AN ANALYSIS OF MAN-MACHINE COMMUNICATION IN AN INTERACTIVE GRAPHICAL ENVIRONMENT

2

Robert Wayne Cassell

NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

AN ANALYSIS OF MAN-MACHINE COMMUNICATION IN AN INTERACTIVE GRAPHICAL ENVIRONMENT

by

Robert Wayne Cassell, Jr.

Thesis Advisor: R. D. DeLaura

December 1971

Approved for public release; distribution unlimited.

An Analysis of Man-Machine Communication in an Interactive Graphical Environment

by

Robert Wayne Cassell, Jr.
Lieutenant, United States Navy
B.S., United States Naval Academy, 1967

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
December 1971

ABSTRACT

The failure of present-day interactive graphical systems to provide for an effective relationship between man and machine is examined. An inefficient communications link is shown to be the basis of the problem. The manner in which the user controls program flow and the hardware utilized in inputting commands are revealed as the basic components of communication.

Program control is investigated. A revised control method is described which consists of a series of function executions and six basic phases of activity for program control.

With respect to input hardware, more human factors analysis is advocated. To assist in this analysis, a function classification system is described. An example test is conducted to compare the relative merits of the light pen with menu, the teletype, and function switches.

TABLE OF CONTENTS

LIST OF DRAWINGS

I. BACKGROUND

In the past decade a new dimension has been added to the field of electronic data processing. Traditionally the computer has possessed the capability of rapid, precise execution of previously defined instructions. By combining this feature with the unique capabilities of interactive graphics a large number of new applications areas can now be addressed. Also, many of the fields which have historically made use of the computer can be augmented by the addition of an interactive graphical processing capability.

In a substantial number of disciplines the primary method of describing a given problem is a symbolic diagram. These may range from electronic circuits to flow charts. In actuality they are topological graphs which are drawn to make the problem more understandable and suggestive to a reader from the particular discipline involved. Such diagrams provide a natural medium for communication between the man and the machine. By properly utilizing this communications link interactive graphics can greatly increase the efficiency of both. The areas which fall into this category span the entire spectrum from engineering analysis and design to oceanographic wave research.

Published case studies have supported this hypothesis that interactive graphics can, if properly implemented, be very beneficial to the user. Not only can the total time

required to solve a problem be reduced, but the total cost for obtaining the solution may be reduced as well. A specific example occurred in the U. S. Military satellite program. The problem consisted of reducing and analyzing the data coming back from an orbiting space platform. Without the use of interactive graphics it required one month to get all of the calculations completed and produce a result. In addition, the total computer time used was seventeen hours. With the installation of an interactive graphical system, however, the final result was calculated in forty-eight hours with eight hours of computer time. [Ref. 1]

In spite of the virtually unlimited possibilities available in an interactive graphical environment users are, in many cases, finding other means for solving their problems. For many, batch processing with its inherent delays is preferable to going through the steps required to use present day interactive systems. A look at the history of interactive systems is required to fully comprehend this failure of graphics to achieve its full potential.

With the introduction of the computer in the early fifties engineers and scientists were provided with an extremely powerful tool to be used in their quest for knowledge. The major stumbling block on the road toward proper utilization of the computer was communication. The communication problem evolved from the fact that the computer's instructions must eventually be specified in binary form, a form which was and is foreign to the engineer

In order to utilize this new tool the engineer was forced to redesign his problem and his thinking into a form which was comprehensible to the machine's very limited vocabulary. Each step of the computation had to be reduced to such simple operations as addition and equality comparison. Some pre-coded routines such as multiplication and subtraction were developed, but their usefulness was limited.

It was at this stage of history that the computer programmer came on the scene. His job was to take over the task of recasting the desired algorithm into a form suitable for ingestion by the computer. Many times, however, the programmer was not familiar with the engineering problem at hand; the engineer was not familiar with programming and the entire process was subject to gross error. Even at its best the process was a source of great delays and frustrations for the user. [Ref. 2]

In 1957, IBM introduced FORTRAN which was developed by J. W. Backus and associates. This language enabled the user to express his problems in more readily understood terms, thereby closing the communication gap between man and machine. FORTRAN was not without its problems, however. In the first place, it was not a problem-oriented language. In addition, the details of data-management were left to the engineer, or if available, the programmer. Be this as it may, computer usage increased as engineers and scientists alike jumped at even this moderate method for communicating with the computer.

At the same time that FORTRAN was being introduced a trend was developing which provided for more and more complex operating systems. So, instead of bringing the user closer to the machine he was being pushed further and further away and was, in a sense, forced to communicate with the computer by letter as opposed to face-to-face. The only area where an interactive interface between man and machine was being actively pursued was in military command and control. An example was the SAGE system which used a display consisting of a Charactron display tube, a light gun (which was the grandfather of today's light pens) and a set of special purpose function keys. The operator could identify targets on the display for "special attention" by pointing to them with the light gun and pushing the appropriate function keys. [Ref. 3]

Early in the sixties people started to realize that a more interactive method of communication with the computer must be developed. It was at this point that interest in graphical displays and the associated input mechanisms began to grow rapidly.

In 1963, Ivan Sutherland introduced his Sketchpad system and demonstrated how effective an interactive graphical display could be for improving man-machine communication. This dissertation did more to attract attention to interactive graphics that any other single development to date.

