educators can be better able to aid students to overcome difficulties which are engendered by the preference for a particular type of user interface associated with a particular cognitive style. Second,

educators in understanding the underlying reasons for preferring a particular type of user interface can be better able to select teaching tools that match the cognitive styles of the students with which they are working. Third, by being aware of the reasons an individual with a different cognitive style prefers a certain type of user interface, software and systems designers can be better able to develop operating environments that meet the needs of users with different cognitive styles.

Also, advantages would exist for educators in understanding why people with different cognitive styles prefer working with a certain user interface. In understanding this, a wide range of tools can be placed in the hands of the instructor. Among these tools is a fundamental understanding that there is a crying need to not only use the right software tools for the right audience but also to attack and emphasize the ideas and subjects that are of particular concern to the individual with a certain way of processing information.

Understanding the nature of interface preference and cognitive style for processing information will offer a number of other distinct benefits for computer specialists. First, this would give computer professionals who design the structure of information processing in organizations, systems analysts, some tangible information about why

individuals like or do not like a particular software user interface. This understanding could offer the analyst a number of advantages. First, it would permit the analyst to select software tools that are appropriate to the environment and to the preferences of the users with which the analyst will be working. Second, it would permit the analyst to direct the user training program to focus on the issues that most concern the people who will work with the system. Third, having some idea of the reasons behind user preferences for software interfaces could give the computer systems designer a more effective concept of when it is appropriate to use a particular interface in an individual organizational setting. Fourth, an understanding of the reasons for users preferring interfaces would open systems analysis and design to a fuller range of tools related to the evaluation of user cognitive style for understanding the organizational dynamic for interacting with computer systems.

Researcher Background

This study has its roots in the background of the researcher who carried out this study. As such, it reflects ideas that have been in development for the more than 20 years that this individual has been a computer professional.

In watching people work with computer systems, it has been very apparent to this researcher that some very intelligent people, who should have little difficulty in working with computer systems, suffer from high levels of anxiety when interacting with computers. One often hears the complaint that individuals are afraid to touch specific buttons because of the fear that the machine will explode.

There has to be a reason for this kind of fear. It must lie somewhere in the way in which computers and software are constructed and used. It must lie somewhere in the way in which computer people interact with new users of computers.

In any event, given the significant time that this researcher has had to spend over the years calming and reassuring the neophyte computer user, this individual has developed an article of faith that understanding the root causes of the fear that many carry into their interactions with computers will help combat the growth and development of the distaste, distrust, and loathing that many bring to their initial work with computers.

Research Questions

Several research questions were addressed in this study. These questions included:

1. How do users with different cognitive styles react to command line interfaces?

2. How do users with different cognitive styles react to graphical user interfaces?
3. What do users with different cognitive styles identify as the best and worst features of the different types of user interfaces?
4. Why do users like or dislike certain features of a particular interface?
5. How do users with different cognitive styles interact with the different interfaces?

Definition of Terms

Cognitive Style: The way in which an individual tends to process the new ideas and information that are encountered in the environment. There are different approaches to Cognitive Style. Some tend to take a look at sensory modalities while others emphasize internal information processing approaches (Nunney, 1978, p. 50).

Command Line Interface (CLI): This type of software interface employs sequences of commands that are typed when the software indicates that it is ready to receive commands. The sequence of commands must adhere to a strictly developed command structure or syntax. Examples include MS-DOS, Unix, and CP/M (Mandel, 1994, chap. 6).

Directories: A segment of a storage device such as a disk or a hard disk that is oriented toward storing similar information. The need for directories is based on the fact that disks as used in modern computer systems tend to be very large, with typical storage capabilities exceeding 100

megabytes. Navigating through directories is a major functional difficulty in Command Line User Interface environments.

Function Keys: These are keys that frequently appear on computer keyboards with legends such as "F1" or "F8." They are usually used to activate specific program or command operations.

Graphical User Interface (GUI): A type of software interface that is characterized by the use of the mouse and by the use of graphical images or icons to represent the programs that can be run. Programs are activated by using the mouse to move a pointer over the icon that represents the program that is to be run. Examples include Windows, OS/2 Presentation Manager, and MacIntosh System 7 (Mandel, 1994, chap. 7).

Groups: A concept from Graphical User Interfaces. It is a collection of related icons that exist under a single controlling icon. The icons in a group represent programs that have a similar purpose or a related function. This is related to the older operating system concept of directories.

Icon: A program symbol in operating environments such as Windows. Users employ a pointing device called a mouse to activate a program by pointing at the icon and by using the mouse to double-click on the icon of the program that they want to run.

Learning Strategy: The basic set of tactics that an individual uses in approaching a learning task. This tends to be relatively more changeable than learning style. Learning strategy will vary from learning task to learning task.

Learning Style: Similar to Cognitive Style, it relates to the way in which individuals tend to learn new information that they encounter in the environment. It also refers to individual preferences for how information is presented to them. For example, some individuals learn best by experimenting while others learn best by observing (Nunney, 1978, p. 50).

Menu: A list of options that is displayed by programs for users from which they make choices. Menus can appear as part of the operating system, as part of a shell in which users choose what programs that they want to run, or as part of a program such as word processor in which users choose what part of a program that they want to run (Mandel, 1994, chap. 6).

Mouse: A pointing device used in Graphical User Interface environments for selecting programs that an individual wants to run.

Operating System: The basic controlling program that enables computer systems to function. Requests to computers to get programs such as word processors to function are usually handled by the operating system. The

user interface is one part of the operating system. Examples of different operating systems include MS-DOS, CP/M, Unix, Windows, X-Windows, and OS/2 (Deitel, 1990, chap. 1).

Prompt: A signal from a program that it is ready to receive commands. In MS-DOS, the prompt is represented by symbols such as C> or A>. The > represents command readiness. The letters represent the currently active disk drive storage device. This is the location from which data or programs will be found.

Shell: This is the face that a piece of software shows to the external world. As such, it is very similar to an interface. However, it is frequently used to refer to programs that are meant to aid in the functioning of Command Line Interface programs. Menu systems for MS-DOS are typical examples.

User: This is an individual who interacts with a computer system by working with computer programs of any sort.

Limitations

Participants were chosen for this study from the local computer-using populace. It is possible that individuals in other regions may react differently to computer user interfaces.

Delimitations

Participants were chosen for this study from a number of different professions. These professions included college professors, college administrators, students, secretaries, engineers, Air Force officers, video technicians, and nurses.

Assumptions

It is believed that the people selected for this study represented a fairly broad spectrum of computer users in ability and knowledge. Further, because of the wide range of learning styles that were represented, it is believed that a very typical range of reasons and approaches to computer use was present in study participants.

CHAPTER 2

REVIEW OF LITERATURE

Learning Styles

While the concept of looking at learning style or the way in which individuals process new ideas and information is relatively new, the idea of classifying individuals according to various different psychological typologies has been a recurring theme among the students of the human mind. One of the oldest and most pre-eminent of these typologies was developed by the great psychologist, C. G. Jung. In his works, he described various personality types that would, according to his vision, affect assorted aspects of personal and social reaction, including approaches to learning and solving problems. Among the different personality types described by Jung (1921) are the Apollonian, the Dionysian, the Epimethian, and the Promethian.

While these personality types focus primarily on broad personality issues, aspects of learning style do intrude in this discussion of different types of people. For example, substantial aspects of Jung's work are devoted to personal approaches to problem solving and to whether or not an

individual looks inward (introversion) as a way of solving problems or outward (extroversion) in problem solving. Furthermore, Jung devotes much discussion to examining whether an individual relies more on the rational mind or on emotions when dealing with difficulties.

The psychological typology proposed by Jung lay in dormancy for a long period of time. However, in recent years, others have taken up this approach to looking at the human mind and its different facets. One of the most prominent attempts to examine psychological types is the Meyers-Briggs Type Indicator. Essentially, this instrument employs the Jungian categories as the basis for classifying the personality types of individuals (Keirsey & Bates, 1978, pp. 3-4).

However, for a number of decades, the idea of examining human temperament and aptitudes from the perspective that there are basic, underlying, built-in reasons that individuals differ in how they do tasks than do others fell into disrepute. Belief in a new form of scientific objectivism called Behaviorism started to grow. Around the beginning of the Twentieth Century, a group of psychologists began to assert that the nature of the human mind could be directly understood through the objective lens of the external study of an individual's response to stimulation. In this regard, some of the more radical members of the Behaviorist school held that internal

individual differences, preferences, and abilities were totally irrelevant. The only thing that should be considered is the nature of the experiment and the person's reaction (Hunt, 1993, pp. 242-275).

Many researchers found the viewpoint of the Behaviorist model of the human mind to be too limiting. Consequently, a new approach to understanding the mind called Cognitivism started to develop. In contrast to the approach advocated by the Behaviorists, Cognitivists believed that internal differences between humans in terms of preferences, abilities, and styles of learning were the most important concepts that could be studied about the human mind. In fact, the interest in the internal structures of the human mind has been carried to such a point that there is a tendency among many students of psychology to use a computer analogy in discussing the mind and to describe the mind in terms of components such as input-output (perception), storage (memory), programming (instinct and learning), and processing (Hunt, 1993, pp. 511-558).

Despite a tendency not to agree on anything, students of psychology have been coming to a conclusion that the human mind is more complicated than previously believed. Consequently, strongly held positions have had to be modified or enlarged.

In fact, each one of the principles confidently enunciated by Skinner in The Science of Learning and the Art of Teaching now turns out to be untrue--at least in as general a sense as he believed at that time. (McKeachie, 1974, p. 186)

In acknowledging the fact that the human mind is much more complex than had been imagined by many and particularly the Behaviorists, numerous different approaches to attempting to understand how and why people learn have emerged. Some of these different schools of psychology and learning include social psychology, perception psychology, motivational psychology, cognitive psychology, and psychotherapy. Each of these schools has a different approach and set of issues that they consider to be important in understanding the human mind. However, certain common themes emerge in considering the essential directions of these different approaches to psychology. Some do not study learning theory at all or only consider it as a distinctly peripheral issue. Others have shifted their emphasis from studying the individual patterns in learning to studying "between-group differences such as racial differences, sexual differences, and social class differences" (Ivey, 1992, p. 38). Still others have become fascinated with micro-level functioning of the human mind and have considered only the neuro-physiological brain, the collective functioning of a series of conjoined organs with a common blood supply.

Three words . . . go a long way toward telling the whole story--"mind," "matter," and "intellect." The contemporary view of mind denies the intellect as a distinct faculty--a special power of the human mind that makes it radically different in kind, not just in degree, from the minds of all other animals on earth. (Adler, 1990, p. vii)

The consequence of these obsessions with issues such as the macro (how groups behave) and the micro (how neurons function) is that the study of a very fundamental issue, the investigation of individuals and of individual learning styles, has been left incomplete.

Hierarchy of Learning Styles

In his 1992 dissertation, Ivey discusses an effort to organize the different theories and instruments that have been developed with respect to learning style. The primary issue that was raised with the different instruments was described as "psychometric acceptability." In arriving at this issue of acceptability, the primary criterion was the meaningful collection and reporting of data concerning the reliability and validity of the various instruments that were assessed for the study (p. 38). Applying this standard, the vast majority of the tests were eliminated from further consideration and nine of the tests were placed in a three-level, concentric hierarchy.

The center of the hierarchy was occupied by those theories that are organized around describing the cognitive personality. Tests in this group include Witkin's Embedded

Figures Test and the Meyers-Briggs Type Indicator. The second level of the hierarchy was oriented toward models of the ways in which individuals process information. Kolb's Learning-Style Model falls into this category. The third level concerns instructional preferences or the environment in which the individual works most effectively (p. 39).

In occupying the center of the model and trying to describe cognitive personality, the primary objective of Witkin's Embedded Figures test is to measure the amount of field dependence or field independence that an individual has. The purpose of this measure is to determine an individual's tendency to apply analytical models in the solution of a problem as compared to the person's proclivity for needing global information to be able to understand things.

As with most of the models that made it through the screening tool of having basic reliability and validity information available, Embedded Figures does have some difficulties. Primary among them is the fact that it only assesses part of what it claims to. In fact, it really only determines the amount of field independence that a person possesses. The amount of field dependence that a person is subject to is implied by the lack of field independence (Bonham, 1988, p. 12). Whether field dependence and field independence are complementary in this fashion remains to be seen.

The outermost part of the hierarchy of tools and theories is oriented to the environment in which individuals would like to learn. One such instrument is the Dunn LSI. It focuses on learning environment, emotionality, sociological needs, and physical requirements (Dunn & Dunn, 1978). Another instrument in the second category is the Grasha-Reichmann Student Learning Style Scales which measures the degree to which students are collaborative, competitive, participant, avoidant, dependent, and independent (1974).

A number of instruments fall into the middle category of learning style theories which purport to deal with the ways in which individuals assimilate and process information. One such example in this category is the work of Entwistle (1981). This theory describes two forms of learning. These forms are "deep" and "surface" learning (p. 105).

However, the most famous of the instruments and theories that fall into the middle category are those of Kolb. There are two basic aspects of Kolb's theories. These include the Learning-Style Inventory and the Experiential Learning Model.

Kolb and Learning Style

In describing the Experiential Learning Model, Kolb characterizes his theory as being very similar to the

personality types theory that was originated by Jung. In this regard, he describes his structure as being essentially dialectical in nature because it is composed of tendencies that are essentially in opposition to one another but that come together in the human being because of the controlling nature of the higher intellectual processes (Kolb, 1981, p. 235).

The four aspects of the Experiential Learning Model include Concrete Experience, Observations and Reflections, Formulation of Abstract Concepts and Generalizations, and Testing Implications in New Situations. According to the model, the individual will in any learning situation use varying combinations of the knowledge-building approaches implicit in the paradigm. What combinations are used will depend on varying factors such as the learning situation and the personality of the learner.

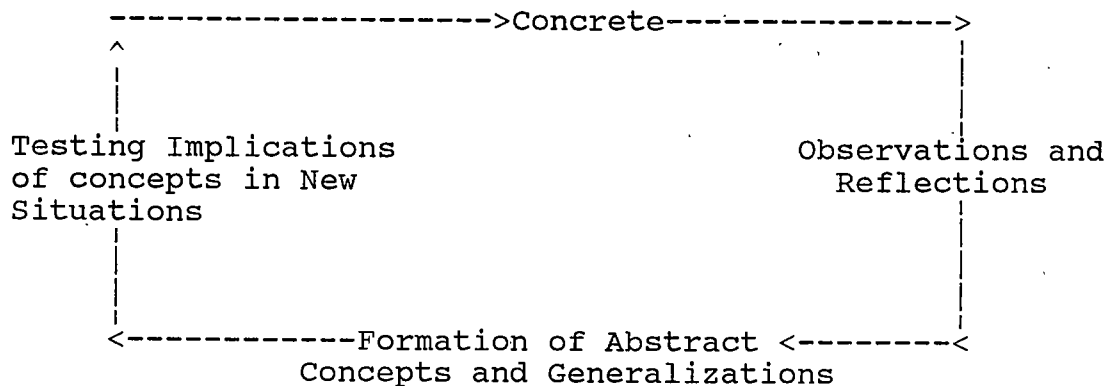


Figure 1. Kolb's Experiential Learning Model.

Kolb's learning styles flow directly from his description of the Experiential Learning Model. Basically, following an information processing approach to understanding how individuals learn, he describes four primary cognitive tasks that an individual must go through in a learning situation: Concrete Experience, Abstract Conceptualization, Active Experimentation, and Reflective Observation. In the Learning Style Model, these tasks are placed on orthogonally situated axes. The vertical portion of the axis represents the two diametrically opposed methods of learning Concrete Experience and Abstract Conceptualization. These two qualities are primarily oriented toward understanding how an individual perceives information.

In contrast, the horizontal dimension of Kolb's Learning Style emphasizes the processing of information. The two elements that describe how individuals process information are Reflective Observation and Active Experimentation.

In Kolb's Learning-Style Inventory, one primary fact that should be emphasized is the fact that both of the scales in the inventory represent scales or spectra. Generally, individuals will not have all of one quality and none of the other on a particular axis. Rather, varying combinations of each of the qualities on an axis will be present in each individual.

There is still a further aspect to the Learning-Style Inventory. The two axes of perception and information processing in the model combine to create four quadrants. These are the learning styles that Kolb seeks to identify. The four distinct learning styles include Convergers, Divergers, Accommodators, and Assimilators.

Kolb describes the different learning styles in some depth. For example, Convergers combine a propensity for Active Experimentation with a fondness for Abstract Conceptualization. These individuals seem to function most effectively in the situation typified by the traditional intelligence test. This is where there is a single correct answer to the question being asked (Kolb, Ruskin, & McIntyre, 1974, p. 30). Kolb suggested that these individuals would function most effectively in areas such as science and engineering.

In contrast, Divergers combine a fondness for Concrete Experience with an orientation toward Reflective Observation. Individuals with this learning style take an entirely different approach to intellectual issues. They are people oriented, like to work in groups, specialize in the generation of ideas during group discussions, and tend to excel in the arts (p. 31).

Accommodators prefer to obtain information through Concrete Experience and to process information through Active Experimentation. According to Kolb, these

individuals are doers and are characterized by a high degree of flexibility. Individuals with this learning style make good sales people (p. 31.).

Assimilators like Reflective Observation and Abstract Conceptualization. These are idea people. They are not necessarily interested in the practicality of ideas (p. 31).