Sutherland's motivation for developing Sketchpad was to provide a system which would do away with the requirement

of inputing analysis statements for program manipulation via the typewriter. His system effectively demonstrated that a graphic display with the appropriate programming support could provide the user with the ability to construct and manipulate drawing-type information. This was a great leap forward because at last the engineers and scientists would be allowed to converse with the machine in the form most natural to them, i.e. graphics. To allow for ease of execution by the user, Sketchpad contained extensive console support. Facilities were provided for creating, duplicating, and manipulating graphical figures with no prior programming required of the user. This characteristic of systems programming being performed for the user is vital to an effective interactive graphical system. [Ref. 4]

Following the introduction of Sketchpad, an increasing number of similar systems were developed. Noteworthy amoung these were Sketchpad II1 which was a three dimensional version of Sutherland's system and the DAC-1. The DAC-1 was introduced in 1964 at the Fall Joint Computer Conference in San Francisco, California. It was an interactive graphical system developed by General Motors Corporation in which the operator created and modified the designs of automobile parts. The sketches which were created were stored in memory and were converted into wooden models with the aid of software programs for controlling the appropriate tools. An important feature of the DAC-1 was the incorporation of a significantly large data base from which the user

could call out previously defined designs in order to modify them or incorporate their features into future designs. It is important to note that the facilities for creating and modifying the drawings were pre-programed for use by the operator. [Ref. 5]

From the middle sixties to the present little has been done to improve upon the original concepts of interactive graphical systems. Many systems have been developed, but invariably they fall short of their ultimate goal. That is, they fail to provide for truly efficient and effective man-machine interaction. As recently as 1969, Edward Streit attempted to develop his own system for interactive graphics stating in part:

> "The original motivation for the development of this system was supplied by the author's ever-increasing dissatisfaction with currently offered cathode-ray display software." [Ref. 6]

In the same year Mr. William H. Ninke stated:

> "Initially, I think we saw having a display on a computer as solving all our design problems. All we needed was the equipment. We soon discovered the shallowness of this attitude, and, having recognized that equipment alone wouldn't solve problems, we began in earnest to lay out a plan of program development....." [Ref. 7]

From the preceding discussion, it can be seen that since the computer's inception the main problem has been one of communication. Any major improvement of the interactive graphical system will require examination of the man and the machine not as separate entities but as the basic components of one system with a dire need for effective communication.

The computer's particular attributes include speed, memory and reliability. Once the calculations to be performed have been specified in detail it is capable of performing them at an astounding rate of speed. Its memory is capable of storing large amounts of data and it may be accessed in a very reliable fashion.

Man, on the other hand, is not good at computation. He is slow, he makes mistakes, and he has a poor memory for figures. He does, however, bring some indispensable features into the relationship. He has experience, imagination, social and esthetic values, an exceptional capacity for pattern recognition, and a tolerance of ambiguity. In addition, he can create, innovate, and be inventive.

In brief, "man is a nonlinear, discrete, adaptive, time-varying statistical controller with a mind of his own" [Ref. 8] while the computer is a "lightning fast, nit-picking, myopic idiot." [Ref. 9]

The critical feature affecting the man, the computer, and the combined operation of the two is the adequacy of the communications link between them. Obviously the creation of a satisfactory interface between such radically different components is a very demanding problem. In fact, despite all of the work which has been done between the development of Sketchpad and the present-day systems the man-to-computer communications link remains the weakest link in the interactive graphical system.

An analysis of man-machine communications reveals two basic components. The first of these is the physical means with which the user inputs his messages to the computer. This includes such things as the teletype, the light pen, and function switches. The second component is comprised of the actual messages with which the user controls the flow of the analysis by the computer. At the most basic level, these messages consist of machine language instructions which cause the manipulation of and operation upon the various registers of the computer. At a higher level, a language such as FORTRAN is used and the user may utilize arithmetic assignment statements and general control options such as the conditional statements. These statements are then translated into machine language prior to execution by the machine. It should be noted that even at this higher level the user is forced to communicate via single statements.

It is the author's contention that these basic components of communication have not been carefully analyzed and implemented in current interactive graphical systems. Consequently, the communication between man and machine is awkward and in many cases non-productive. It is the purpose of this thesis to examine the basic components of the communication process with the emphasis placed upon the man as opposed to the machine.

II. CONTROLLING PROGRAM FLOW

A. GENERAL

When discussing program flow in an interactive graphical environment, the traditional concepts are no longer valid. In the typical "off-line: application, the flow follows the linear structure depicted in Figure 1. The program (which has been previously defined by the user) reads in data, operates on it, and produces what sometimes amounts to volumes of output. If a variation of data is desired or if a change in the general flow of the program is necessary, the program must be changed accordingly and re-executed. This type of operation is sufficient for the mathematician looking for the square root of two to the millionth decimal place, but it is a constant source of frustration for the engineer who is interested in studying the reactions of a particular circuit when various inputs are applied.