Learning Style and Intellectual Ability

In terms of linking an individual's success with different subjects and learning style preferences, there have been mixed results. On the positive side, it appears that the ability to successfully work with subjects that are based in high levels of abstract conceptualization is associated with a learning style which includes a preference for abstraction. This was demonstrated in a study in which learning style was compared to a student's ability to function with abstraction as represented by effectiveness in operating with experimental science processes. In this regard, those students who preferred work at the concrete level did less well in the test of experimental processes than did students who liked abstraction (Nakayama, 1988).

In contrast, a number of studies have not found any relationship between intellectual ability and results on learning style. The studies that have shown no

relationship have occurred in a number of different academic settings and with different academic disciplines.

One example of a study that found no association between learning style and subject success came from an examination of a collection of freshman college students. In this investigation (Davis, Murrell, & Davis, 1988), the learning style as determined by the Kolb Learning-Style Inventory was compared to the grade that a student obtained in a course. Further, the student's learning style was compared to the learning style of the instructor. In this regard, it was found that there was no relationship between grade and learning style and that there was no significant difference in the grades of students whose learning style matched the learning style of their instructors.

Learning Style and Disciplinary Area

While there is no strong reason to associate learning style and the ability to work in a particular subject, there does seem to be a tendency for individuals with similar academic backgrounds to have similar learning styles (Kolb, 1981, pp. 240-244). For example, individuals who have been trained as engineers or computer scientists tend to be what Kolb would describe as Convergers or individuals with a preference for Active Experimentation and Abstract Conceptualization. Similarly, individuals who have been trained in business, accounting, or finance tend

to be Accommodators because they have a preference for Active Experimentation and Concrete Experience.

Researchers have reported that work in particular academic areas can cause individual learning styles to shift. In one study (Clariana & Smith, 1988), students were tested for learning style before an extensive exposure to a mathematics course. Afterwards, they were tested again. A shift toward a preference for Reflective Observation occurred.

Results such as this can lead to a number of conclusions. First, if learning style can shift due to some external influence, then it is probably true that learning style is not a permanently wired-in aspect of how the human mind processes information. Secondly, understanding how individuals learn is more complex than is even imagined by the advocates of learning style. In fact, in order to improve teaching and learning in all disciplines, "more is needed than a knowledge of a student's learning style" (Conti & Welborn, 1986, p. 22).

Criticisms of Learning Style

There are a number of criticisms of learning style. These criticisms come from a number of different individuals and have a number of different bases. Sometimes the criticisms are fair, but in some instances they are not.

One area in which learning style receives substantial criticism lies in the fact that there are often validity and reliability issues that are legitimately raised with respect to the instruments that are supposed to measure them. For example, of 21 different learning style instruments that were examined for psychometric acceptability only 9 were accepted (Ivey, 1992, p. 38).

Another area in which learning style instruments have been criticized comes from them potentially suffering from cultural or economic biases. In this regard, a number of the instruments tend to be biased in favor of those individuals who are from the upper portion of the social, educational, and economic spectrum. In particular, Kolb's instrument has been directly condemned for using normative groups that are not so typical of the population as a whole (Sewall, 1986, p. 25).

However, a number of the critiques of learning style and learning style inventories have not been completely well-founded. For example, in criticizing Kolb for having an instrument that is culturally biased, some authors miss the fact that this is a criticism that can be levied against many forms of cognitive ability testing (Helms, 1992, pp. 1083) and that this has been a very egregious problem for intelligence testing for some time (Gould, 1981).

In another example of criticisms that may not have been appropriate, some commentators attacked learning style and Kolb for lapses that were present in their own methodologies. One study purported to investigate the validity of Kolb's instrument by using an instrument that, in turn, had not been validated and that may not have been reliable (Fox, 1984, pp. 83-84).

Another study (Ferrell, 1983) was oriented toward furthering a certain viewpoint of cognition. This approach to cognition was enunciated by Keefe (1982) and asserts that learning style is based on three components, which consists of cognitive, affective, and physiological behaviors. In Ferrell's paper, four learning style instruments were examined (Grasha and Reichmann's SLSS, Kolb's LSI, Johnson's DMI, and Dunn's LSI). Each of the different learning style instruments was strongly criticized by Ferrell (1983) for not following Keefe's framework.

Other Aspects of Learning Style

There have been many different approaches to trying to understand the different ways in which human beings learn. One of the most prominent of these is called Multiple Intelligence Theory. In this framework, it is believed that there are different aspects to intelligence and that these different aspects are possessed in varying degrees by

different individuals. The different intelligences include linguistic, musical, logical-mathematical, spatial, bodily-kinesthetic, and interpersonal (Gardner, 1993).

According to the advocates of Multiple Intelligence Theory, this concept is not really the same thing as learning style because it does not attempt to explain or discuss different preferences for how information is obtained or processed. Instead, what it attempts to explain is the differing abilities that individuals have in different subject areas (Gardner, 1993, pp. ix-xxv).

Divided Learning Style. Some students of learning style take an approach similar to Gardner in advocating that there are different aspects to intelligence. However, in distinct contrast, researchers such as Keefe identify only three basic aspects to intelligence (cognitive, affective, and physiological), and they have studied ways of assessing particular individual styles within the three primary aspects of intelligence (1982).

Following this approach, Keefe has identified a number of instruments that fit into the different aspects of intelligence. For example, a number of instruments are designed to identify the cognitive aspect of intelligence. Included in this area is the Edmond's Learning Style Identification Exercise (ELSIE) which is "concerned with the ways students internalize individual words" (Keefe,

1982, p. 45). In accomplishing this particular objective, ELSIE is a method for detecting the perceptual modes that an individual has.

Also falling into the category of assessing the cognitive aspect of intelligence are the Group Embedded Figures Test (GEFT) developed by Witkin (1978) and Gregorc's Style Delineator (Gregorc, 1979, pp. 234-236). In Gregorc's model, information processing is the primary idea that is examined. As such, this knowledge handling is described as having two dimensions, abstract vs. concrete and random vs. sequential. Therefore, individuals will have a learning style that falls into a combination of the two dimensions. For example, an individual might be classified as Abstract Random (AR), Abstract Sequential (AS), Concrete Random (CR), or Concrete Sequential (CS).

In the Group Embedded Figures Test, individuals are asked to find shapes inside increasingly complex geometric figures. This ability to find the shapes determines how Field Dependent or Field Independent that an individual is (Oltman, Raskin, & Witkin, 1971). Field Dependency is a measure of how much an individual needs associated and background information to understand how something works (Witkin, 1978).

Several different instruments have been identified to assess the affective aspect of intelligence. One such instrument is the Paragraph Completion Method (Hunt,

Butler, Noy, & Rosser, 1978). With this instrument, respondents are given statements about which they are supposed to write additional sentences which reflect their feelings about the subject. The manner of completion of the paragraphs tells how much structure individuals need in their learning process (Keefe, 1982, p. 49).

Another instrument that seeks to examine the affective aspects of intelligence is the I/E Scale (Rotter, 1966). This questionnaire presents a series of alternatives that describe events in a person's life. An individual taking this test is expected to select from one of the alternatives in each scenario. Each alternative will either represent an internal or external locus of control of the individual's actions (Keefe, 1982, p. 49).

Physiological styles are "biologically based modes of response that are founded on sex-related differences, personal nutrition and health, and reaction to the physical environment" (Keefe, 1982, p. 49). Several instruments address the different possible learning styles which are implicit in this aspect of intelligence. Among them are Dunn's Time Questionnaire which permits an individual to chart the preferred working times throughout the day (Dunn, & Dunn, 1978).

As can be seen, there are many different aspects to intelligence, and "no current learning style instrument provides a truly comprehensive assessment of the cognitive,

affective, and physiological domains of learning style (Keefe, 1982, p. 53). Consequently, much room remains for study and understanding in this field.

Learning Strategy

There is a vast richness to the way in which individuals deal with learning and problem solving. One aspect of problem solving is learning strategy. While learning style is considered to be a relatively persistent trait in that it is considered to be a fairly permanent part of the way in which a person deals with the world, learning strategy is the differing skills that an individual brings in the solution of a particular problem or in learning a particular subject (Conti & Welborn, 1986, p. 20). In dealing with the subject of how well different individuals adapt to different computing environments and to varying computer programs, this subject will inevitably come up.

Metacognition, metamotivation, memory, resource management, and critical thinking are the five key components of learning strategy (Conti & Fellenz, 1992). Metacognition is the self-examination of internal processes that one uses in learning. Metamotivation is the internal processes that an individual uses to describe the importance of learning a particular subject. Resource management is the propensity that one has for effectively

using various resources to help learn a particular subject. These resources could include friends, experts, or books, for example. Critical thinking is the ability that an individual has for skeptically examining the context and background of the concepts that are presented to them. Memory is the ability that one has to retrieve information that has been assimilated.

Learning Style and Cognitive Style

There is a tendency to link the concepts of learning style and cognitive style. In many ways this is legitimate because the two concepts are intimately related.

However, despite the closeness of the connection, there are some important differences between the two concepts. In many ways, learning style is a much broader concept than is cognitive style. Learning style addresses a broad range of factors that can affect the effectiveness of the learning process. According to Keefe (1982, p. 44), these factors include the cognitive, affective, and physiological traits that can affect the learning process. In contrast, cognitive style is directly focused on only those things that affect the processing of information. This would include both the input of information (sensory modalities) and the way in which information is processed by the brain (pp. 44-45).

However, the similarities between the concepts outweigh the differences. This could be due to the dominance that cognitive style has in the various learning style models (Keefe, 1982, chap. 1).

Computers and Cognition

A Brief History of Computing

Understanding the problems and opportunities that individuals have in interacting with computers is at least partially based on a comprehension of the developmental history of computers, and microcomputers in particular.

Until the late 1970's, the dominant form of computer was something called the mainframe. To say the least, these devices were not user-friendly. To get any information from them, they required special operating environments and armies of specially trained programmers (Long, 1991, chap. 2).

The difficult nature of getting information into and out of these early mainframe computers was reflected in the fact that operating these computers was a technical specialist job of its own that required a separate knowledge of a different programming language such as something called JCL or Job Control Language. The purpose of devices such as JCL was very similar to that of modern operating systems in controlling devices and in assisting regular programs in running. (Long, 1991, p. 502)

However, the late 1970's saw the advent of the microcomputer. The basis for their invention, the microprocessor that had been invented earlier in the decade by an engineer named Ted Hoff, was incorporated by a number of engineers as part of kit-based personal microcomputer systems. Later, faced with the increasing popularity of certain types of software systems, notably spreadsheets, manufacturers such as Apple started shipping pre-assembled microcomputer systems to their non-engineer customers. (Ranade & Nash, 1994, pp. 3-7)

During this initial period of computer use, microcomputers definitely reflected their origins in mainframe computers. This was demonstrated by the nature of the operating systems that were used in computers to help run programs. In this period, the microcomputer industry was dominated by an operating system called CP/M for Control Program/Microprocessor. This operating system was very much like its mainframe-based predecessors such as JCL in that it was cryptic, difficult to understand, and required a high level of expertise to be able to use effectively. Casual users were not welcomed in this operating environment where it was extremely easy to lose every file on your disk. (Kildall, 1994, pp. 205-211)

Glimmerings of change started to be felt in the computer industry in the early 1980's with the introduction of the IBM PC. Accompanying the introduction of IBM PC and

spurring its popularity was a program called Lotus 1-2-3. Here was a program that almost everyone who needed to track financial information absolutely had to own. Thousands of IBM PCs were sold on the simple basis that individuals wanted to track their personal and corporate finances by using Lotus 1-2-3. (Williams, 1994a, pp. 417-422)

However, one major problem existed in this environment. The operating system that was used to help other programs, such as Lotus 1-2-3, operate properly was something called MS-DOS or Microsoft's Disk Operating System. In order to keep commands that were compatible to those that were used in CP/M, which dominated the computer market at the time, MS-DOS was designed to look, act, and feel just like its earlier, cryptic cousin. The consequence of this was that computer users still had to be initiated into an arcane language in order to be able to run one simple program. This is the origin of the command line interface. (Paterson, 1994, pp. 212-219)

Not long after the introduction of the IBM PC, other developers of computer systems began to wonder why it was necessary to continue to perpetuate the unfathomable methods of communicating with computers that had been used up to that point. Consequently, the engineers at Apple Computers introduced computers that were based on the use of program symbols called icons, on a pointing tool called a mouse, and on an activation process that involved

clicking a button when you wanted to run a program. This process of running programs had been developed a few years earlier by engineers at Xerox's Palo Alto Research Center (PARC) and had been intended to make it substantially easier for untechnically-oriented people to use computers with ease and facility. This is the origin of the graphical user interface. (Williams, 1994b, pp. 397-412)

Since the development of the two major operating environments, command line interfaces and graphical user interfaces, there has been an ongoing argument in computer circles as to which method is better. Both sides have many adherents and as many different reasons for liking what they like. Some users state that ease of use must be the prime consideration and that graphical user interfaces are superior. Others state that the ability to control all aspects of computer operations is more important and that command line interfaces are better. In any event, it is difficult to reconcile the two groups. One might suspect that the issues are more profound and complex than might be inferred from the surface of the argument.

The Psychology of Computer Programming

One of the seminal works in attempting to understand the psychological aspects of computing is The Psychology of Computer Programming (Weinberg, 1971). While much of the text is devoted to the technical aspects of good computer

programming languages and to the requirements for writing good computer programs, substantial parts of it are oriented toward the social and psychological aspects of computing.

Weinberg makes several major points about the sociology and psychology of programming. First, organizations which are going to depend on computer technology to run a business should insure that software is developed according to the Egoless Programming paradigm. This means that computer programs are developed according to standardized methods that are understandable and that these programs are completely documented. This will help to keep the programming "artist" from developing programs that work elegantly but that are composed of programming statements that nobody else can understand or alter. Organizations which permit their programming staffs to write programs that do not adhere to Egoless Programming will inevitably suffer harmful consequences because individuals die, retire, take other jobs, or get an inflated view of their own self-importance. The inescapable consequence of over-dependence on a single individual is software ending up as useless junk because nobody can fix the problems with the programs (Weinberg, 1971, pp. 47-65). This inevitability of having to alter computer programs was underlined by Brooks (1982) when he wrote, "A program doesn't stop changing when it is

delivered for customer use. . . . The total cost of maintaining a widely used program is typically 40 percent or more of the cost of developing it" (pp. 120-121).

Another major theme of Weinberg is that computer programming is, or should be, a social activity. In this regard, when working on large projects, it is destined that differing individuals will have diverging levels of expertise when working with varying technical projects. Therefore, to work the most effectively as a computer programmer, an individual must have the ability to work competently in groups (pp. 67-93).

Furthermore, Weinberg attempted to identify the psychological characteristics of the effective computer programmer. Among these characteristics include flexibility, adaptability, tenacity, level of intelligence, the ability to tolerate stress, some humility, and a sense of humor. However, it should be noted that Weinberg is skeptical of the use of psychological tests to measure either the personality or aptitudes of programmers because of the "inadequacy of our knowledge of which personality factors play what role in which part of the programming process" (p. 158).

However, one of the most important parts of Weinberg's work is his attempt to identify the characteristics of good computer programs. Among the characteristics identified were efficiency, reliability, and ease of use (pp. 15-26).

Nonetheless, his attempts to identify the characteristics of computer languages that would more effectively fit the cognitive processing patterns of computer programmers may actually be more informative for the typical user of modern complex computer programs. The list of such characteristics includes consistency of operation, extensibility of operation (the ability to get the language to do a variety of things), the compactness of the language, and uniformity of operation (pp. 206-243).

Other writers have identified similar issues in the design of effective software user interfaces. When asked to consult on what should be important in the design of software interfaces, Cognitive psychologists have listed three primary principles of design, including placing users in control, reducing user memory load, and making the interface consistent (Mandel, 1994, pp. 77-94).

Computer Semiotics

Since semiotics is the study of how the meaning conveyed in symbols affects the communications process, one group of researchers in the design of computer software asserts that understanding the nature of effective user interface design should be based on the use and meaning of signs that are inherent in language. Furthermore, these symbols should be based on the social and cultural aspects of the role of language in communicating meaning. As

stated by Andersen (1990), a semiotic approach to the designing computer interfaces must involve the following elements:

- it should recognize that most work is collaborative, involving more than one person, and that cooperation involves conversation that could be supported by computer systems;

- it must recognize the aspects of work that do not consist of information processing;

- it should involve users actively in the design process;

- it should focus on the current practice of the users since needs for design have their roots here;

- it must view language as a social phenomenon used for communicating and coordinating work, not only as a phenomenon of the individual mind;

- it must give understanding of creative use of signs, an understanding which must be formalizable, since creative use of computer-based signs involves formalization;

- it must respect actual language usage as the basis of analysis. (p. 5)

In making statements such as these, writers are making the assertion that the meaning that individuals give to words, ideas, and symbols are a critical element in the information that is conveyed in communication and that it is this meaning that forms the basis of individual preferences for communicating with different kinds of symbols in computer systems.

Learning Style and Computers

One idea that has received the attention of a number of researchers is whether learning style has an affect on one's ability to work with computers. Mixed results have come from these various attempts to link learning style and success with computers.

One measure of learning style is that of reflection-impulsivity (Kagan, 1965; Kagan, Rosman, Day, Albert, & Phillips, 1964). In this concept, it is believed that aspects of an individual's problem solving capabilities can be determined. Basically, those people who tend to be less impulsive tend to more completely consider the nature of a problem before taking action (Van Merrienboer, 1990, p. 45). In considering the issue of how problem-solving might affect an individual's success in writing computer programs, no relationship was found between these two factors (p. 50).

Other writers have attempted to link learning style and performance in applications-oriented computer courses where individuals attempt to learn the fundamentals of word processors, spread sheets, and databases. Researchers used Gregorc's Style Delineator to group students into four groups that were based on each individual's learning style: Abstract Random, Abstract Sequential, Concrete Sequential, Concrete Random (Davidson, Savenye, & Orr, 1992, pp. 348-358). Applying classroom performance measures, mixed

results came from this study. Two groups (Concrete Sequential and Concrete Random) showed no particular difference in their performance levels. However, two groups (Abstract Random and Abstract Sequential) did show significant statistical differences in the level of their classroom performances. Abstract Random learners performed significantly lower than average and Abstract Sequential learners performed significantly better than average (p. 348).

Another theme sounded by writers related to how learning styles affected the success of students in adapting to Computer Aided Instruction (CAI). In particular, authors have examined whether or not success with different styles of CAI were affected by learning style (Cordell, 1990, p. 175). While the study suggested that teachers needed to make changes in instructional design, no significant effect from learning style was found by researchers in the educational process (p. 179).

A similar study done by Rowland and Stuessy (1988) found that learning style did have an effect on the success and effectiveness of CAI on student learners in the science laboratory. However, the model for learning style used in this research was a comparison of CAI success to whether the student was a holistic or a serialistic learner (p. 36).

Furthermore, some researchers applied the concept of learning style to the general issue of how information system structures are analyzed and designed. In this regard, it has been suggested that individuals with different cognitive styles prefer to have their information organized in varying ways. Therefore, the entire internal structure of information systems should be adaptable to the different needs of people with different learning styles (Cline, 1991, p. 3). This could be achieved by having a number of different internal operating modes built into the design of the system (p. 36). It was also advised that systems designers should consider using tests such as the Embedded Figures Test and the Meyers-Briggs Type Indicator (Coan, 1978, pp. 629-631) in order to determine what different operating modes should be constructed as part of the development process (p. 642).

One major theme in literature relating to learning style and computers related to the use of computers as tools for helping teachers assess and work with differing individual cognitive styles. One researcher demonstrated how computers could be used by teachers to help develop individuals who possess differing elements of Gardener's Multiple Intelligences (Davis, 1991). In this regard, the investigator paid particular attention to how computers could be used to help intelligences (spatial, musical,

bodily-kinesthetic, and interpersonal) that are not normally addressed by the educational process.

Sounding a similar theme, others have suggested that microcomputers can be used to aid students with different learning styles during four different phases of instructional delivery. These phases include assessment, prescription, instruction, and evaluation (Chadwick & Watson, 1986, pp. 125-131).

Researchers have even considered the effects of learning style on the level of computer anxiety that people have (Chu & Spires, 1991). Using the Jungian personality types as the cognitive style framework and the Meyers-Briggs Type Indicator as the differentiator, investigators found that personality type did have a relationship with level of computer anxiety. Individuals who were evaluated as Thinking and Intuitive scored lower on the anxiety scale than did Sensing and Feeling individuals (p. 18).

To summarize, researchers definitely have a mixed collection of results in their attempts to link learning style and computing. Several factors could account for these mixed results. First, there are possibly so many dimensions to the human-computer transaction that researchers are not completely sure of what they are actually measuring. Secondly, there are even more dimensions to learning style. Because of this, it is highly possible that researchers have obtained varying

results with the changing learning style frameworks that have been applied to the problem.

Human Factors in Computing

The general rubric of "human factors" has been used to describe a wide range of research in the computing field and in related fields. However, as used by computer software designers and the journals that are devoted to the technical aspects of having computers effectively serve human beings, this term usually means the physical and psychological factors that can affect how successfully individuals interact with computers (Marsh, 1990, pp. 16-22).

The journals that address the technical aspects of human factors in software design are typically engineering oriented. Generally, they take existing areas of computing and cognitive theory and try to see how computer programs can be implemented to address these issues.

Cognitive issues in software use are extensively addressed in these journals. However, once again, the theme primarily is the solution of particular problems in particular software systems. For example, a typical research effort focused on how sequences of different possible logical conclusions could be represented and implemented in hypertext software systems (Shum, 1991, pp. 38-40).

In summary, the vast majority of interest by researchers in the area of human factors in computing is ironically on software, the methods for specifying the internals of the software, and the computer itself. Cognitive issues are not extensively addressed, and when they are addressed, it is usually as a subsidiary concept to the primary one of methodology or programming technique.

Computers, Communication, and Culture

The realization that communication with computers is a linguistic act is a relatively new idea. However, the concept of communications and linguistics being a cultural activity that can in turn affect the use of computers in the organization has been present for quite some time.

The cultural aspects of communication have been discussed extensively by a number of writers. The importance of culture in communications is recognized by an anthropologist who stated that,

The cultural and psychological insight that is important for us to accept is that denying culture and obscuring the effects that it can have on human talents can be as destructive and potentially dangerous as denying evil. (Hall, 1976, p. 7)

This importance of culture in communications was more directly presented with the following,

Culture is man's medium; there is not one aspect of human life that is not touched and altered by culture. This means personality, how people express themselves (including shows of emotion),

the way they think, how they move, how problems are solved, how their cities are planned and laid out, how transportation systems function and are organized, as well as how economic and government systems are put together and function. (Hall, 1976, p. 16)

Despite the importance of culture in communication, not all cultures or aspects of cultures are completely understood. This is particularly true of the particular sub-culture that has been labelled as computing. However, because the artifacts of culture are not only formed by it but also change the culture that they are a part of, this lack of understanding of the computer culture can be quite problematical. Anthropologists label this process of cultural modification that is created by the tools that are manufactured by the inhabitants of the culture as extension-transference (Hall, 1976, pp. 25-40). Not understanding the computing milieu increases the risk of not understanding one of the major areas in which new cultural symbols are created and transferred.

The reality that the nature of the computing culture is quite singular has been documented a number of times. One of the earliest attempts to understand this culture was Hackers: Heroes of the Computer Revolution (Levy, 1984). In this work, some of the basic principles of the hacker ethic were outlined,

Access to computers--and anything which might teach you something about the way the world works--should be unlimited and total. Always yield to the Hands-On Imperative;

All information should be free;

Distrust authority--promote decentralization;

Hackers should be judged by their hacking, not bogus criteria such as degrees, age, race, or position;

You can create art and beauty on a computer;

Computers can change your life for the better.
(37-49)

While these concepts are central to the ethic of many who drive the computer revolution, they are clearly at odds with many of the belief systems of society as a whole. For example, the idea that information should be free conflicts with the ideas of intellectual property and copyright which have driven much of the legal processes over the years.

Also, the idea that art and beauty can be created on a computer is an anathema to many who believe that computer systems are forces for dehumanization. Further, many people who have lost jobs to computerization or who simply fear computers for their own sake would wonder at the assertion that computers can change one's life for the better.

However, despite the bizarre belief systems of the computing folk, they are having their impact on the social fabric. For example, the ethic of free information has driven the development of the Internet. Also, computing people are altering the meaning and use of symbols.

Consider that the meaning of terms such as prompt, hack,

bus, drive, and terminal have either been extended or altered. Further, words such as cyberpunk and infoglut have been added to the general vocabulary.

As a consequence, it is critical that the culture of computing be understood. Any organization that proposes to install computers will be irretrievably altered by this process. Understanding the personal and social upheaval engendered by the installation of computing technology or by the installation of new generations of computing technology will naturally help those in the organization to better manage the chaos that comes from such change. Understanding the computer culture is an important step in this process.

CHAPTER 3

METHODOLOGY

Descriptive Case Studies

This research was a descriptive case study. As such, both qualitative and quantitative methods were used to obtain information from study participants. Interview data were obtained through a series of personal interviews with each of the participants. Further data were obtained through an application of Kolb's Learning-Style Inventory. This last instrument was used as a tool for verifying learning style assessments, which were done primarily through questions asked during the interviews.

The qualitative case study has been defined as "an intensive, holistic description and analysis of a single entity" (Merriam, 1988, p. 16). In particular, case studies have the ultimate characteristic of permitting a researcher to focus on a single situation, case, or phenomenon. They are described as being "bounded" (p. 11). The issue of being bounded refers to the singularity of the situation being described.

In addition to focusing on a particular event or situation, qualitative case studies are descriptive. This

means that they can use a variety of data collection and analysis techniques to build an in-depth description of the phenomenon to be studied. Furthermore, these qualitative studies tend to report results by using narration and quotations rather than the tables, statistics, and graphs that are typical of the quantitative study.

In applying description and narration to the reporting of information that is inherent in the qualitative case study, the objective of the qualitative researcher is to "illuminate the reader's understanding of the phenomenon" in question (p. 13). As such, the importance and meaning of the information being discussed tends to be very much dependent on the context of the situation being studied. The consequence of dependency on context tends to be a limitation of the amount to which the researcher can generalize the information that has been found through the study. Therefore, in generalizing, the researcher has to be careful to insure that any extrapolations are to appropriate groups and individuals for the phenomenon being described. "What the researcher looks for are the systematic connections among the observable behaviors, speculations, causes, and treatments" among individuals for whom the analysis is suitable (Smith, quoted in Stake, 1988, p. 255).

A qualitative case study was chosen for this study because of the uniqueness of each individual. In essence,

each person, with unique needs and reasons for working with computers, formed their own bounded system which deserved to be addressed within its own context. This is the reason for using the case study approach.

In addition, qualitative methods were necessary because of the questions being asked in the research. Ultimately, this reduced to "why?" For example, why do individuals with a certain learning style like a particular user interface? As stated by Merriam, questions that are posed in this manner demand the application of qualitative methods (1988, p. 9).

Procedures

In this descriptive case study, 30 individuals were purposively selected to participate. These individuals came from different categories of users who typically make use of computer systems in different ways. These included computer professionals (systems analysts and programmers), college computer science majors, secretaries, experienced professional computer users, and non-experienced professional computer users. The individuals selected came from a number of sources including local colleges, some local businesses, and a local hospital. Five faculty from the local colleges (two historians, one computer scientist, one accountant, and one physicist), four secretaries employed at the colleges, one sociology student, one

criminal justice student, three video technicians at the colleges, one computer science student, one graduate student in professional counseling, two professional programmers from local businesses, one civil engineer, one missile launch officer, three librarians, six administrators from the colleges, and one nurse from a local hospital all participated in the study.

The preceding categories of users were chosen for a number of reasons. First, these categories represented a fairly broad spectrum of users with different needs and reasons for interacting with computer systems. Secondly, the different categories and locales were selected to insure that a wide variety of learning styles were represented in the study. Since learning style is at least partially related to the profession that one is in (Kolb, 1981, pp. 232-254), these diverse categories increased the likelihood of having all learning styles involved in the study.

In this case study, data were garnered by observation and by interviews. In both of the data gathering methods, certain common pieces of information were obtained. First, cognitive styles for computer-related problem solving were assessed for each participant in the study. Secondly, preferences for interfaces were determined. Finally, the reasons for the preferences were ascertained.

During the observation phase of the study, one of the major objectives was an initial assessment of the learning style of the study participants. To implement this assessment, each of the participants was watched while working with the user interface. There were specific behaviors that helped to manifest the dominance of a certain learning style. For example, a preference to watch others before getting involved with a new technique was interpreted as an indication of a preference for reflection as a learning style. Therefore, to get information which was as complete as possible about the individual's learning while working in the computer environment, each participant was observed using their preferred user interface. In those instances in which an individual used more than one user interface, then observation was made of the individual using each environment.

Kolb's Framework

Kolb's learning style framework was used to identify behaviors for assessing individual learning style. Kolb's framework assumes that there are four primary types of learning styles. These styles include Concrete Experience, Reflective Observation, Abstract Conceptualization, and Active Experimentation. Furthermore, these learning styles represent two primary axes of learning styles with Concrete Experience and Reflective Observation forming an axis and

with Abstract Conceptualization and Active Experimentation forming another axis (Figure 2). The two axes are thought to represent two distinct aspects of the learning process and are generally graphically represented as being orthogonal to one another (Kolb, 1985).

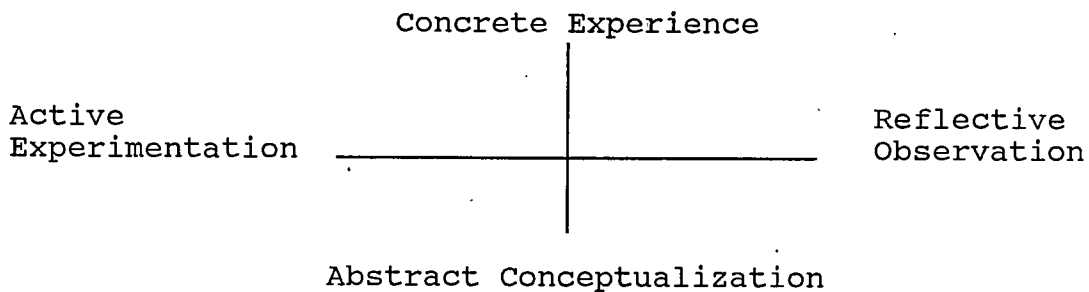


Figure 2. Kolb's Learning Styles.

Kolb's framework was suitable for this study for a number of reasons. While some writers in the field of cognitive and learning styles are skeptical about the psychometric properties of the Kolb Learning-Style Inventory, most generally accept the validity of Kolb's framework for viewing the methods that individuals use to accomplish learning (Geiger & Boyle, 1992). The validity of Kolb's framework has been supported in numerous studies (e.g., Katz, 1986; Merritt & Marshall, 1984; Wilson, 1986).

A second major reason for using Kolb's framework for understanding learning style is that it provided an effective viewpoint for different learning methods that might be employed in working with computers. Distinctive

actions in working with computers were representative of the various different learning styles in Kolb's framework. For example, certain distinctive actions were indicative of the individual's preference for Reflective Observation. This included a hesitancy to try new things immediately. Instead, the individual preferred to try something new only after having thought about it for a period of time. Another behavior that indicated a preference for Reflective Observation was a need to watch somebody do something before attempting it oneself. This tendency toward Reflective Observation was revealed through a combination of the application of observation and interviews. Questions that helped to reveal Reflective Observation included: Do you like to experiment with new software systems, or is learning more effective if you can observe someone first?

A behavior that revealed a tendency toward Concrete Experience is the obvious pleasure that an individual might take in working with the computer. Furthermore, participants were asked if they preferred to learn through lecture or through hands-on work. A preference for hands-on indicated a tendency toward Concrete Experience.

In a similar fashion, Active Experimentation was also revealed by participants' actions and their answers to questions. Individuals who like to try new features of software were demonstrating a penchant for Active

Experimentation. In the interviews, questions about experimenting with software features were used to probe for the practice of Active Experimentation.

Since manuals are difficult to read and require individuals who use them to have a fairly complete understanding of the theoretical framework of the computer, effective manual use was a clue that the individual has a tendency toward Abstract Conceptualization. Again, questions also helped to reveal the presence of this quality in participants. For example, participants were asked if they refer to manuals often. They were also asked if they like to get information through lectures because of the association of a preference for lectures with Abstract Conceptualizers.

After the observation, interviews with participants began. While the process of observation and interviewing was not fully separable, working with each participant had two distinct stages: an observation phase (watching participants at the computer) and an interview phase. Some of the ideas that have been pursued through observation were also pursued through the interview process. For example, issues in cognitive style were further pursued during the interviews.

The first question that was asked related to what user interface each individual preferred to use. The major direction of the questions related to what the user liked

and disliked about their programs and why they reacted as they did to these particular features.

To help focus the issue of like and dislike, questions relating to features of interfaces that were helpful and that were hindering were also asked. This aided the participants to focus on specific concepts relating to the user interface that they liked and that they disliked.

Participants were also asked to specify features that they felt should be added to or deleted from each type of user interface that they use. In this regard, the presence of shell environments in Command Line Interfaces were discussed. Shells, or menu systems, exhibit some characteristics of Graphical User Interfaces. Similarly, providing the user with the ability to access operating system prompts in Graphical User Interfaces was also discussed. Additionally, features that should be added to each category of user interface were pursued.

As a result of this interview and observation process, several pieces of information became available from each participant. First, cognitive styles for each user were assessed. Secondly, preferences for each interface were extracted. Finally, the reasons for these preferences were determined. The combination of these pieces of information permitted patterns of preferences and cognitive styles to be related to the reasons for these preferences. This

information provided additional insights about preferences and their causes.

As a reliability check on the accuracy of the observed cognitive style of the individuals in the study, the Kolb Learning-Style Inventory was employed. This was done with each study participant after the preference for user interface type was determined. In this way, the instrument itself did not bias or influence participants in their responses or the researcher in the observations.

To implement the reliability check, each participant's learning style, which had already been determined through an assessment based on interview information, was calculated from Kolb's instrument that had been administered at the end of the interviewing sessions. The assessment of learning style coming from Kolb's instrument was then compared, on an individual basis, with the qualitatively determined learning style that had been inserted into the field notes kept for each study participant.

Interview Questions

To understand the nature of the interface preferences of the study participants, the following interview questions were used to direct the discussion with the participants:

1. What user interface do you commonly work with?
2. What user interface do you prefer to work with?
3. What do you dislike about a particular user interface? Why?
4. What do you like about a particular user interface? Why?
5. Which features do you find particularly helpful in a particular user interface? Why?
6. Which features do you find particularly hindering in a particular user interface? Why?
7. What features would you change on the next release of the software? Why?
8. What features would you preserve on the next release of the software? Why?
9. What features would you like to see added to the software on its next release? Why?
10. Do you prefer interacting with a computer using a keyboard or by using a mouse? Why?
11. Do you find menus to be helpful or to be hindering? Why?
12. Do you find user command lines to be confusing?
13. Do you find user command lines to be intimidating?
14. Is ease of use or the ability to explore more important in a computer system? Why?
15. What advantages do you see computer systems providing for you?
16. What disadvantages do you see computer systems bringing to you?