With an interactive graphical system there are at any point in time a number of directions in which program flow can proceed. Because of this, a system of control is proposed in which the user is permitted to move between the various stages of program execution at will. Inputting data, analyzing results, and performing calculations are examples of these stages and are known as basic execution blocks. This more complicated version of program flow is shown in Figure 2. Within the basic execution blocks, the program must wait for

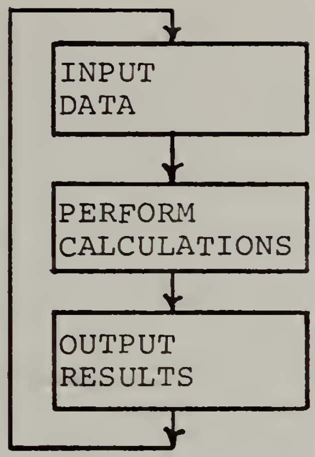


Figure 1. Program Flow in an off-line environment

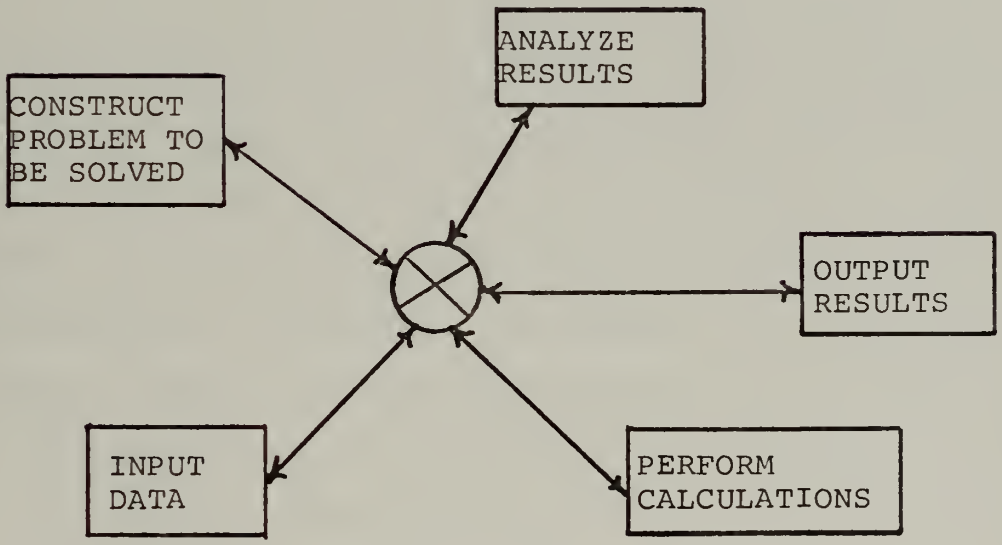


Figure 2. Flow in an interactive graphical system.

directions from the user at the completion of each command. When the program has responded to the action dictated by the user, it waits for another command indicating the direction in which analysis is to proceed. The process continues in this manner until the termination of execution by the user.

The problem with this version of controlling program flow is in determining how big to make the step the computer takes before pausing. Obviously you don't want the program to idle following the completion of each machine instruction. This would be awkward and frustrating for the user. In addition, you do not want the traditional concept of having the program pause after it has done all of the calculations for a particular data set. This is also frustrating since the user possesses too little control over the execution. Instead, program flow is best represented by a series of function executions. The user indicates which function he would like executed, the computer complies and so on. It is felt that by working on the level of function execution naturalness is achieved. For example, if a user desires to rotate a figure by forty-five degrees he visualizes the figure rotating by forty-five degrees. He does not imagine the particular programming instructions that would cause this rotation to occur.

As used here, the term function describes any sequence of instructions which, when executed, cause a specific event to occur. For example, the hypothetical function "Grid" would cause a grid to appear on the face of the CRT when

executed. The instructions that go to make up the function may be in two possible forms. They may be basic machine instructions or they may be hardware implemented. In addition, the execution of one function may require the execution of any number of other functions to achieve the desired result. Finally, a function may have any number of parameters from zero to infinity. The parameters are used to define the value of specific variables within the function that cannot be specified until execution time under normal circumstances. An example of this occurs in the function "Rotate" where the number of degrees to be rotated is specified at execution time.

Because of the wide range of applications that can be encountered in the interactive graphical environment, there are many different functions that can be met. For example, the functions required to analyze the molecular structure of a compound in chemistry are a good deal different from the circuit analysis functions required in electrical engineering. In addition, the chemist would not be interested in having the electrical engineering functions available to him and vice versa. Because of this, each particular discipline would have its own set of functions to use when conversing with the machine. This unique set of functions would be known as the discipline set. Each of the sets however, would be based upon a subset of instructions which all disciplines have in common. All discipline sets, for example would require functions enabling the user to display

results, draw a line, and identify a particular point. Functions of this type would be referred to as basic functions. It would seem appropriate to maintain a library of basic functions so that the programmer for a particular discipline need only supplement them with the functions which are unique to his area in order to obtain the required discipline set. This relationship between two discipline sets is illustrated by the Venn diagram in Figure 3. The intersection of the two sets is the set of basic functions while the disjoint portions of the diagram represent the functions associated with the individual disciplines.

It is important at this point to distinguish between the programmer and the analyst as first conceived by Corbin and Frank in 1967. [Ref. 10] The analyst is the ultimate user for whom the hardware and software exists. He is the problem solver and consequently should not have to concern himself with the details of data processing hardware. The analyst operates at a level which is above the world of

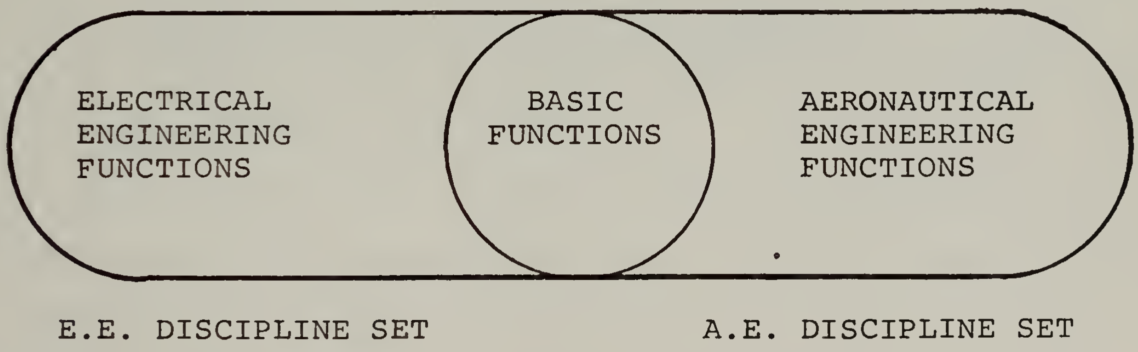


Figure 3. The relationship between two discipline sets.

register transfers and machine cycles. His concern is with application and manipulation oriented functions. Application functions include such functions as would be required to determine the total resistance of a circuit or the stress on a beam. Manipulation functions, on the other hand, would perform such operations as rotating or translating an object.

The programmer is responsible for preparing the various functions making up the discipline set of the area in which he is working. As such he is very much concerned with the hardware and the use of the basic computer language in which the functions are to be written.

B. A METHOD FOR PROBLEM SOLUTION

For all of the applications found in the interactive graphical environment there is a generalized method for problem solution. This method describes six basic phases of activity which an analyst engages in during the process of solving a particular problem. A diagram of the method is shown in Figure 4. The six phases are:

A. Selecting primitives
B. Defining any new primitives required
C. Constructing a model of the problem
D. Selecting the analysis functions to be applied
E. Defining any new analysis functions required
F. Outputting results

Phase A covers the process of selecting primitives. This consists of choosing the basic building blocks for the model of the program to be constructed. In the case of the electrical engineer, for example, typical primitives

would be capacitors, resistors, and voltage regulators. In most cases the primitives required for any particular problem would be pre-defined functions contained in the analyst's discipline set. If, however, an analyst was working with a device which had not been previously defined, he would set forth the definition for this new primitive in Phase B.

There are three methods for defining new primitives. These are analytic definition, empirical definition, and macro definition. The analyst can supply some symbol for the primitive he wishes to define by constructing the symbol graphically. He may then define the nature of the primitive using one of the three methods mentioned above.

The analytic method consists of inputting an explicit, well-defined, analytic expression in some previously agreed upon notation such as an arithmetic assignment statement.

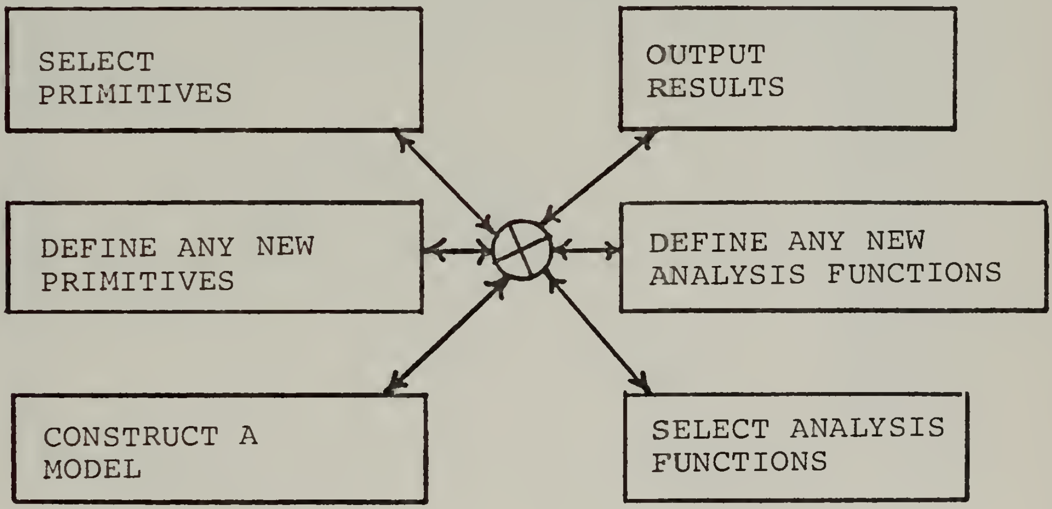
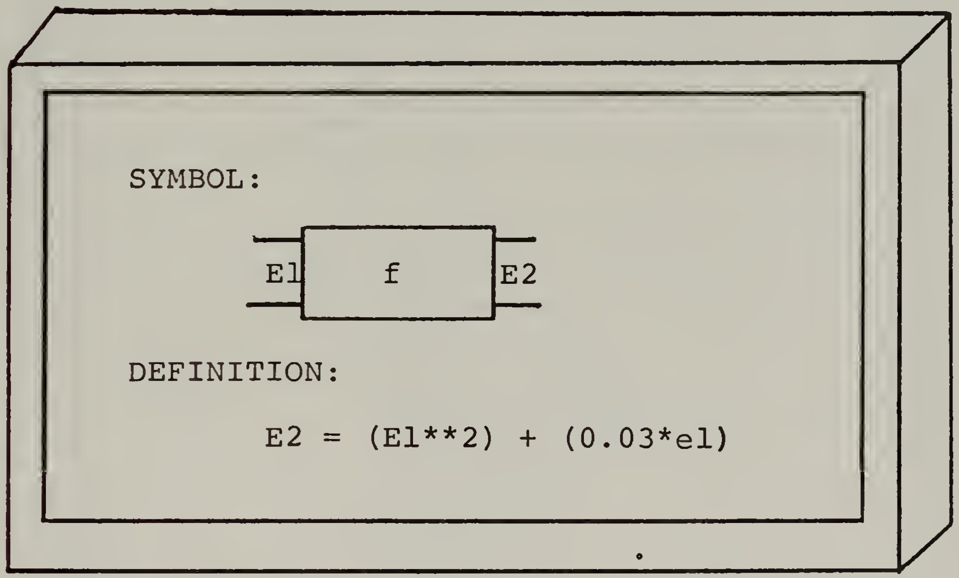