In addition, a series of questions were asked that were designed to determine the learning style of the participants. These questions included:

1. When working with computers, do you find yourself hesitant to try new things?
2. Do you like to think about ideas before trying them?
3. Do you like to watch others work with new ideas or techniques first?
4. Do you like to work with computer systems?
5. Do you prefer lecture or hands-on work as a way of learning about computers? Why?
6. Do you use manuals? Do you like to use them? Why?

The interview questions were used with a wide range of research participants. Therefore, they were employed in differing ways with individuals with different backgrounds. For example, some participants had a high degree of computer literacy and had worked with more than one type of user interface. With these individuals, it was appropriate to get them to do a direct comparison of the relative qualities of the different user interfaces with which they had worked. On the other hand, some of the participants in this study had substantially less exposure to computer systems and consequently were unable to make a direct comparison between different types of user interfaces. In these instances, the individuals were asked to simply describe the likes and dislikes that they had with respect to the software with which they had worked.

Kolb Learning-Style Inventory

In addition to the interview questions, information was gathered with the Kolb Learning-Style Inventory (LSI). This tool was developed as a simple method for measuring the learning style preferences of an individual (Kolb, 1976, p. 30). Originally, the questionnaire consisted of nine questions. However, it was eventually expanded to consist of 12 items (Gregg, 1989, p. 441).

In the instrument, each question has four items. Participants rank items in order of preference. Each of the items for each question represent different qualities of one of the four learning styles. All four learning styles are represented in each question.

A typical example question from the inventory is as follows:

I learn best when:

- ☐ I rely on my feelings.
 - ☐ I rely on my observations.
 - ☐ I rely on my ideas.
 - ☐ I can try things out for myself.
- (Kolb, 1985, p. 2)

In this example, the first item refers to Concrete Experience, the second refers to Reflective Observation, the third refers to Abstract Conceptualization, and the fourth refers to Active Experimentation. Because the order of the items referring to a quality is consistent in each question, a score for each of the four qualities can be developed by summing the preference scores in the

appropriate position for each question. Consequently, the inventory produces scores for each of the four qualities.

The instrument has a number of qualities to recommend it for use in this study. Foremost is the fact that it is relatively short. Since it consists of just a few questions, it is easy and quick to administer.

Much discussion has been done on the psychometric qualities of Kolb's Learning-Style Inventory. However, it does appear to have acceptable levels of internal consistency. In one research study, a coefficient alpha of .86 was cited (Ruble & Stout, 1990). However, there has been some discussion in the literature of some reliability problems from individuals taking the inventory, reacting to the lists of words that make it up, and having their individual scores vary significantly from one time to another (Wilson, 1986, p. 5). However, Kolb explains that this variation is due to the perception of an individual's personal learning style varying from time to time (1986, pp. 5-6). Nevertheless, this difficulty did not threaten this study because the participant's learning style was determined primarily by observation. While reliability of Kolb's Learning-Style Inventory has occasionally been questioned, the framework has been generally accepted.

The other major psychometric quality to be considered with Kolb's Learning-Style Inventory (LSI) is validity. In this area, the criticism of the LSI is generally more

severe. For example, some researchers have examined his inventory and have concluded that it does not really measure any underlying learning construct (Certo & Lamb, 1979). On the other side, as a criterion-related validation of the Learning-Style Inventory, Kolb cites a study in which the results of the instrument were compared to another instrument that was developed to evaluate learning styles in the work place (Gypen, 1980). Kolb (1985) used these results as "empirical support for the bipolar nature of the experiential learning model" (p. 76).

Despite the criticisms that have been levelled against Kolb's Learning-Style Inventory, the flaws do not appear to be fatal. This is especially true when one considers the role in which Kolb intended the Learning-Style Inventory to be used. His proposal was that it be used as an "analytic heuristic" for studying learning (Kolb, 1985, pg. 76). In this, Kolb appears to be saying that it should not be regarded as an absolute measure of the exact amount of the various qualities that individuals possess. In this study, the instrument's use as a verifying tool fit Kolb's idea of an "analytic heuristic."

Methods of Data Recording

Two primary methods of data recording were employed. During the observation process, field notes were used to record the computer interactions of the various

participants. Since the observations also involved questions, these too were recorded using field notes.

During the interview process, both field notes and audio recording were used when the participant granted permission for the session to be recorded. Both data collection techniques were used in order to assure the accurate collection of multiple types of data. The checking process inherent in multiple sources can help investigators increase the clarity of the final conclusions and can help avoid jumping to erroneous conclusions (Glaser & Strauss, 1967).

In applying these two methods of information gathering, 1 to 2 hours were spent with each person in the study. This time, depending on how the data gathering process was progressing, was sometimes divided into more than one session. Whether or not sessions were divided depended on the work schedule of each of the participants. If a large block of time was available, then data gathering continued until complete.

CHAPTER 4

FINDINGS

Data were gathered from several sources. First, a learning style assessment was made through a number of open-ended questions that were asked during interviews with the participants. In this regard, Kolb's Learning Cycle was used as the basic framework for a determination of the learning style of the participants. Secondly, study participants were observed and were interviewed to determine their reactions to the software packages that they use. Finally, each individual took Kolb's Learning-Style Inventory to provide a validation of the learning style that was initially assessed qualitatively.

Learning Style Assessment

The determination of learning style was based on the participant's answers to a series of questions, including:

- a. When working with computers, do you find yourself hesitant to try new things?
- b. Do you like to think about ideas before trying them?
- c. Do you like to watch others work with new ideas or techniques first?
- d. Do you like to work with computer systems?

- e. Do you prefer lecture or hands-on work as a way of learning about computers? Why?
- f. Do you use manuals? Do you like to use them? Why?

Each of these questions was designed to get at a different aspect of learning style. For example, the question about trying new things was designed to help determine the participant's propensity for Kolb's concept of Active Experimentation. The second and third questions were meant to determine a participant's penchant for Reflective Observation. The question pertaining to liking to work with computer systems was oriented toward a determination of a preference for both Active Experimentation and Concrete Experience. The last two questions, referring to lecture as compared to hands-on and referring to manual use, were oriented toward determining if a person tended more toward Abstract Conceptualization or Concrete Experience.

Assessing the learning style of the participants qualitatively turned out to be remarkably similar to the written results. The conclusions drawn from these open-ended questions yielded results that later compared, quite favorably, to the values for learning style as determined by Kolb's Learning-Style Inventory. For example, of the 30 participants in the study, the qualitatively determined learning style agreed with the value coming from Kolb's inventory 22 out of 30 times. Thus, this means that

qualitative assessment and Kolb's inventory agreed 73% of the time. In 6 of the remaining 8 instances, at least one of the two axes of learning style was predicted accurately. In those cases, the difficult item to assess was the preference for Abstract Conceptualization as compared to a preference for Concrete Experience.

Given the preceding, it can be seen that there were only two egregious errors that were made in the determination of learning style. This simply came from the fact that the individuals in question did not have a particularly strong learning style in any way. Their discussions of activities and predilections that would indicate a preference for Reflective Observation over Active Experimentation indicated preferences that, for assessment purposes, tended to cancel each other. For example, an individual might express both a strong desire for getting his hands dirty in physical experimentation and a very real need to be reflective about new ideas. The same tendency for some individuals to have predilections that cancelled each other was also true for Abstract Conceptualization and Concrete Experience. In these cases, an assessment was made about the learning style which the instrument did not validate.

The determination of learning style was done partially through observation and partially through asking the interview questions. In this process, the observation

generally took place first. This would occur at the participant's work place and would involve some computer-oriented task that the individual had to complete. As a result of the observation, a general idea of the work patterns and preferences of the participant was obtained. This was used to guide the asking of the research questions for both learning style and software interface. In this process, the questions pertaining to learning style were posed first. Hence, formal questioning of participants was done after they had been observed at work on the computer.

To summarize the process of assessment of learning style for the survey participants, two learning style determinations were made for each individual. A Reflective Observation-Active Experimentation assessment was made and a Concrete Experience-Abstract Conceptualization assessment was made. This was done because Reflective Observation and Active Experimentation are in Kolb's structure two parts of the same axis, and, as such, they represent two ends of a spectrum of possible learning style alternatives. Similarly, Abstract Conceptualization and Concrete Experience represent another such axis. Therefore, of the 60 assessments that were made of learning style for the 30 participants, 50 were done in agreement with Kolb's instrument. Thus, 83% of the qualitative assessments of learning style agreed with the value determined quantitatively.

Table 1. Study participants in each learning style category.

Learning Style	Number of Participants
Converger	8
Diverger	4
Assimilator	8
Accommodator	10

Interview and Observation Results

The majority of the interaction with study participants involved interviews and observations. The interviews took place on a one-on-one basis and generally lasted from 45 minutes to an hour and 30 minutes. In some instances, follow-up interviews were done when particularly intriguing ideas were put forth by a study participant.

In contrast, observations were handled in a couple of different ways. Some participants were observed in formally arranged sessions. Others were observed more informally without specific times being set aside for the observation process.

Learning Strategies

The interview and observation process yielded some useful results with regard to learning strategies that are employed in learning computer related subjects. In many

ways, there seemed to be no overall pattern to the approaches used by learners with different learning strategies for mastering computer subjects. However, small patterns did arise periodically that were affected by learning style.

Strategies Unique to Learning Computers. It sometimes has been heard as a issue of common faith that learning computers is such a unique task that individuals have to employ different methods in order to accomplish a learning task with computers (Weinberg, 1971, pp. 3-4). The comments of the participants did not support this assumption. In fact, with participants of all learning styles, a constancy of learning strategy was a commonly sounded theme.

For example, one individual, who would be described as an Accommodator (a person with a preference for Active Experimentation and Concrete Experience), felt that "my approach to learning other things is basically the same as with computers. Sometimes, I have to read things aloud. Basically, I have to be actively involved."

Others sounded the recurring idea. A nurse who has responsibility for working with computers was determined to be an Assimilator (someone with a preference for Reflective Observation and for Abstract Conceptualization). She described her approach to the learning process: "I find

myself volunteering to do some project that will force me to learn something new. . . . My learning process for computers and for non-computer subjects is similar."

This theme of using one's preferred learning style to learn about computers just as with other learning tasks was also repeated by Divergers (individuals who are prone to Reflective Observation and Concrete Experimentation) and by Converggers (those who prefer Active Experimentation and Abstract Conceptualization). Consequently, the learning strategies employed in working with computers are not that different from working with other subjects.

However, this theme did not apply to all participants, and there was some minor dissension from the previous assertion. A college professor declared, "Here, it depends on the material that I am trying to master. With computers, I definitely prefer to learn hands-on. The theory is something that I can learn from the textbook." In further explanation, he also stated, "In dealing with computers, it is difficult to translate from book to reality. In contrast, in physics, you can go a lot faster with lecture." The difficulty with the response from this particular participant comes from the fact that he may be mixing his own preferences for learning a computer-related subject with his preference for a teaching style in the subject that he instructs, physics.

However, the viewpoint of computers as a non-theoretical subject was a somewhat common theme among many of the participants. One participant stated, "Lecture is definitely better for theoretical subjects. I don't want to learn computers through a lecture." This view of computing as a non-theoretical subject is probably due to the nature of the contact that the majority of learners have with computer-oriented subjects. In most instances, this contact is non-theoretical, does not emphasize programming, and causes the students to have more experience with common applications software (Ivey, 1992, pp. 120-121).

Strategies for Learning Different Computer Systems.

The participants used different learning strategies for approaching different kinds of computer systems. In many ways, this seemed to be related to the different kinds of resources that were available on different kinds of computers. One technically-inclined video producer summarized the use of different learning strategies for different computer systems as follows:

This varies with the system. On a stand alone PC, I use the documentation. I'll find the function in the documentation or something similar to what I want. Then I'll check the high points of the discussion and try to implement the technique on the computer. At the minicomputer, I'll call the wonks and ask for advice. What are the steps that are involved? Then, I'll check with the administration to see if it would be too much of a hassle to get this particular function

implemented. At the paint program on the Amiga, it is more important to fool with it at the Graphical User Interface. Here, I have a tendency not to read a manual.

The conclusions expressed by the video manager seem to be the result of the general state of affairs in modern computing. For example, many computers and software systems that are available for personal computer systems are very popular. Many people will be expert on them, and many books and learning aids will be very commonly available. For these popular computers and software systems, users will have many different alternatives when seeking help and information about software and computer functions. In contrast, less popular computer systems, such as business-oriented mini-computers, will not have so many self-help manuals available with which people can work. Furthermore, the only manuals that may be available may be the ones that have been put together by the manufacturer of the hardware or software. This, in turn, may have a deleterious effect on installers and users of computer systems because of the almost universal contempt that most computer users have for computer manuals (Weinberg, 1971, p. 262).

Manuals. Manuals are critical resources for users of computer systems, and they represent a critical element, for either the good or the bad, in almost every user's learning strategy with respect to computer systems.

Regardless of learning style, there is massive distaste for computer manuals. This is true even among those who use them extensively. Of the 30 participants, 25 made uncomplimentary remarks about manuals. Typical comments include:

I use manuals. I don't like using manuals. Usually, when I pick up a manual, it is because I don't know something. I don't like not knowing.

Yes, I use them for word processing. I don't like them. I think that I am following them, but I don't get the result. Maybe I'm in the wrong place. They're too wordy. Pocket summaries are better.

I like "how-to" manuals and instructions. On the other hand, I dislike computer manuals. The language that they use tends to be too technical. They also tend to be missing lots of information. All this means that I usually can't follow them for problem solving. Besides, manuals never seem to address the problem that I am facing. The problems that they work with always seem to be different.

Respondents of all learning styles dislike manuals. Each individual seems to have their own reasons for not liking manuals. However, this distaste does not seem to extend to the books that have been written by individuals not associated with the manufacturer as "how-to's" about a particular computer or software system. A college administrator declared, "I hate manuals! They're confusing! What planet are those people from? If I have to learn about software, I go and find one of those how-to manuals at the bookstore." A college instructor declared, "I like how-to manuals and instructions. On the other

hand, I dislike computer manuals. The language that they use tends to be too technical."

Technical People and Manuals. All technically-oriented participants expressed reluctance to use manuals. Some expressed the idea that they felt that they should know all about a particular subject without having to consult the manual. A technical librarian with computer responsibilities who is an Accommodator said, "I use manuals, but only when I'm stuck. I like to think that I am smart enough to use the system without help." Another participant, a computer science student (who is an Assimilator), admitted, "I only use manuals as a last resort. I don't like them. I feel like I should be smart enough to figure it out myself. Use of a manual is wimping-out." Another student, who works as a lab tutor, agreed, "I don't like to use manuals because it shows weakness. I don't like to show weakness in front of underclassmen!" In general, it seemed as if many computer people would make use of a manual, but they usually would not make it their first recourse when learning about computers or even when trying to solve a problem.

Improvement of Manuals and Documentation. Despite the general distaste that participants had for manuals, a number firmly asserted that quality documentation was essential to effective software. A librarian believed that

"any software should have all steps present in its documentation. The screens should be designed so that you can look forward and have the computer tell you what your next step should be."

There were mixed reviews with respect to online manuals and help systems. One participant, a Converger who works in an engineering office, explained that "they are vague and generally inadequate. I don't use them much." An Accommodator disagreed, "The online help feature that is available all the time in any application is really helpful. The fact that DOS doesn't have this is a problem."

While some disliked all computer manuals, others desired more complete on-line help. A Converger reacted favorably to online documentation when he seemed to ask for more online help by saying, "I find it restricting to have to leave the computer to find a manual so that I can look something up about what I'm doing. It is more convenient to look something up through online help." One participant, a nurse with a preference for abstraction and reflection [Assimilator] asked for more extensive online help manuals: "They still expect people to go to the manual. As a result, people stop using help. They might put the complete operating manual online."

Gender Differences in Learning Strategy with Computer Systems. One issue that is often raised with respect to learning strategy is use of human resources to help answer questions. In this regard, the willingness of individuals to seek assistance with technical problems can be affected by a hesitancy engendered by a need to feel independent of others. For female computer users, this seems to be a larger issue than for males working with computers. Half the female participants raised this issue. In contrast, only one male discussed the idea of independence. For example, a female school administrator felt "the problem with computers comes from the feeling that I have that I have poor problem-solving abilities when it comes to computers. This has a tendency to make me more quickly dependent on others than I would like." A female librarian explained, "However, I feel like manuals help to make me a more independent person." A female graduate student cited the help feature of a program because "I can independently take care of what I need."

The gender difference in the use of resources is also confirmed by observation. In general, female users of computer systems tend to be more reluctant to ask for help from technical experts. This seems to be all the more true of those situations in which the female computer user has to deal with an established information systems or data processing department. A female administrator asserted, "I

don't always like to go to the computer people for help. It takes too much time, and they tell you what they want you to know but not what you need." In this environment, the issue of independence appears to be all the more critical for women who use computers.

Problem Solving Approaches. Participants in the study were observed to see how they approached problem solving in working with computer software. This was done in the participant's normal working environment. Sometimes, the observation was done formally as part of the interview process. At other times, the observation was done informally.