SELECT PRIMITIVES
OUTPUT RESULTS
DEFINE ANY NEW PRIMITIVES
DEFINE ANY NEW ANALYSIS FUNCTIONS
CONSTRUCT A MODEL
SELECT ANALYSIS FUNCTIONS

Figure 4. Illustration of the generalized method for problem solution.

It is important to note that no experience in or knowledge of computer programming is required on the part of the analyst for defining primitives in this manner. An example of analytic definition is shown in Figure 5.

In many cases the analyst may wish to define a primitive in terms of experimental data or in some way other that an arithmetic assignment statement. In this instance, he would use the empirical definition. This consists of inputting a data set in some pre-determined ordering or simply supplying a hand drawn curve for analysis by the computer. An example is shown in Figure 6.

The final method of definition is through the use of the macro. This is most useful in those cases where a new primitive is actually a combination of previously defined primitives. In this instance the analyst would construct a



SYMBOL:

DEFINITION:

E2 = (E1**2) + (0.03*el)

Figure 5. An example of analytic definition.

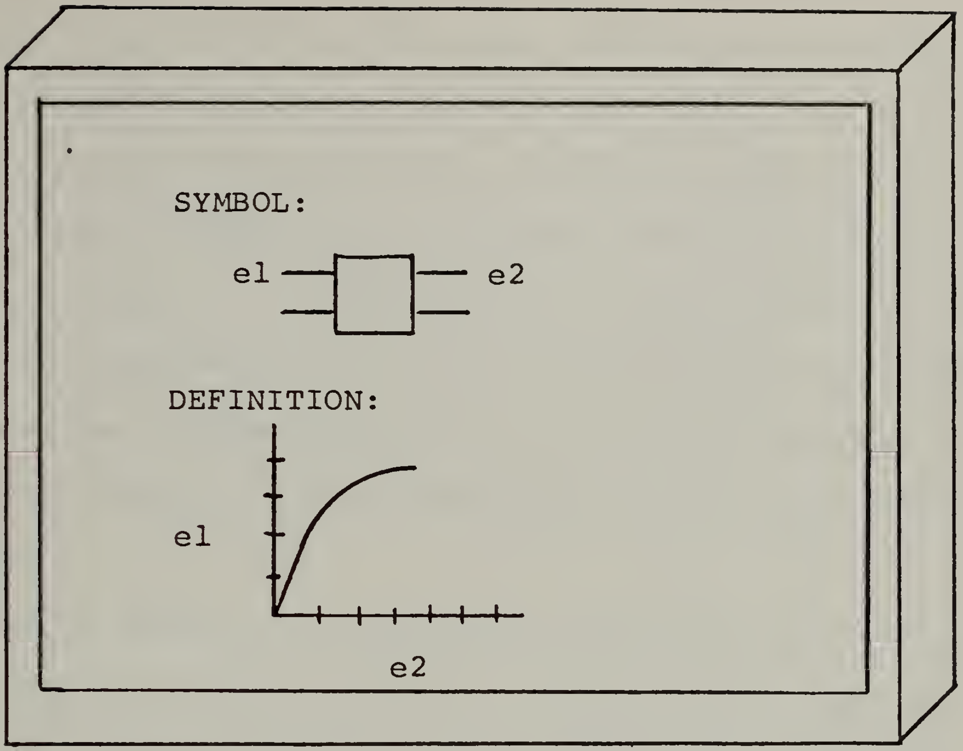


Figure 6. An example of empirical definition.

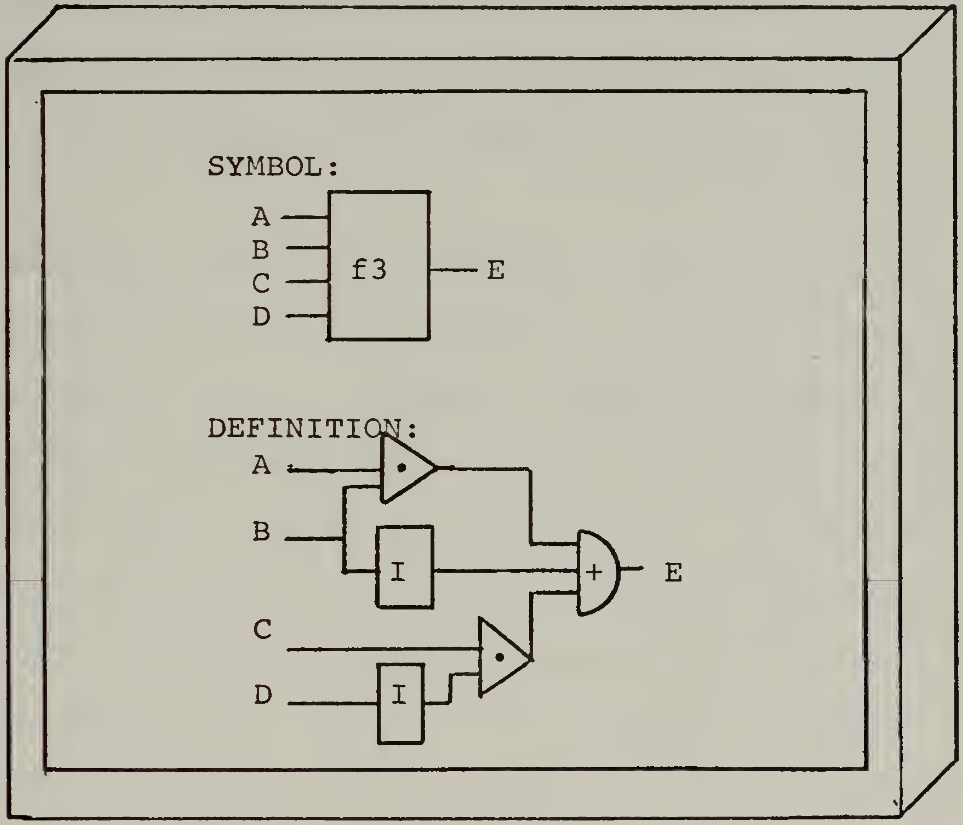


Figure 7. An example of macro definition.

network of primitives in the proper configuration utilizing the facilities available in the modeling phase. He would then equate this new network to a primitive whose symbol he has defined graphically. An example of this procedure is shown in Figure 7. The definition of primitives in this phase would necessitate the use of basic functions providing the capability to draw primitives, name them, file and retrieve previously defined primitives, and connect existing primitives.

Having defined all of the primitives which he expects to utilize, the analyst proceeds to use these primitives to construct a symbolic diagram representing his formulation of the problem to be analyzed. It is this process of constructing a symbolic diagram that is known as the modeling phase. In the case of an electrical engineer, this phase would consist of drawing the dircuit diagram of the circuit in question.

The modeling phase would necessarily contain a set of basic functions which could be used to create, modify, store, retrieve, name, and combine symbolic diagrams. The basic drawing facilities such as draw curve, draw straight line, erase, delete, move, and copy would be included in this set of functions. In addition, the basic functions enabling the analyst to enter alphanumeric data which would be associated with any instance of a primitive must be included in this phase. For example, the analyst could select a resistor symbol as a primitive and use it several times

in the construction of a circuit diagram. After completing the drawing, he would go back to each occurance of the resistor and enter the resistance for that primitive. This can be accomplished by selecting the particular instance and associating it with the set of alphanumeric data which is entered immediately thereafter. It should be noted that the inclusion of this feature would necessitate a sufficient data structure to be able to identify the separate occurances of the primitives.

After completing the modeling phase the analyst is ready to select the specific analysis functions which he would like to have applied to his model. The functions used here would actually be a collection of programs representing the most widely used numerical analysis and problem solving techniques. Functions for simulation (both fixed time step and next event models), a relaxation function, an integration function, and an algebraic equation solver would be typical. In addition, the analyst would be required to select and adjust the manner of execution of any particular analysis technique. Some examples of such choices and controls are: The interval of integration, the period of time to be simulated, and the number of iterations. Typically, this type of information would be entered through an interactive exchange between the computer and the analyst with default conditions available if the analyst has no particular choice.

The analyst also has the option of continuing on to Phase D which consists of defining any new analysis functions which he may require. This phase is required since it would be impossible to predict all of the required analysis techniques for any given application. In some respects, this phase can be considered as a programming activity, but it should be noted that this phase would not normally be required since most analysis functions would be previously defined. In addition, the language used for specifying the new function would be as high-level as possible in order to maintain the naturalness of the input. At the very least, the specification would require the input of FORTRAN-like statements on the part of the analyst.

The final phase of the analysis method is the output phase. In this phase, the results of the analysis phase operations on the model are fed back to the analyst. Although this phase is conceptually the simplest, it has the potential of being the most extensive. The subject of data presentation is a field in itself and can become extremely elaborate. Functions to display two dimensional plots of variables, families of curves, and multiple simultaneous plots are typical of the types of functions encountered in this phase. Once more the user would indicate the variables he desired plotted, scale factors, etc., in an interactive mode with defaults available.

The implementation of the basic facilities described here would greatly increase the capabilities of the analyst

during the period in which he is forming and evaluating conceptual ideas. It would allow him to deal with the concepts of his particular discipline through "natural" communication with and utilization of the computer. In addition, the use of the six analysis phases would not require any excessive learning on his part thereby minimizing any distraction from his work. Also, it is inherently possible for the analyst to make mistakes in the use of the system without the loss of any significant amount of time other than the amount involved in correcting the error. To an extent, this is accomplished by the division of the process into distinct phases. It is supported, however, by the fact that execution may be started or stopped at any time at the discretion of the analyst. In addition, he may move between phases at will. As an example, suppose that an electrical engineer realized in the modeling phase that he had failed to specify a capacitor as a primitive. At this point, he need only revert to the phase for selecting primitives, call out the capacitor, and continue back in the modeling phase.