Solving problems with computers is more closely related to learning strategy than it is to learning style. Learning strategy is the tactics and resources that an individual applies in a particular learning situation. When encountering problems with programs, it appeared to be a fairly natural extension of a person's methods for resource acquisition, utilization, and management to use the same approaches for managing learning resources to handling difficulties in learning about computers.

This theme of learning strategies remaining constant was echoed by a number of participants. An engineer reiterated, "I like to look at models of the subject and to

talk to others who have already done it. I use the same approach in working with computers." A nurse explained:

First, I would obtain information on the subject to help me get a feel of what the people are saying on the subject. With computers, I might buy some magazines such as Computer Shopper to help with this process. I might buy any magazine that applies specifically to my problem. These resources will help me to get specific ideas of things to keep my eye on. Once I get a feel for what I want, then I will make a decision. Then, I'll look in a book for things that I want to know. These will usually be related to the accomplishment of a specific task. I tend to use the manual a lot. I use telephone help-lines a lot. After researching and reading the manual, I am able to effectively verbalize my questions over the help line. Once I have gone through this initial learning process, then I will try to teach someone else how to use the program in question. I try to tell people who could benefit from the knowledge of this software. This helps me reinforce my own knowledge of the program. In the cases where things are more complicated, I will go to computer books, even if I legitimately own the software myself, before calling the people in Information Systems. A lot of times, I find myself volunteering to do some project that will force me to learn some new software package that I have been putting off learning. My learning process for computers and for non-computer subjects is similar.

Consequently, if a person was normally inclined to make extensive use of human resources, then this would continue when the individual dealt with computer difficulties. Also, if an individual had an aversion to printed matter, then all the inducements in the world (including well designed manuals and online help) could not persuade the person to open a manual.

However, a word of caution should be offered at this point. As mentioned previously, there are some gender differences in the level of willingness of computer users to seek assistance from technical individuals. The exact cause of this reluctance has yet to be determined.

Based on the observations of the participants in the study, it is possible to identify five potential basic categories of problem solvers in the computer field. These categories are Resource Managers, Mechanics, Fiddlers, Confronters, and the Intimidated (Table 2).

Table 2. Study participants in each category of computer use and problem solving.

Category	Types of Computer Users				Totals
	Conver.	Diver.	Assim.	Accom.	
Mechanics	2	0	0	1	3
Confronters	2	0	0	0	2
Res. Managers	1	2	2	3	8
Intimidated	2	1	6	4	13
Fiddlers	1	1	0	2	4
Totals	8	4	8	10	30

Resource Manager is a broad-brush descriptor for a fairly heterogeneous group of computer users. These individuals almost immediately revert to their normal

learning strategy sequence of resource usage to get a problem solved. This learning strategy might involve using help from sources such as friends, technical associates, manuals, and magazines. It might include reflection about how solving this problem will be of help to getting the job done. Resource Managers include individuals of all learning styles and learning strategies. An engineer, who might be described as the prototypical Resource Manager, explained, "First, I research it. Then, I do a lot of networking. If somebody I know already has experience with this, all the better. That means I don't have to reinvent the wheel!" A school administrator fell into the ranks of Resource Managers when she declared, "Usually, I'll go ask somebody. Getting them to show me is my preference. The second thing that I'll do is go get a book and read about it. I'm not a good person to simply go and play with things!"

One primary characteristic of the Resource Manager is that their interest in computers is tied primarily to their job and to the accomplishment of tasks within their jobs. This means that they are usually not willing to mess with computer problems when they occur. Consequently, they immediately use their usual learning strategy approaches to solving the difficulty that they are facing. An engineer, who is a strong Resource Manager, remarked that "the primary issue is getting the job done!" Another Resource

Manager explained, "Because of time invested, I use computers when I have a goal in mind or a task to do. I get the help necessary to accomplish this objective!"

Mechanics are closely related to Resource Managers. They too are prone to immediately revert to standard problem solving procedures in dealing with software difficulties. However, there is a big difference between the Mechanic and the Resource Manager. Mechanics generally have more experience with computer systems and software than Resource Managers do. Therefore, when confronted by a problem, the Mechanic will apply a standard set of solutions, a bag of tricks, to the solution of the problem. These tricks come from the knowledge, which has been gained through experience or training, that the Mechanic has about the software.

If the standard tricks work, then the problem is solved. However, if they do not, then the regular form of resource management that is dictated by the preferred learning strategy of the individual will be applied immediately. A fairly typical approach for Mechanics when dealing with an intractable software problem is to use their regular resources such as manuals or technical experts and then to simply re-install the balky software. A computer science student who works as a lab assistant in a computer lab of a local college grumbled, "I tried the usual things. They didn't work. I even looked in the

manual. It didn't make sense. The lab director wasn't around. So, I got disgusted and re-installed the program. It still didn't work!" A local computer professional explained that "experience has taught me some fairly standard problem solving steps, which I will use first. Then, if an expert is available, then I will try him. Failing that, I'll go to the manual, and then to the customer support line."

Fiddlers are very much different from Resource Managers and Mechanics. Fiddlers are usually, though not exclusively, individuals who have fairly extensive backgrounds with computers. Furthermore, they generally like computers, and they like exploring the different things that computers can do for them. Often, their interest in computers is not strictly job-related. Therefore, when they are encountered with a program that does not work as it should, they usually don't mind tinkering with it to get it to work right. Dealing with problem software is generally not problematical for these folks. In fact, software troubleshooting is considered to be a gratifying intellectual challenge. Experimenting with a program to "tune" its performance is a fairly common pastime for these individuals. A computer science instructor commented, "I prefer to jump in and fiddle and see what happens. It's fun to see what I can get the program to do!" A technical librarian complained, "They

eat holes in my time and wallet. I end up spending too many dollars and hours on them. I was up until midnight last night getting my communications program to work!"

Computer professionals and dedicated computer amateurs such as hackers appear to be dominated by individuals in this category. This simply comes from the fact that Fiddlers genuinely like to use the machine outside the task for which it is currently needed.

As with Fiddlers, Confronters also like to work with computers outside the task for which they are immediately needed. Also, they will explore new uses and techniques for working with the computer. However, they do not play with computers for the sheer pleasure of it as Fiddlers do. To a much larger extent, Confronters tend to view the computer as a tool.

The true nature of the Confronter is revealed when a problem is encountered on the computer system. Basically, when this happens, the Confronter will not let go of the problem. Individuals of this sort will continue to batter against the wall represented by the difficulty until it gives way. In this regard, the Confronter will try every possible solution to the problem that can be conceived in an attempt to make it give way. When every possible solution has been attempted and has failed, Confronters will go through the list again, and again, and again. During an observation, a Confronter was heard to growl at a

computer, "I don't care what you do. You are not going to beat me!" Further, while talking to a computer, another Confronter exclaimed, "See, I told you that you weren't going to win!"

It is only after a high level of frustration has built up that the Confronter will begin an even minimal level of resource management. Usually, this resource management will involve a minimalist consultation of the manuals. It is only under the most dire and extreme circumstances such as an imminent deadline for which the Confronter is responsible that the Confronter will make the ultimate concession and actually ask somebody for help.

For the Confronter, the process of problem solving very definitely seems to be one of the individual against the computer. This is also a trait that is useful for some technical professionals.

The Intimidated are individuals who seem to have no problem solving strategy at all. Surrender seems to be their motto. In this regard, a very common phrase uttered by a member of the Intimidated is, "See, I knew it wouldn't work." When confronted by a difficulty, the reaction of a member of this group is most usually to give up immediately without even attempting to deal with the problem.

The Intimidated usually come to work with computers under extreme protest. Many reasons exist for this hesitancy. Sometimes it is sheer inertia or the lack of

willingness to try anything new or different that accounts for this reluctance. Sometimes it is a fear of technology that accounts for this attitude. Oftentimes this outlook has been created by previous unsatisfactory encounters with technical professionals who, in their obsession for control, have deliberately attempted to frighten an individual about using computer systems. All members of the Intimidated group mentioned a previous bad encounter with computer people. One member of this group was "afraid that I was going to break the computer. The programmer at my last job kept telling that was what I was going to do." Frequently, this view of computers is the consequence of bad previous encounters with computer systems. "No matter what I did, the program broke. I got to be afraid to do anything with computers," one individual explained.

Teaching Strategies

Many instructors have long accepted the idea that knowledge needs to be related to the context of old knowledge. In this way, learners are better able to master new information by associating it with old (Ausubel, 1986, p. 216). However, participants indicated that this may not be universally useful in the teaching of computer-related subjects.

Give Me Only What I Need to Get the Job Done. Six participants described their impatience with technical help

and educators who seemed obsessed with teaching them things that were not directly applicable to understanding and working with the immediate learning task. Partially, this focus on learning only what was necessary for the immediate task was related to the fact that many of the participants limited their interaction with computer systems to those things that were necessary to accomplish their job. A school administrator explained, "My tendency is to look at computers and this program in terms of the primary task for which I use it." In a similar vein, an Accommodator explained what she wanted from computer courses: "I enjoy working with computers. I believe that they are fun. However, I don't want to know what is inside of them!" However, one participant with a preference for knowing only what was necessary to accomplish the task did make a concession: "Sometimes I'll even try to troubleshoot, but that troubleshooting is limited by my only wanting to know enough about the program to use it."

The Big Picture. While some only wanted to learn the basics of the subject, eight participants expressed sentiments about knowing the background of a subject. For these individuals, the presence of contextual and background information was viewed as entirely essential to the learning process. One participant exclaimed, "When I am getting training, I'll frequently get the experts to

write step-by-step instructions for me on how to use the program. However, for me to learn how to use a program, I need to know the big picture!"

In two instances, the need for context in the learning process for computers was expressed in terms of models. In this regard, participants would state the need for some general contextual model of what the process was and for what it was supposed to do. A college professor explained his need for context: "I will look at a manual to see what it does, to get models of how it is supposed to operate. If it is on the Mac or on Windows, then I'll start to play." A technical librarian who is a Converger echoed these ideas: "In general, I'll see a model and fill in the details later."

The process of putting information in a context got mixed reviews by participants. While some needed things put in context, a number of others felt that it was unnecessary. An even larger number opposed that view. As might be expected from past learning style research with the Kolb instrument, those who felt the need to put things in a larger informational context were dominated by Convergents and Assimilators, both groups which have a penchant for Abstract Conceptualization (Kolb, 1981, pp. 232-254).

Preferences for Different Types of Software

Much of the observation and discussion with participants was oriented toward determining why individuals liked and worked effectively with different kinds of software. While some of the reactions that people have to software can be linked to learning style as Kolb defines it, a substantial portion of it cannot.

Software Used by Participants. There was a wide range of software that was employed by study participants. This reflected the varied backgrounds, education, jobs, and levels of computer sophistication of the respondents. However, some elements of commonality were discernable. First, Word Perfect was by far the most widely used program by participants. This reflects the high level of popularity enjoyed by Word Perfect in many segments of the computer-using public. Secondly, the nature of software that individuals used was commonly dictated by work place necessity. Typically, individuals in this situation state that they like a particular program because it is that with which they are familiar. A typical explanation was provided by a school administrator: "I like Word Perfect because that is what I am familiar with." A nurse gave a similar explanation: "I generally try to use the level of program that is necessary for the job. Generally, I like whatever will help me accomplish my goal."

Software Preferences. There was little relationship between learning style and software preferences. Of the 30 participants, 28 seemed interested in computers only as a tool for helping them in their work. This was reflected in their responses to questions about the relative importance of ease of use as compared to flexibility of software and the ease of exploration. With only one exception, participants opted for ease of use because it would help them accomplish their job more effectively. Software flexibility and the ability to accomplish more things was definitely less important than the ability to meet deadlines. A secretary stated, "Ease of use is much more important. I'm here to produce a product. I am task-oriented in the use of a computer." A college administrator echoed these feelings: "The ease of use is much more important. Because of time invested, I use computers when I have a goal in mind or a task to do. I dread the time that it takes to learn a new task."

Software Analytics. One pattern that did develop relating to learning style came from the level of analysis that individuals did in understanding what makes for good software and for bad software. While all participants were basically satisfied with their software, especially if it did their task for them, and while others had specific functions that they would like to see added to a specific

program (spell check or voice recognition, for example), eight individuals reflected upon what they thought would make more effective computer programs. These individuals were, with three exceptions, Convergers (Abstract Conceptualizers and Active Experimenters). Their suggestions for more effective software design usually fell into two categories. These were a desire for consistency of software design and simplifying software structure.

Consistency of software design has long been an issue in the software development field (Weinberg, 1971, p. 218). This has led to extensive work on subjects such as Common User Access to insure that programs will do similar functions in a similar way and that programs that function under a common operating environment such as Windows will basically look and feel the same way (Mandel, 1994, p. 98). This was a common plea among study participants. A television studio technician commented, "I like features that are uniform and processes that are consistent so that the same function key does the same thing. This helps in the learning of the software." A similar theme was sounded by a college instructor who complimented the designers of Windows by saying, "Further, the general program operational patterns are very consistent. Most programs operate in the same basic way." An accountant made the following request:

Why don't all programs have some similar exit procedure? I really get leery of just clicking the machine off. This could hurt the software. Although it hasn't yet. Most software is OK for providing an exit. However, some software isn't OK. This exit process should be universal and should be easy to do.

Simplifying software structure was a theme that was mentioned by a number of Convergents. Basically, these individuals were concerned that software was becoming much too complicated and much too difficult to use. They suggested that designers might make it easier for users if they made their programs so that they would do fewer things much more easily--instead of unnecessarily complicating things with more functions that they would never use. A video technician commented, "Too many steps are involved. This may be very logical. The excess steps are still annoying, especially when dedicated machines are better." Further, a technical librarian mentioned, "Keep the fact that when you check on a word, you get four or fewer options. . . . This makes it easier to use. This is better than MS-Word where you get 16 options, many of which are not clear."

While predominantly mentioned by Convergents, individuals with other learning styles also mentioned the issue of simplicity. An Accommodator put an interesting twist on the issue of simplicity by asking for software that has different levels of simplicity and ability. This librarian explained, "There really should be two versions

of software. The first would be an easy one to get comfortable with. The other would be a powerful one with the bells and the whistles." This was echoed by an English instructor who said, "Menus are nice for learning. However, they are a little annoying after the software is learned." A Diverger, stated, "It's too complicated. I hate several steps to get into programs."

Software Features to Be Changed. In addition to considering specific additional features that they felt should be added to software to enhance program operations and in addition to considering functional changes that might alter the basic structure of software by enhancing consistency and simplicity of programs, some participants considered detailed technical issues that might affect the operational efficiency of computer programs. These detailed technical issues often revolved around the amount of memory in the computer, the type of microprocessor, and the amount of hard disk space.

While this kind of discussion was not limited to technical professionals, it tended to be dominated by Assimilators and Convergors. This supports the shared common characteristic of Assimilators and Convergors of a preference for abstraction in their learning processes.

There were a number of participants who used this approach in talking about revisions in software. In

talking about MS-DOS, an Assimilator was bothered because "it's the backward compatibility that makes it clumsy. It can't work with upper memory very well. Better memory management would help." A Converger took a similar analytic approach to discussing technical operating characteristics by saying, "One nice feature for Windows applications would be the ability to do DOS commands while still in a Windows app [application]. This would include the ability to create a directory, the ability to erase files, etc., all while still in an application." A Converger continued the discussion of technical operations by asserting, "I need a better screen and a bigger hard drive. My computer won't support video fonts very well. Also, I need a RIF viewer [which is a common MS-DOS video data file format]."

Command Line Interfaces. Command Line Interfaces, as represented by MS-DOS, Unix, and CP/M, have been until recently a fairly common method for interacting with computer systems. Many computer systems still utilize the Command Line approach which requires the user knowing an extensive command syntax which had to be typed into the computer accurately in order to get results.

While four participants expressed a preference for command lines as a way of accessing computers, the vast majority of individuals across all learning styles seemed

to have disdain for the Command Line Interface as an approach to interacting with computer systems. Eight participants expressed dislike for command lines. A typical comment about command lines was offered by a head librarian, a Diverger, "I find them confusing. I also find them intimidating. When I see one, I holler."

Another fairly typical comment came from a Converger:

"They are confusing and intimidating. Command lines in Unix blew me out of the water. This was even worse than DOS. At least DOS has some English in it!"

One fairly common theme among those who used computer systems extensively and who did not care for Command Line Interfaces is that the complicated nature of Command Lines had a tendency to distract the user from the thing for which they actually wanted to use the computer. Instead of being creative, one had to spend time and energy being concerned with the specific command steps that were necessary to accomplish the task. One long-time computer user commented, "With DOS, I feel like I am a slave to the machine. This comes from only being able to run one program at a time."

Even many of those who had used computers for a long time did not know what a Command Line Interface was. Only the highly technical participants were familiar with this term. In fact, with programs such as Windows and with various menuing systems, many people have been totally

insulated from the need to interact with a computer at the command level. In this regard, it was fairly typical to hear comments such as "I never have experienced them."

On the other hand, there was a minority of four individuals who not only liked but also expected to be able to work with the computer system at the Command Line level. These individuals were dominated by Convergents. Some of the comments that Convergents offered in this area include:

They are OK. I don't find them intimidating.

I find them to be very straight forward. If you know what the command is, then navigating around DOS and Unix directories is quite easy.

I don't find them confusing or intimidating. In fact, I am suspicious of a computing operation that doesn't allow me under the hood. It is as if it doesn't like me, or it has a lack of trust, or it fears a lack of control.