III. INPUT HARDWARE

A. GENERAL

In the past, the choice of a particular input device to be used by the analyst has been largely a matter of applying the "I think this better" hypothesis by the system designer. As an example, when Sutherland was designing Sketchpad, he was disillusioned with the teletype as an input device. Consequently, he utilized the light pen with menu to a large degree in his system. [Ref. 11] This decision was apparently based upon personal preference as opposed to the application of human factors techniques to determine which of a variety of available devices was preferable. This type of approach can and has led to the situation where the analyst is frustrated in his efforts to control a possibly superior software system because he has been provided with inadequate control hardware. In 1966 Nilo Lindgren stated in part:

> "The problems of "man-computer symbiosis" as defined by Licklider are the result of an inadequate physical interface. That is, the console with its displays and controls. [Ref. 12]

At the present time, man's inputs to the computer in an interactive graphical system tend to be limited to such devices as the teletype, the function selection box with push buttons, and the light pen. Recently, however, more intricate hardware has been added to the inventory of available devices. The following is a description of some of the more interesting.

The Rand Tablet is so called because it was originally developed by the Rand Corporation of Santa Monica, California. It consists of an array of X and Y electrical conductors arranged just below the top surface of the writing area. Each line is fed by a distinct ten bit serial pulse train generated by capacitive encoders. A stylus containing a high-gain amplifier is positioned over the working surface and detects the pulse trains from the nearest X and Y lines thereby fixing the location the stylus.

The track-ball can produce either an analog or digital output. By rolling it in any direction, the operator can drive a cursor on a display.

The joy-stick, which is a descendent of the control device used in aircraft, supplies outputs similar to the track-ball. In addition, it can be designed to provide an output proportional to either the force applied to the joy-stick or its deflection

The SRI Mouse which was developed by the Stanford Research Institute is constructed of two potentiometers, mounted orthogonally, each of which has a wheel attached to its shaft. As the case is moved over a surface, the motion of the wheels on the surface will generate an output proportional to the X and Y motion of the device. [Ref. 13]

It is interesting to note that few, if any, of the aforementioned devices have the associated data on operating characteristics required to properly perform a human-engineering evaluation during system development. [Ref. 14]

There appear to be two prevailing philosophies concerning the inadequacy of the control hardware. The first seems to advocate waiting for the perfection of more exotic devices such as automatic speech recognition and interpretation hardware and the problem will cease to exist. The second philosophy acknowledges the seriousness and the immediacy of the problem but believes that improved man-machine communication evaluation requires new knowledge in the field of human factors engineering. [Ref. 15]

With respect to the need for advances in the field of human factors engineering, the truth is that while new knowledge would be of value there is sufficient knowledge at present to radically improve the man-machine interface. We are not in the habit of applying current know-how in the design of man-machine interfaces because we are not even applying our knowledge to devices which we encounter every day in other walks of like. Our road signs, traffic signals, schedule boards in airports, typewriters, books, and magazines, in fact, nearly all methods of presenting information for human consumption exhibit an almost total lack of respect for simple, well-known principles of human factors engineering. This is due mainly to the fact that human beings, once they construct and adjust to a machine, no matter how cumbersome, resist having it changed even though the change would mean improvement in the function or ease of use. [Ref. 16]

This failure to apply basic human factors engineering principles is evidenced by most of the interactive graphical systems in use today. For example, systems engineers will go to great lengths to design and implement a more effective data structure. In the process, they will often do extensive simulation and validation testing to insure that the data structure will be efficient and appropriate for the application involved. In comparison, the hardware devices used by the human to input the information to be stored in the data structure receive little or no consideration. The point here is that while it is important to analyze the operations to be performed by the machine, it is equally important to analyze the human side of the system. The human must be given equal consideration through the application of established techniques for human factors analysis. In designing the computer side of the system, consideration is given to the computer's basic characteristics such as the best operating temperature, the maximum line length, the delay time required, and the size of its memory. So, when designing the devices to be used by the human side of the system, it is only appropriate that equal consideration be given to the man's basic attributes--his sensory capabilities, his muscular strength, his perception, his native skills, and his optimum work load. [Ref. 17]

With regard to the philosophy that is is best to wait until more sophisticated hardware devices are developed, the fact is that the state-of-the-art devices are not being used

at their maximum efficiency. More research is required to fix the environment in which the various mechanisms at the disposal of the system designer are best suited before such a rash statement of inadequacy can be made. Without being fully aware of the capabilities that currently exist, it is virtually impossible to state categorically that present-day equipment is not sufficient. Once more a lack of research and the application of basic human factors engineering principles is responsible.

B. FUNCTION CLASSIFICATION

When analyzing an input device for use in the interactive graphical environment, it can be noted that it will have two possible tasks. These tasks are to initiate a function or to provide the parameters for a function which has been initiated. A further analysis of functions reveals that their compatability with an interactive system is dependent to a large extent on the number of inputs which must be supplied by the analyst in order to completely define the parameters associated with the particular function involved. If, for example, a function required seven inputs, this would certainly be more clumsy for the user than one requiring only one or two inputs. Along this line, it is useful to classify functions according to this number of inputs required. This number will be an integer from zero to infinity. A function which requires two parameters to be completely defined would be a Class two

function. As an example, consider the hypothetical function "Dot" which would place a dot at a specific location when executed. This would be a Class two function since two parameters (an X and a Y coordinate) would be required to uniquely identify the location of the dot. The function "Line," on the other hand, would be a Class four function since the X and Y coordinates of both end points would be necessary to fully define the entity.