I don't find them to be confusing. I am not intimidated by them. I will use a menu instead of using a command line. I will do command lines if they are simple, for quick operations where it is simple and fast to do. Sometimes, I have to remove menus because of computer memory limitations.

I like GUI's [Graphical User Interfaces--such as Windows], but I also like the CLI's [Command Line Interfaces] because of the flexibility, power, and control. GUI is more confining. I use CLI more often. I started out in Windows. Almost immediately, I found the door to the guts. GUI consumes processor time, and frankly GUI software isn't always better. I am still more productive in a CLI.

All of this should not be interpreted to mean that Convergents in general were not intimidated by Command Line Interfaces. In fact, some of the individuals who were most

turned-off by this aspect of computing were Convergers.

One such computer professional commented, "You couldn't say confusing. Actually, cryptic is more accurate. Sometimes, the syntax gets in the way of what I want to accomplish. I have to concentrate on the how instead of the what."

However, those individuals who are more likely to be accepting of Command Line Interfaces will have a higher likelihood of being Convergers.

Graphical User Interfaces. With programs that are classified as Graphical User Interfaces, such as Windows and MacIntosh System 7, there are a wide variety of reactions to this kind of software. Many strongly like and many strongly dislike it. Because the individuals who like this type of software interface come equally from all different types of learning styles, no relationship was found between learning style and a like or a dislike for this method of interacting with computer systems. For example, an Assimilator declared, "I dislike Windows. I like to be in control. Windows doesn't permit that. I hate those silly icons on the screen. Some other jerk's image doesn't always connect with program meaning!" On the other hand, another Assimilator stated, "I prefer Windows. I am familiar with it. It seems faster."

There were some fairly consistent elements of praise for Graphical User Interfaces among those who liked this

kind of software. For example, many praised the consistent way in which programs worked. Others lauded the simplicity of operations and the quick way in which they were able to get the program to accomplish jobs for them. An engineer who is an Accommodator stated, "I also like Windows. It seems to be much quicker for many tasks. Instead of having to type information on the keyboard, I can get similar work done with a few clicks of the mouse." Others liked the fact that Graphical User Interfaces are visual and as a consequence cut down on the need for memorization. As explained by a counselor who is a Diverger, "Windows is more visual. You don't have to remember command sequences. With DOS and Word Perfect you have to remember command sequences."

Another theme related to the general visual nature of Graphical User Interfaces. This is something that a number of participants felt was a big advantage for programs such as Windows and was something that distinctly appealed to them. An instructor at a vocational-technical center explained, "I like the graphics ability of Windows. . . . The colorful nature really appeals to me."

Another theme that was sounded by those who like Graphical User Interfaces was the general ease of use of these kinds of software. In this vein, individuals usually mentioned that Graphical User Interfaces could do more for them. This was summarized by an Accommodator, "I use both

Windows and DOS. I prefer Windows because it is a lot easier to get around in and to things. It is more efficient and can do more things than DOS."

Like those who like Graphical User Interfaces, those who dislike Graphical User Interfaces could not be traced to any particular learning style. However, despite being distributed across different learning styles, there was some consistency in the themes sounded by those who profess a dislike for the graphical style of computer interfacing. One such reason for dislike resided in the area of control. Many individuals felt that the Graphical User Interface represented something that tended to insulate and isolate them from a fuller interaction with the computer system. A Converger complained, "On the GUI, I dislike the lack of control." An Accommodator explained things this way, "It is not flexible. You always have to accept the premade functions that are part of the program. Frankly, I find Windows to be more than a little insulting."

There were variations on the theme of control. One such variation resided in the area of configurability. Some participants felt that the lack of control meant that they could not set up the software to effectively run in a way that was most convenient for them. "It would be nice if Windows was more organizable. For example, I want to be able to put groups in groups," explained a Converger.

Groups are collections of icons (program symbols) which have some similar purpose. Word processing is an example.

One major issue for different users of Graphical User Interfaces related to the idea of general performance. Several issues affect the performance of computer programs. These issues include how fast the primary program runs, how fast any secondary programs run, how much memory the program consumes during operation, and how much disk space the program uses when located on hard disk. Graphical User Interfaces were criticized for each of these issues.

How fast a computer program runs is one of the most obvious manifestations of software quality. Unfortunately, because program speed is dependent on hardware quality, this is not something over which the software designer has complete control. However, program designers do have substantial control over this factor, and Graphical User Interfaces were roundly criticized on this account. One Assimilator, a consistent user of MacIntosh and the System 7 Graphical User Interface, stated, "The Mac tends to be slow. This is particularly true with the Powerbook."

Running more than one program at a time is a fairly common issue with new operating systems that permit multi-tasking or running more than one program at a time. In this situation, the program that the user is working with at the time is considered to be the primary program, and the other programs that the computer is also working on are

considered to be working in background partitions. Graphical User Interface operating systems generally permit this capability and were criticized by a number of respondents for the slowness with which programs ran in background partitions. A college instructor stated, "The way in which applications slow down when they are put into background annoys me."

Two respondents were particularly annoyed with the amount of working memory that Graphical Interfaces consumed. A sociology student explained, "Windows tends to be a memory hog."

The amount of hard disk storage space consumed by Graphical User Interface programs was even criticized. One Computer Science Student, describing Microsoft Windows, explained, "First, it should be trimmed down so that it doesn't use as much processing time and disk space."

Menus. Individuals who are computer professionals often find menus to be restrictive. As such, many try to liberate themselves from the menu as soon as it is possible. One computer professional stated, "I hate them. They annoy me. This is something that I give end-users to work with because they need to be controlled and because they don't know how to do anything else."

In direct contrast, there was almost universal praise and acceptance for menus among the non-computer

professionals. This was true regardless of the learning style of the individual. Typically, most respondents found menus to be something that helped them to understand the program and that helped them to figure out where they were in operating the program. Special praise was reserved for menus that prompted adequately and that let users know what was expected of them at each stage of operations.

One participant stated, "I like menus. I don't find them to be confining. The person who finds them confining wants to experiment. I don't!" Another participant added to this sentiment by saying, "I like them. I don't find them hindering. This makes it easier to trace errors. Also, menus make it easier to discover when you have made a wrong turn." A school administrator summarized, "Menus are good. Otherwise, I wouldn't know where to find anything."

Why Individuals Dislike Software and Its Features.

The reasons people dislike software are as enlightening as the reasons they like software. Again, because the things that were disliked varied substantially with each individual, there was no relationship between learning style and the reason for disliking a program and its features.

Despite the lack of a discernable pattern that was attributable to learning style, common threads of complaint about software did emerge. One of the most common related

to features that the users have observed in other programs that their program does not possess. One participant criticized another word processing program because it "doesn't show page breaks and it could make more use of the function keys" like Word Perfect does. Another participant lamented the fact that Word Perfect for DOS did not have Windows-like features that would permit one to switch rapidly between applications. This participant explained, "I wish Word Perfect had a feature that permitted me to work back and forth between dBase and Word Perfect. This makes it more difficult to do complex jobs."

In addition to complaining about features that their software did not possess, participants were also prone to lament something called the application error, which occurs when the software ceases to function properly for no really good reason. One participant explained, "In Windows, I dislike the applications errors, especially the ones that shouldn't occur and that do, for no apparent reason." An accountant echoed this dismay about application errors by asserting, "Computers can be very scary. Recently, I lost everything on my hard disk. This was from a simple CMOS failure. The lights flickered, and all my stuff went south!"

In a number of instances, the inexplicable error that computer systems have been known to be prone to cause individuals to learn more about computers than they

expected to, or even wanted to. One respondent commented, "I really didn't want to know about this." Similarly, a college administrator lamented, "If you are going to work with computers on a regular basis and not be dependent on somebody else, you'd better count on learning more about computers than you wanted to!" Further, a college professor quipped, "You know, they really have a way of drawing you in. I developed more expertise than I wanted to! It was either that or lose my data!"

Mode of Interaction. With advancing technology, there are now more alternatives for choosing the way in which information is put into the computer system. The mode of input will strongly affect the operational characteristics of any software that is used on a computer system. Traditionally, the keyboard has been employed because most individuals who used computers made the transition from typewriters. However, other alternatives are available which can distinctly affect the functionality of software. Some of these alternatives include the mouse, the trackball (a close relative of the mouse), the touch screen, scanning devices, and pen-based data entry. Many respondents were familiar with only the keyboard. Most were familiar with the mouse. Of the other modes of data entry, only a small percentage of the participants were familiar with

trackballs. Participants had not been exposed to any other mode of data entry.

As with so many other aspects of computer usage, the pattern of reaction to the different types of data input systems was not associated with learning style. However, other patterns of reaction did develop.

Ten participants preferred a mode of data entry that is familiar to them. A participant commented, "I prefer the keyboard right now. It is what I am used to." Another participant stated, "I would have to take time getting used to a mouse. I prefer the keyboard because it is what I am used to."

Those who have made the transition to working with the mouse generally like it. A participant commented, "I prefer the mouse. It is easier and faster to use." Another stated, "I prefer the mouse. I am not totally uncomfortable with the keyboard. However, the mouse is faster for me." One mouse user who is a computer professional commented, "I like mousing better. Because I'm visual, I like to picture things. It seems easier but takes longer. It is less stressful, smoother, and more consistent."

However, praise for the mouse is not universal among those who use it extensively. Some users cite its clumsiness: "I prefer working with the keyboard. Mice annoy me. All too often, you run out of space in which to

move them." Ironically, one very strong advocate for Graphical User Interfaces commented, "It is easier to push two keys rather than moving your hand on a mouse to catch the menu and pull it down. Moving the mouse pulls my hand off the keyboard."

Observing individuals working with devices such as the trackball was particularly instructive. Many trackball devices are not constructed to make them easy for women to use. Most men can operate almost all trackball devices with just one hand. However, because of the relative smallness of their hands and their inability to touch all the required buttons easily, many women require two hands for operation of devices such as the Kensington Professional Mouse. This situation tends to exacerbate the clumsiness of pointing devices and to enhance the problems created by having to remove one's hands from the keyboard.

Expectations

In dealing with anything, one carries a set of expectations in encounters with peoples, places, and things. These expectations shape people's reactions to the things with which they are working, and they guide their reactions. Expectations are manifested by the perceived advantages and disadvantages of things. The same is true with computer systems. A secretary explained, "It really took me a long time to get used to working with Word

Perfect. I expected it to be hard, and it was. Now that I'm finally used to it, I like it!" The advantages and disadvantages of software and computer systems were discussed with the participants to get their perspective on expectations.

Advantages. Like many other aspects of the study, the perceived advantages of software and computers had no relationship to learning style. Nevertheless, common themes were repeated by the participants.

Most commonly, the advantage addressed by all participants related to an improved effectiveness of getting the job done. This highlighted other study results that suggested that most people's interest in computer systems was directed and determined by the role that the computer had in helping on the job. A typical comment was offered by a secretary: "My ability to use software cuts down on the time that I have to spend on tasks. It streamlines everything that I do. I can produce twice as much." Another secretary explained things in a similar vein: "Computers ease my workload. I can save and edit for somebody else without having to do a document all over again. It is a big time saver." Similarly, a college instructor explained, "It permits me to get things done faster. I get better quality work as a result. I don't have to redo so much."

Disadvantages. Much less of a pattern existed in the disadvantages that people found coming from the use of computers and software. Some folks cited the fact that the presence of computer systems did not actually lessen the level of work that an individual had to do; instead, the flexibility of the computer increased expectations and as a consequence workload. Other participants cited the ecological aspect of computers by explaining that, "Paper is wasted. This has an environmental impact." Still others complained that computer systems had a negative impact on typing skills because the editing features of computers meant that you did not have to get it right the first time. Others discussed the high cost and difficulty of obtaining training by saying, "It just takes too much time and money to keep current."

Despite the apparent variability of participants' reactions to the disadvantages of computer systems, some common themes did develop. One such theme was the overabundance and overavailability of information that is caused by the use of computer systems. One participant stated, "It becomes impossible to restrict information. You know, there are some things that people simply don't need to know." Another individual explained, "The computer produces so much extra information that it can lead to information overload." A nurse suggested, "One is the potential to generate a lot of garbage. People tend to

think that if it came from a computer, then it must be right."

In citing these hesitations about the availability of information, participants were cautioning about the tendency that exists for manufacturing excessive amounts of useless information. In addition, the study participants were concerned about the possible invasiveness into personal privacy because of the large amounts of information that are available because of computer systems. A college administrator exclaimed, "Computers can't help but invade your life! They keep so much information about you!" A computer professional explained, "They are too invasive. They know everything about you. With Social Security numbers, they can link to information about any part of your life!"

In another theme that was cited as a disadvantage of computer systems, some participants were concerned about individuals becoming too tied to computers. Many felt ill-at-ease in becoming too dependent on computers. A computer-support professional explained, "People find themselves relying on the computer more rather than using problem-solving abilities. They come to rely on the computer for things like grammar checkers and spell checkers." A secretary stated, "When they go down, the world freezes up!" An Air Force officer elaborated, "I can get too tied to a computer. Frankly, there are some tasks

that can be done better away from a computer, but I am too tied up with computers."

Impersonalization was a concern for a number of study participants. These individuals seem to feel that computers and computer software often caused them to be treated like numbers instead of individuals. This was something that these individuals definitely felt should be avoided. "Computers have taken away the personal element! There are limitations of the computer in the ability to treat the affective domain," asserted a graduate student. An undergraduate student in computer science said, "They exert too much control over our lives. Humans end up not making choices that they should. They end up giving responsibility to computers. People become robots." A librarian agreed, "Sometimes, computers bring an impersonalness to the process. For example, with computerized research, everybody has the same references in a paper."

The expense of computers and software was also mentioned by a number of participants as significant disadvantages of computer systems. A technical librarian explained, "They eat a hole in my pocket." A college administrator agreed, "Equipment becomes outmoded as soon as you buy it. Money to keep current becomes difficult to obtain."

CHAPTER 5

SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

Summary

The purpose of this study was to investigate the nature of the relationship between learning style and user interface preferences. In accomplishing this purpose, qualitative methods of interviews and observations were employed in working with individuals in a wide range of professions who participated in this study. Further, Kolb's Learning-Style Inventory was used to provide triangulation of the qualitatively assessed learning style of each of the participants.

A major finding from observing study participants was that learning style was not a major factor in an individual's reactions to different computer user interfaces. Rather, their choice of and preference for a particular interface was that it helped them in their jobs and was available for them to work with in the accomplishment of job-related tasks.

However, this does not mean that learning style was not a factor in the ways in which individuals approached computer systems. For example, learning style did affect

the level of analysis that a person was prone to do in considering what made for good user interfaces and bad user interfaces. For example, it was a Converger who suggested the themes of simplifying menus and icon lists to enable users to more effectively work with more limited lists of tasks.

In addition to the preceding issues, it was discovered that an individual's preferred learning strategy could significantly affect this person's approach to working with computers. In this regard, the resource management approaches that an individual used in general were carried forward to work with computers, and these resource management approaches affected such issues as whether or when an individual would contact somebody else for help, whether an individual would use manuals to learn, or whether the individual would enroll in a class related to computers.

Different types of problem solvers in relationship to the computer were identified. These included Resource Managers, Mechanics, Fiddlers, Confronters, and the Intimidated.

Furthermore, some interesting information was revealed about female computer users and their resource management approaches. For a number of female computer users, independence may be a significant issue affecting their

approach to resource management and their work with computer people.

Also, some of the findings would affect teaching strategies. Not everyone needs background-associated information to effectively learn computer-related subjects. Those with a preference for Abstract Conceptualization requested that they be given just the information that they needed to deal with the immediate task. However, others stated that background information helped in the learning process.

A major portion of the findings of the research concerned the different things that individuals liked and disliked about software. Broad patterns of likes and dislikes for both Command Line and Graphical User Interfaces emerged. Issues such as control, flexibility, ease of use, and access to computer resources were cited by various users as having different levels of importance, depending on their already confirmed preferences for different types of software.

Conclusions

Conclusions for Teaching and Learning

Based upon the findings that have come from this study, the following conclusions pertaining to the teaching and learning process are drawn:

Learning style as described by Kolb's framework is not an effective predictor of an individual's success in working with computer systems.

Learning style affects the way in which individuals work with the language of computers.

Learning style is related to the level and amount of abstraction that individuals prefer to have in working with the subject of computing.

Regardless of the learning style of the individual using them, computer manuals are not adequately designed to meet the needs of the vast majority of computer users.

The distinct categories of problem-solvers and learners with respect to computers include Fiddlers, Mechanics, Confronters, Resource Managers and the Intimidated. A person's placement in one of these categories can affect how a person reacts to computers.

Conclusions for Computers

Based on the findings that have come from this study, the following conclusions can be made for computers:

Interface preference is a function of task-related factors rather than an element of cognitive processing.

The learning style of an individual can affect the language that one uses in talking about computer preferences.

The preponderance of computer use is task related.

The design of computers and software systems needs to be enhanced to meet the needs of the tasks for which they are intended.

In continuing to design computer programs that are large, clumsy, complex, and largely non-task related, software designers are missing some essential issues about what users need in their software.

Problems have arisen in the way in which computer professionals interact with segments of organizations.

Qualitative research methods can be used to augment the data gathering tools of the systems analyst.

Discussion

The implications from these findings and conclusions have the potential to be far-reaching. These implications exist in the areas of teaching strategies, new software architectures, the appropriate use of existing computer technologies, and approaches to systems design and development.

Teaching Strategies

Since there is not universal acceptance among students of the importance of background information in their learning processes, computer instructors should consider the mixture of preferences that students might have in planning their approach to teaching a computer-based subject. This is simply the result of students bringing diverse learning styles and preferences into the classroom. Because learning style affects student perceptions of the classroom learning process with any subject including computing, computer instructors for the sake of classroom effectiveness must understand from where students are coming. There are instruments that can help the instructor

in this assessment of student preferences for learning in the computer environment. These instruments include Kolb's Learning-Style Inventory (1985) and Witkin's Embedded Figures Test (Oltman et al., 1971).

Assessing Student Learning Preferences. Since the preponderance of individuals who liked and needed background information to truly master a subject had a preference for Abstract Conceptualization as a component in their learning style, instructors in introductory computer areas might consider using Kolb's Learning-Style Inventory as a tool for determining how many individuals rated strongly in a preference for abstraction. Given an assessment of the learning style preferences of their students, the instructor could tailor the mixture of theory and practice to match. In this fashion, instructors might be more sure to present something that is appealing to each student in their classes.

Additionally, the need for background information as compared to a preference for just the material that is pertinent to the particular subject has been discussed by researchers in terms of Field Dependence and Field Independence (Witkin, 1978, pp. 15-17). Determining whether a person needs background information or external referents (Field Dependence) or if a person can do without this extra information (Field Independence) can be assessed

by the Embedded Figures Test (Witkin & Goodenough, 1981, p. 15). Instructors might want to consider using this test to help them determine the degree to which background and theoretical information should be included in the classroom presentation.

Approaching the Subject. Because of the diversity of approaches that students bring to learning, an instructor should carefully monitor the approach that is employed in presenting the subject in the classroom. This comes from some students having a need to think in the abstract and from some students disliking the abstract. Consequently, to appeal to both sensibilities, an instructor should never go too long into the theoretical before introducing some practical demonstrations and hands-on work. Since the vast majority of participants were primarily interested in applications software, it should be safe to assert that the importance of early hands-on work is particularly important in classes that are oriented toward teaching individuals the use of applications software.

Problem Solving Approaches to Computers. In working with students in the classroom, teachers can make effective use of the categories of Confronter, Fiddler, Mechanic, Resource Manager, and the Intimidated to help their students with the learning process. The primary role that these categories can play relate to the amount and kinds of

help that the facilitator will want to provide individuals from the various groups.

The Intimidated will need the most help and the most encouragement. These individuals can be easily identified by their evident hesitancy to work with computers. When doing hands-on work in the classroom, they will rarely do anything unless explicitly told to do so. Consequently, individuals falling into this category need the maximum amount of patience and tolerance from the instructor when the class is engaged in lab work.

Confronters, Mechanics, and Fiddlers are generally capable of fairly independent work. In fact, they will generally have a tendency to try and explore on their own. While this can make group lab work somewhat complicated for the computing facilitator, the instructor should try to be as encouraging of this activity as possible. This will simply come from the fact that individuals from these categories will probably learn more from their explorations than from class activities.

Resource Managers will be identified because of their readiness to ask questions of their peers in the classroom. Again, this can be disruptive by traditional standards, but since it is an integral part of the approach that many have to learning computers, it should be encouraged.

Another area in which these categories of learners can have an effect resides in the ability of students to

benefit from independent and research assignments in the area of computers. Because of the inherent stubbornness of Confronters and because of the joy of tinkering that exists in Fiddlers, members of these two groups stand to gain the most from independent assignments.

Flaws in Computer Manuals. Computer manuals as they are currently designed do not facilitate the learning process. If anything, they harm it. Consequently, designers of computer software should take a hint from the large computer-book publishing industry and start designing their manuals to match the ways in which the book publishers organize their products. Instructors should be prepared to overcome these hurdles by having a reference list of usable books that are available for students. Further, the instructor might have a presentation planned in manual decoding.

New Software Architectures

Presently, menus in their various forms dominate the way in which users interact with software. This is true for both Graphical User Interface based computer systems and Command Line Interface based computers. In many instances, the only real difference between these two paradigms is simply that Graphical User Interfaces use icons (program symbols) and pop-up menus that are activated by a pointing device such as a mouse to get at the various

functions which are controlled by the menu. This is highlighted by the fact that recent versions of MS-DOS come with a shell program which acts like a menu for the various commands that the user would otherwise have to activate by a command to the Command Line Interface. The essential role of the menus is to act as a central control point. It receives the user's request for a program function. It then causes the activation of the appropriate set of functions in the program, which was requested by the user. The process is quite simple: request and response.

It is in this simplicity that lies much of the complexity of many software systems. To provide all of the different functions that many users require in their programs, the menus that are used to respond to this large number of different potential user requests have to be very complex. In many instances, these menus have to be long. In other instances, these menus have to be multi-level and convoluted to provide for the different combinations of functions that are required by a large heterogeneous group of software users.

Use of Intelligent Agents to Improve Software. In the study, a number of users pleaded for simplified software. It is very apparent that software users are thoroughly frustrated with software which is clearly the product of the refusal of computer companies to provide user friendly

programs. Some complained about too many menu options. Others decried convoluted functions that required too many steps to get a job done. Still others suggested that dedicated systems where the computer was intended to accomplish a single function were superior to the multiple use computers and their complexity of operation. One group suggested that computers could adjust themselves to the user's abilities. Given all these different cries for help, corporate software developers should take more direct action to insure that programs are friendly and easy to use.

A growing new technology that is referred to as Intelligent Agents could help to solve many of the problems of excess software complexity that the participants have helped to identify. While there are many aspects to the definition of Intelligent Agent, generally it is software that will adaptively "engage and help all types of end users" by being able to use a body of "shared knowledge" in order to communicate the needs of the end users that they are designed to serve (Reiken, 1994, pp. 20-21).

In the context of the complaints of excess complexity mentioned by participants, software functionality could be immensely improved through the addition of Intelligent Agents. Instead of equipping computer programs exclusively with menu generation subroutines whose only purposes are to present menu choices for users and to launch the

appropriate functions when the user makes a request for an operation, an Intelligent Agent could be used in combination with a menuing program to determine the functions with which a user has been working and to determine at a probabilistic level what functions the user is likely to need.

The menu and the Intelligent Agent for available functions would work together very extensively. In addition to its traditional functions of presenting options and receiving instructions, the menu program would also record information into a data structure called a Transaction Log. The Transaction Log would simply be a running list of the functions that the user had called for over the last few minutes of operations. Using the Transaction Log, the Intelligent Agent for available functions would calculate the functions that the user is likely to need when a menu is next presented for them. This information would be recorded into a data structure called Available Items.

Several benefits would accrue from having devices such as an Intelligent Agent for available functions. First, menus could be immensely simplified. Instead of having menus with 15 or more items as some large complex programs do, just a few relevant items could be presented. This follows the principle of user interface design that the program should never place too much reliance on the memory

of the user (Mandel, 1994, pp. 77-94). In asking users to remember and differentiate between too many different options in a program, this is exactly what is being done.

Secondly, having programs with many different options that are continually available tends to make program operations unnecessarily convoluted. Different program operations tend to have some functions in common. This means that users can often end up with the same result by starting with several different initial menu positions. Again, all these different ways of getting the same result places unnecessary emphasis on the user's memory to have them remember which combination of menu options is the optimal one. With excessively complicated menu systems that have overlapping functions, individual users of computer systems can forget which option is the preferred one and can use a needlessly complicated one.

One of the most attractive aspects of having Intelligent Agents tracking user work patterns with software and having Intelligent Agents determine the functions that should be presented for users within menu structures is that this would result in software that would automatically adjust to the ability and needs of the current user. While the particular functions that would be included at any one particular time by the Intelligent Agent into the options available for presentation by the menu program would have to vary from user to user based on

use patterns that would alter with each different computer worker, the different functions that should be included could be effectively calculated using well defined probabilistic methods such as Bayesian Calculation, Certainty Factors, Fuzzy Logic, or Criterion Tables (Barr, Cohen, & Feigenbaum, 1989, pp. 171-172). While this is internally sophisticated, it would produce the end result desired by the participants: simple, effective software.

Appropriate Use of Existing Computer Technologies

For many people, the first encounter that they have with computer software is through the install program that leads them through the process of transferring a program from floppy disk onto the hard drive. However, most program developers put relatively little effort into the design and implementation of this process that is certain to affect the way in which individuals view their product. In this regard, most developers of computer programs tend to give most software users relatively few options concerning the implementation of different functions and capabilities. Therefore, the essential consequence of this practice is that any program looks essentially the same no matter the computer on which it is installed, with minor allowances being made for the availability of different printers or monitors. These approaches to software are in

direct conflict with the desire that has been repeatedly expressed by participants that software should be built that gives them just the functions that they need and that programs should be built so that they do not have to be bothered with program capabilities that will never be used. Therefore, any existing technologies that can appropriately be used to answer this cry for software simplicity should be implemented in the internal organization of program systems.

To accommodate the requests from different participants in this study that software be simplified, that it be less of an absorber of system resources, and that it be less convoluted in its operation, software manufacturers should consider the application of Expert Systems technology to help determine the functions that should be installed from the program's installation floppy disks. Expert Systems are computer programs that take an expert's knowledge of a situation or process and codify that knowledge in a collection of information called a rule-base. This rule-base represents computer-accessible knowledge of how an expert would act in a particular situation. Therefore, Expert Systems can be set up to ask the potential user of a piece of software that is being installed questions about how this individual uses and works with software. Based on the user's responses, the program can use the expertise that has been programmed into

the rule-base to make a selection of the functions that will be needed by that particular program's user. The appropriate functions could then be moved onto the hard drive during the installation process. Since Expert Systems technology would only be present during the installation process, the resulting software that would be present permanently on the hard disk would be simpler, smaller, and easier to use.

Several distinct benefits would accrue from applying advanced Expert Systems technology during the installation process. First and foremost, manufacturers of software could still sell software with all the functions that are necessary for a program to compete in the modern software marketplace. However, the material that would be transferred to the hard drive of the computer user would only be that which is necessary to handle the needs and problems of the individual in question. In this way, programs that individuals purchased would tend to consume less of the hard-drive resources of a person's computer.

Another benefit of using Expert Systems technology to help the user choose the functions that are to be installed on computers is that the smaller programs that would be installed would be easier to learn and use. This would simply result from there being fewer options in the programs for users to learn, understand, and differentiate. Having smaller, simpler computer programs would directly

answer a major request repeatedly sounded by study participants.

Approaches to Systems Development

Several aspects of the findings have the potential to affect the viewpoint of individuals who analyze and design new computer-based information systems. The areas with the potential for change include methods and tools for systems analysis and using the tools of qualitative research to better understand the corporate culture in which computer systems exist.

Methods and Tools for Systems Analysis. A knowledge of learning styles can be an effective tool for the systems analyst. While preferences for software systems are not linked to learning style, some aspects of this concept are of definite use to the computer person in the process of understanding, designing, and installing effective computer systems.

The role played by learning style in the analysis, design, and development of computer-based information systems is related to a phenomenon that has long been identified by computer people. There can sometimes be a significant amount of resistance from end-users to the installation of computer systems (Whitten et al., 1989, pp. 34-37). While this resistance can come from a number of sources including the poor design of computer systems

and an unwillingness of people to change, analysts can make use of tools such as Kolb's Learning-Style Inventory to help them identify people who might aid in the analysis, design, and implementation of computer systems and who might help in breaking down the resistance that many of the other potential users of computer systems might have.

One aspect of learning style that was highlighted by participants was that those individuals who have a penchant for abstraction also have a tendency to be more analytical in their thinking about computer systems and the software that is used to implement them. In this regard, they tend to think in more depth about the nature of good and bad software, and they tend to speak more voluminously about the nature of good and bad computer programs.

Therefore, one very effective role of Kolb's Learning-Style Inventory could be to help the analyst identify end users who would be more interested in speaking in depth with the analyst about the good and bad aspects of the software or computer system that is being designed. A direct benefit of this could be a more effective computer system because the analyst would be able to obtain a more complete picture of the problems and needs of the end users who would be working with the software that moves the information.

Furthermore, the analyst could employ these end users as sales people for the computer system in question.

Numerous management specialists have identified this process of identifying individuals to target in spreading information, insight, and the acceptance of new approaches to management. This is sometimes referred to as the creation of cadre (Siu, 1979, pp. 77-83).

Other aspects of methods and procedures in systems analysis can be affected by the results from this study. This is emphasized by the following conclusion: Qualitative research techniques can be an effective tool in the systems analysis and design process. A number of study findings highlight this. In recent years, computer software designers have emphasized the development of programs with increasing numbers of user options for each and every function. This tendency has been identified and described by individuals working in the field as the development of "Fatware." In general, Fatware has had the tendency to add more and more functions at the price of speed, consistency, and effectiveness of software function (Perratorre, Thompson, Udell, & Malloy, 1993, p. 98). In direct contrast to this tendency in software, the study participants repeatedly asked for software that operates simply and consistently and which configures itself directly to the needs of the user. It seems that with all the extensive analysis and design tools that have been employed, computer people have missed a fundamental truth about what software users really want. A research tool

that is designed to uncover this type of information is qualitative research. Therefore, people who design computer software should pay more attention to qualitative research methods in trying to understand the people with which they are working. In addition, software development organizations which can afford it should consider having a qualitative researcher on staff to help the more technically-oriented systems analysts truly understand the nature of user working environments, needs, and wants.

Because at least one fundamental truth about what users want in their software has been missed, a re-appraisal of curriculum for professional systems analysts is in order. Currently, this curriculum includes subjects such as systems analysis methods, data organization analysis, project planning, electronic data processing administration, business economics, management, computer programming, systems testing, systems implementation, computer technology, prototyping, communications technology, and Computer Aided Software Engineering (Lord, 1990, p. vii). Little, if any, explicit attention is paid to the social, psychological, and cultural factors that may affect the ways in which computer systems and software are employed in organizations. It is issues of this nature that qualitative research methods have been designed to assess. Therefore, readings on qualitative research should be included in the curriculum of study of potential systems

analysts. Additionally, for organizations that can afford it, full-time qualitative researchers could be put on the staff to work with systems analysts to help alert these analysts to issues that might otherwise be skipped because the majority of their educational background comes from computer technology and is based on the quantitative research methodology. Readings and cooperative work with qualitative researchers could have an additional advantage for current systems analysts; recognizing that they have a lot in common with qualitative research in the use of interviews and artifact gathering as the basis for learning about people (Whitten et al., 1989, pp. 726-740) will give analysts an additional perspective on research methods.

Computers in the Corporate Culture. Qualitative research methods have the ability to identify cultural biases that may exist within the organization that can affect the ability of the analyst to implement an effective computer-based software system. As discovered by discussion with study participants, female computer users often raised the issue of being able to operate independently as being important to them. This was further highlighted by females mentioning that they were reluctant to talk to the data processing department.

Since the cooperation of end users is critical to the success of information systems implementation, systems

analysts applying qualitative research methods should be sensitive to the possibility that cultural difficulties within the organization may exist with respect to the ways in which female computer users interact with data processing departments. These difficulties in interaction can affect the system's analysis, design, and implementation process in a number of ways. First, reluctance to speak and work with information systems departments may make it difficult to obtain complete information about how organizations function in certain operational contexts. Since the role of computer systems in business is to support the organizational function of the business entity, the completeness of information about how an organization functions is absolutely critical to the process of effective systems analysis.

The second major way in which problems in the interaction of female computer users and data processing departments can negatively affect the success of computer-based information systems lies in the realm of systems implementation. If female computer users find it difficult to rely on data processing people for help, this reluctance may mean that help will not be obtained at a critical juncture in the operation of a system when difficulties with computers do occur as they inevitably will. Spread widely enough through a system, this could cause the

failure of a computerization project or seriously affect its efficiency.

However, a number of potential solutions exist for dealing with these difficulties. Most of these solutions revolve around developing computer organizational structures that are more friendly and more accessible to the reluctant or hesitant computer user.

One potential partial solution to making computer systems more accessible for the user who needs operational independence from data processing departments resides in the development of computer systems that are based on the distributed processing paradigm. In contrast to traditional data processing methods involving centralized computers with individuals accessing information through terminals, distributed data-processing envisions the widespread use of microcomputers that share data and programs through local area networking technology. With traditional data processing, all functions and access to data are controlled by the central computer and the individuals who program it. With distributed data processing, users get to employ personal computer software with which they are already familiar in order to manipulate the information in the computer (Kahn, 1992, p. 227). Because of this ability to let users employ microcomputer-based software with which they are already familiar, the ability of users to be independent of centralized data

processing departments could be enhanced. This ability for enhanced operational autonomy when working with computers could be of distinct benefit to those individuals for whom independence is an issue.

User programming is another possible solution for those who need additional independence from data processing departments. User programming assumes the use of advanced applications software such as spreadsheets, word processors, or data base management systems that can be easily modified by non-technical software users to accomplish tasks that would normally only be accomplished through programming. This ability is enabled through the presence of user-friendly, high-level query languages that permit users to learn enough about the program and data retrieval to handle their own common data inquiries after a small amount of work with the application software. The net effect of the presence of software systems of this sort is that the role of computer technicians will be that of trouble-shooters whose help will only be needed by the user when the most extreme problems occur with the computer and the software (Wetherbe & Vitalari, 1994, pp. 342-343).

Clearly, in addition to the traditional approaches of determining the components of the architecture of computer-based information systems, another set of ingredients should be considered in deciding the shape that these computer systems should have. These additional factors are

related to culture, and they extend far beyond the traditional bounds of the organizational culture that traditional management specialists have examined extensively. However, these additional factors appear to have the potential to affect computer people's ability to successfully implement computer systems.

Perhaps the most direct solution to the cultural issues that may be present in the design, development, and implementation of information systems is to encourage more women to take up computer-related professions. It may be possible that women would be better able to get technical help from technologically-oriented women. This might be due to some individuals being able to more effectively communicate with people of the same gender. Unfortunately, the current male dominance of technologically-oriented computer science and telecommunications fields is likely to continue. While more women work with technology than before, there is still a preponderance of men who earn technical degrees. Recent estimates suggest that the ratio of men to women earning Computer Science degrees exceeds 3 to 1 and that it is growing (Rosenberg, King, & Springen, 1994, p. 51).

Recommendations for Further Research

A number of potential areas for research have been raised by this study. Several of these areas are cultural in nature.

First, it could be useful to determine the complete nature of the relationship between women and those individuals in computer-related technical disciplines. Such investigations could determine the extent of the conflict between these groups and why the conflict exists. Further, it could help to discover whether or not female technical people are actually more effective in providing help to female computer users. Also, discovering the nature of the barriers that discourage females from going into technical areas such as computing would also be beneficial.

While Kolb's Learning-Style Model was not effective in explaining why individuals react to computers as they do; it did describe some behaviors that individuals had with respect to computer systems. Therefore, future studies could examine some other models of learning style and consider the relationships that they have to the ways in which people work with computer systems. Among the models that might be examined are Witkin (1978), Canfield (1983), Reichmann and Grasha (1974), and Entwistle (1981).

In the area of software design, a number of areas of research can be pursued. First, the area of adaptive interfaces and menuing systems should be considered. Menu systems that use artificial intelligence technology to adapt their contents and meaning to the level of ability of the user should be developed. It would be useful to see if users would be able to adapt more quickly to software that operated on this interface paradigm. Furthermore, if users did indeed adapt more readily to software functioning with an automatically adaptive interface, studies could examine how much faster users were able to learn software that functioned using an interface of this style.

A second issue worth further study in the area of software design relates to the use of Expert Systems technology during the software installation process. Studies in this area could determine whether installation software could be written with Expert Systems technology so that user preferences for software functionality could be effectively predicted. Further, it would be helpful to find out if software developed and installed according to this method actually operated more efficiently than traditional software. These studies could utilize the numerous methods that have already been developed for measuring software operational efficiency. Devices such as hardware and software monitors which can count the number of units of information processed by a program per unit

time can be effectively employed to determine relative program efficiency (Pressman, 1992, p. 653).

The Future

While learning style may not be a direct determinant of how individuals react to computer software, it will have a relationship to the way in which people think about software and about computer systems. As a consequence, a knowledge of learning style can be an effective tool for both software and systems designers in determining who can be counted on for an in-depth analysis of computer functionality. Therefore, a knowledge of learning styles should be integrated into the curricular structures of software engineering and systems analysis. Such knowledge could help analysts and engineers be better practitioners of their craft.

Since organizational and cultural issues appear to have more of an impact on computer systems than was previously thought, technical people will benefit in their ability to do an effective job by an exposure to those cultural issues that can affect a person's behavior in the organization. This means that technical individuals should not only be exposed to qualitative research methods to help them assess aspects of the corporate culture that may affect the behavior of individuals within the organization, but also technical individuals should be exposed to the

broad range of issues that have been labelled under the rubric of cultural diversity. Without the background of having been exposed to the various ways in which different cultures experience the world and thence technology, no collection of methods is going to give the technical person the vision to understand how the tools of technology are going to affect the individuals in the organization.

As a consequence, the drive to create software and computer systems will be based on enhancing the education of the technical individuals who are detailed to design the computer systems and software on which so many rely. However, the necessary education will be one that is broad in its selection of subjects. This should include a strong expose to those issues which affect the culture of an organization.

REFERENCES CITED

- Adler, M. (1990). Intellect: Mind over matter. New York: MacMillan.
- Ahituv, N., & Neumann, S. (1990). Principles of information systems for management. New York: William C. Brown.
- Andersen, P. B. (1990). A theory of computer semiotics: Semiotic approaches to the construction and assessment of computer systems. Cambridge: Cambridge University Press.
- Ausubel, D. (1986). A cognitive structure theory of school learning. Learner Behavior, 207-257.
- Barba, R. (1990). The effects of embedding generative cognitive strategies in science software. Journal of Mathematics and Science Teaching, 10(1) 59-65.
- Barr A., Cohen, P., & Feigenbaum, E. (1989). Handbook of artificial intelligence, Volume IV. Reading, MA: Addison-Wesley.
- Bonham, L. A. (1988). Learning style instruments: Let the buyer beware. Lifelong Learning, 11(6), 12-16.
- Brooks, F. (1982). The mythical man-month: Essays on software engineering. Reading, MA: Addison-Wesley.
- Canfield, A. A. (1983). Canfield Learning Styles Inventory, Form S-A Manual. Birmingham, MI: Humanics Media.
- Certo, S., & Lamb, S. (1979). Identification and measurement of instrument bias within the learning style instrument through a monte carlo technique. Southern Management Proceedings, 1979.
- Chadwick, S., & Watson, J. (1986). Computers and cognitive styles. Academic Therapy, 22(2), 125-132.
- Chu, P. C., & Spires, E. E. (1991). Validating the Computer Anxiety Rating Scale: Effects of cognitive style and computer courses on computer anxiety. Computers in Human Behavior, 7, 7-21.
- Clariana, R., & Smith, L. (1988). Learning style shifts in computer-assisted instructional settings. Paper presented at the International Association for Computers in Education. New Orleans, LA.
- Cline, J. (1991). Cognitive style in system design. Thesis, Kent State University.

- Coan, R. (1978). Meyers-Briggs Type Indicator. In O. Buros (Ed.), Mental measurements yearbook Vol. 1 (pp. 629-631). Highland Park, NJ: Gryphon Press.
- Conti, G. J., & Fellenz, R. A. (1992). Assessing adult learning strategies. Proceedings of the 32nd Adult Education Research Conference. Norman, OK.
- Conti, G. J., & Welborn, R. B. (1986). Teaching-learning styles and the adult learner. Lifelong Learning, 9(8), 20-24.
- Cordell, B. (1990). A study of learning styles and computer assisted instruction. Computer Education, 16(2), pp. 175-183.
- Cringely, R. (1992). Accidental empires: How the boys of Silicon Valley battle foreign competition, and still can't get a date. Reading, MA: Addison-Wesley.
- Davidson, G., Savenye, J., & Orr, K. (1992). How do learning styles relate to performance in a computer applications course? Journal of Research on Computing in Education, 24(3), 348-358.
- Davis, R. R. (1991). Learning how to learn: Technology, the seven multiple intelligences and learning. Paper presented at the May 11, 1991 CUE Conference, Palm Springs, CA.
- Davis, J. F., Murrell, P. H., & Davis, T. M. (1988). On matching teaching approach with student learning style: Are we asking the right question? Paper presented at the Mid-South Educational Research Association, Louisville, KY, November 8-11, 1988. (ERIC Document Reproduction Service No. ED 303 859)
- Deitel, H. (1990). Operating systems. Reading, MA: Addison-Wesley.
- Dunn, R., & Dunn, K. (1978). Teaching students through their individual learning styles: A practical approach. Reston, VA: Reston Publishing Company.
- Ellis, D. (1989). A behavioral approach to information retrieval systems design. The Journal of Documentation, 45(3), 171-211.

- Entwistle, N. (1981). Styles of learning and teaching: An integrated outline of educational psychology for students, teachers, and lecturers. New York: John Wiley & Sons.
- Ferrell, B. G. (1983). A factor analytic comparison of four learning-styles instruments. Journal of Educational Psychology, 75(1), 33-39.
- Fischer, G., & Lemke, A. (1988). Constrained design processes: Steps towards convivial computing. In Raymonde Guidon (Ed.), Cognitive science and its applications for human-computer interaction. Hillsdale, NJ: Lawrence Erlbaum Associates.
- Foreman, K. (1990). Cognitive characteristics and initial acquisition of computer programming competence. School of Education Review, 2, 55-61.
- Fox, R. (1984). Learning styles and instructional preferences in continuing education for health professionals: A validity study of the LSI. Adult Education Quarterly, 35(2), 72-85.
- Gardner, H. (1993). Frames of mind: The theory of multiple intelligences. New York: Basic Books.
- Geiger, M., & Boyle, E. (1992). A factor analysis of Kolb's revised learning-style inventory. Educational and Psychological Measurement, 52, 753-759.
- Glaser, B., & Strauss, A. (1967). The discovery of grounded theory. Hawthorne, NY: Aldine De Gruyter.
- Gould, S. J. (1981). The mismeasure of man. New York: W.W. Norton & Co.
- Grabinger, R. (1989). Screen layout design: Research into the overall appearance of the screen. Paper presented at the annual conference of the Association for Educational Communications and Technology, Dallas, TX.
- Gregg, N. (1989). Learning-style inventory. In J. C. Conoley & J. J. Kramer (Eds.), The tenth mental measurements yearbook (pp. 441-442). Lincoln, NB: University of Nebraska Press.
- Gregorc, A. (1979). Learning/teaching styles: Potent forces behind them. Educational Leadership, January, 1979, 234-236.

- Gypen, J. (1980). Learning style adaptation in professional careers: The case of engineers and social workers. Unpublished doctoral dissertation, Case Western Reserve University.
- Hall, E. T. (1976). Beyond culture. New York: Anchor Books.
- Helms, J. (1992). Why is there no study of cultural equivalence in standardized cognitive ability testing? American Psychologist, 47(9), 1083-1101.
- Hunt, D., Butler, L. Noy, J., & Rosser, M. (1978). Assessing conceptual level by the paragraph completion method. Toronto: Ontario Institute for Studies in Education.
- Hunt, M. (1993). The story of psychology. New York: Doubleday.
- Ince, D. (1988). Software development: Fashioning the baroque. Oxford University: Oxford Science Publications.
- Ivey, B. (1992). A case study of student learning in a microcomputer-based chemistry laboratory. Unpublished doctoral dissertation Montana State University.
- Jung, C. G. (1921). Psychological types. Princeton, NJ: Princeton University Press.
- Kagan, J. (1965). Impulsive and reflective children: Significance of conceptual tempo. In J. Krumboltz (Ed.), Learning and the educational process (pp. 133-161). Chicago: Rand McNally.
- Kagan, J. Rosman, B.L., Day, D., Albert, J., & Phillips, W. (1964). Information processing in the child: Significance of analytic and reflective attitudes. Psychological Monographs, 78(1, Whole No. 578).
- Kahn, P. (1992). Online complex processing. In Rick Segal (Ed.), Insider's guide to personal computing and networking. Carmel, IN: Sam's Publishing.
- Katz, N. (1986). Construct validity of Kolb's Learning-Style Inventory, using factor analysis and Guttman's space analysis. Perceptual and Motor Skills, 63, 1323-1326.

- Keefe, J. (1982). Assessing student learning styles: an overview. In J. Keefe (Ed.), Student learning styles and brain behavior. Reston, Virginia: National Association of Secondary School Principles.
- Keirse, D., & Bates, M. (1978). Please understand me: Character and temperament types. Del Mar, CA: Prometheus Nemesis Book Company.
- Kildall, G. (1994). CP/M: A family of operating systems. In Jay Rana and Alan Nash (Eds.), The best of Byte: Two decades on the leading edge. New York: McGraw-Hill, 205-211.
- Kolb, D. A., Ruskin, I. M., & McIntyre, J. M. (1974). Organizational psychology: A book of readings (2nd ed.). Chicago: University of Chicago Press.
- Kolb, D. A. (1976). Learning-Style Inventory technical manual. Boston: McBer.
- Kolb, D. A. (1981). Learning styles and disciplinary differences. In A. W. Chickering (Ed.), The modern American college (pp. 232-254). San Francisco: Jossey-Bass.
- Kolb, D. A. (1985). Learning-Style Inventory. Boston: McBer and Company.
- Kolb, D. A., Ruskin, I. M., & McIntyre, J. M. (1974). Organizational psychology: A book of readings (2nd ed.). Chicago: University of Chicago Press.
- Levy, S. (1984). Hackers: Heroes of the computer revolution. New York: Dell.
- Long, L. (1991). Introduction to computers and information processing. New York: Prentice Hall.
- Lord, K. (1990). CSP review manual: Tools of the systems professional (2nd ed.). New York: Van Nostrand Reinhold.
- Mandel, T. (1994). The gui--ooui war: Windows vs. OS/2. New York, Van Nostrand Reinhold.
- Marsh, S. (1990). Human-computer interaction: An operational definition. SIGCHI Bulletin, 22(1), 16-22.
- Martin, M. (1991) The analysis and design of business information systems. New York, MacMillan.

- Merriam, S. B. (1988). Case study research in education: A qualitative approach. San Francisco: Jossey-Bass.
- Merritt, S. L., & Marshall, J. C. (1984). Reliability and construct validity of ipsative and normative forms of the learning-style inventory. Educational and Psychological Measurement, 44, 463-472.
- Nakayama, G. (1988). A study of the relationship between cognitive styles and integrated science process skills. Presented at the National Association for Research in Science Teaching, Lake Ozark, MO. (ERIC Document Reproduction Service No. ED 291 592)
- Norman, D. A. (1987). Cognitive engineering, cognitive science. In John M. Carrol (Ed.), Interfacing thought: Cognitive aspects of human-computer interaction (pp.325-336). Cambridge, MA: MIT Press, 1987. 325-336.
- Nunney, D. (1978). Cognitive style mapping. Training and Development Journal, September 1978, 50-57.
- Oltman, P., Raskin, E., & Witkin, H. (1971). Group Embedded Figures Test. Palo Alto, CA: Consulting Psychologists Press.
- Paterson, T. (1994). An inside look at MS-DOS. In Jay Ranade and Alan Nash (Eds.), The best of Byte: Two decades on the leading edge. New York: McGraw-Hill, 212-219.
- Perratorre, E., Thompson, T., Udell, J., & Malloy, R. (1993). Fighting fatware. Byte, April, 98-108.
- Pressman, R. (1992). Software engineering: A practitioner's approach (3rd ed.). New York: McGraw-Hill.
- Ranade, J., & Nash, A. (1994). Introduction. In Jay Ranade and Alan Nash (Eds.), The best of Byte: Two decades on the leading edge. New York: McGraw-Hill, 1994, 4-7.
- Reichmann, S. W., & Grasha, A. F. (1974). A rational approach to developing and assessing the construct validity of a student learning style scales instrument. Journal of Psychology, 87, 213-223.
- Reiken, D. (1994). Intelligent agents. Communications of the ACM, 37(7), 20-21.

- Rowland, P., & Stuessy, C. (1988). Matching mode of CAI to cognitive style: An exploratory study. Journal of Computers in Science and Mathematics Teaching, 7(4), 36-40, 55.
- Rosenberg, D., King, P., & Springen, K. (1994). Men, women, and computers. Newsweek, May 16, 48-55.
- Rotter, J. B. (1966). Generalized expectancies for internal versus external control of requirements. Psychological Monographs, 80(10), 1-28.
- Ruble, T., & Stout, D. (1990). Reliability, construct validity, and response set bias of the revised Learning-Style Inventory (LSI-1985). Educational and Psychological Measurement, 50, 619-629.
- Sewall, W. (1986). Observations of end-user online searching behavior over eleven years. Journal of the American Society for Information Science, 37(4), 234-245.
- Sharma, S. (1987). Learners' cognitive styles and psychological types as intervening variables influencing performance in computer science courses. Journal of Educational Technology Systems, 15(4), 391-399.
- Shum, S. (1991). Cognitive issues in representing design reasoning as hypertext. SIGCHI Bulletin, 23(1), 38-40.
- Siu, R. G. H. (1979). The craft of power. New York: John Wiley and Sons.
- Stake, R. E. (1988). Case study methods in educational research: Seeking sweet water. In R. M. Jaeger (Ed.), Complementary methods for research in education (pp. 253-270). Washington, DC: American Educational Research Association.
- Van Merriënboer, J. (1990). Instructional strategies for teaching computer programming: Interactions with the cognitive style reflection-impulsivity. Journal of Research on Computing in Education, 23(1), 45-53.
- Weinberg, G. M. (1971). The psychology of computer programming. New York: Van Nostrand Reinhold.
- Wetherbe, J., & Vitalari, N. (1994). Systems analysis and design: Best practices. St. Paul, MN: West Publishing.

- Whitten, J., Bentley, L., & Barlow, V. (1989). Systems analysis and design methods. Boston: Irwin.
- Williams, G. (1994a). Lotus Development Corporation's 123. In Jay Ranade and Alan Nash (Eds.), The best of Byte: Two decades on the leading edge. New York: McGraw-Hill, 417-422.
- Williams, G. (1994b). The Apple MacIntosh computer. In Jay Ranade and Alan Nash (Eds.), The best of Byte: Two decades on the leading edge. New York: McGraw-Hill, 397-412.
- Wilson, D.K. (1986). An investigation of the properties of Kolb's Learning-Style Inventory. Leadership and Organizational Development; 7, 3-15.
- Witkin, H. (1978). Cognitive styles in personal and cultural adaptation. Clark University: Clark University Press.
- Witkin, H., & Goodenough, D. (1981). Cognitive styles: Essence and origins. Madison, CT: International Universities Press.
- Zuboff, S. (1988). In the age of the smart machine: The future of work and power. New York: Basic Books.

MONTANA STATE UNIVERSITY LIBRARIES



3 1762 10255460 5

HOUGHEN
BINDERY LTD
UTICA/OMAHA
NE.