This method of function classification is of value to the system engineer because it gives him a feel for the amount of information which an analyst would be required to supply when executing a particular function. The higher the class of the function, the more information required and consequently the more action demanded of the analyst.

It is worthwhile to note that the hardware device used to supply the required inputs can change the effective class of the function. In the case of the "Dot" function, for example, if a teletype was being used to input the information it would remain a Class two function since the analyst would be required to input two coordinates. If, however, a joy-stick was being utilized to indicate the position at which the dot was to be placed, the function would effectively become a Class one since only one input (the proper positioning of the joy-stick) would be required of the analyst. This ability to lower the effective class of a function is one of the reasons for the popularity of such input devices as the SRI Mouse and the joy-stick. This

is not to infer, however, that this is the only characteristic worth considering. Other characteristics such as naturalness of use, speed, and error rate must also be taken into consideration when choosing between several available devices.

C. AN EXAMPLE TEST

1. General

As an example of the type of research that should be performed when choosing an input device for use in an interactive graphical system, tests were designed and conducted to compare the relative merits of the light pen with menu, the teletype, and function switches. These particular devices were used since they represent the most commonly used hardware for selection of a function, regardless of its Class. The tests were conducted in a manner designed to simulate the general situation facing an analyst who must select one of a number of available functions. In addition to judging the ability of the three devices in this situation, the relative merits of displaying a large number of functions at once, using a limited number of functions, and displaying a large number of functions with only a limited number shown at any instant were analyzed.

The experiment consisted of seven separate phases of activity. For each phase, the total time required to complete the phase and the total number of errors made were recorded. The participants were taken through the various phases in random order to avoid any possible bias.

is not to infer, however, that this is the only characteristic worth considering. Other characteristics such as naturalness of use, speed, and error rate must also be taken into consideration when choosing between several available devices.

C. AN EXAMPLE TEST

1. General

As an example of the type of research that should be performed when choosing an input device for use in an interactive graphical system, tests were designed and conducted to compare the relative merits of the light pen with menu, the teletype, and function switches. These particular devices were used since they represent the most commonly used hardware for selection of a function, regardless of its Class. The tests were conducted in a manner designed to simulate the general situation facing an analyst who must select one of a number of available functions. In addition to judging the ability of the three devices in this situation, the relative merits of displaying a large number of functions at once, using a limited number of functions, and displaying a large number of functions with only a limited number shown at any instant were analyzed.

The experiment consisted of seven separate phases of activity. For each phase, the total time required to complete the phase and the total number of errors made were recorded. The participants were taken through the various phases in random order to avoid any possible bias.

All of the subjects tested had some prior knowledge of the hardware being used. In addition, a presentation explaining the format of the experiment and a review of the hardware operation was given to the entire group of participants prior to the individual testing.

Phases one and four were conducted to simulate the case where a large number of functions (twenty-eight) are available to the analyst and all are displayed simultaneously. In Phase one, the functions were displayed on the face of the CRT and individual selections were made with the light pen. Phase four consisted of selecting the desired function by pushing the proper function switch.

In Phases two and five the situation was simulated in which a large number of functions are in use; however, only a limited number (seven) contained on one of four "pages" are displayed at any given instant. These phases were conducted to test the hypothesis that an analyst would be able to function more efficiently when confronted by a small number of functions at any given instant as opposed to displaying them all at once. For Phase two, the functions were displayed on the CRT one page at a time. In Phase five, the functions were listed on four templates and the user was required to place the proper template over the function switches and press the required switch.

Phases three and six consisted of utilizing only a limited number of functions (seven) which were displayed at one time. This was done to test the assertion that the

analyst would, when confronted by a limited number of choices, become familiar with them more rapidly and make fewer errors with greater speed than in the case of a large number of functions.

For Phase seven, the situation in which the user must type in the names of the functions required via the teletype was simulated. This was done in an effort to compare a device requiring some amount of skill to those requiring very little.

2. Description of the Hardware Used

The teletype used in the experiment was a Teletype Corporation Model 33 teletypewriter. This is, basically, an electromechanical input device which is used to interface with the Adage AGT/10 graphical display unit.

Because a function box with a sufficient number of switches was not available, a wooden model was constructed for use in the evaluation. The box consisted of twenty-eight function switches arranged in four rows of seven each. In addition, each of the switches was connected to a separate light to indicate activation of that switch. To select a particular function, the button was depressed and the indicator associated with that switch would light. When released, the button would return to the original position automatically and the light would go out.

The light pen which was utilized was a hand held pen with a fingertip switch on the barrel for making a selection. The selection was made from a list or menu of function names

which were displayed on the CRT of the AGT/10. The names were composed of characters which were 0.30 inches high and 0.20 inches wide.

3. The Experiment

The various phases of the experiment were conducted as follows:

a. Phase 1

The subject was given a list of 15 functions to be selected. He then picked up the light pen and proceeded to pick the proper function from a menu containing 28 function names. The menu was displayed on the CRT as shown in Figure 8. A selection was indicated by having the name of the function displayed below the menu. An example of this is illustrated in Figure 9 with the function "Point" selected. After selecting a function, the subject placed the light pen on the table in front of him and proceeded to read the name of the next function to be chosen. He then picked up the light pen to select the function as above. This process was continued until all 15 functions had been completed.

b. Phase 2

In this phase, the subject was given a list of 15 functions to be selected as before. After reading the name of the first function to be selected the page on which the function was listed had to be located. This was accomplished by moving from page to page with the aid of the light pen and menu. To select page two, for example,

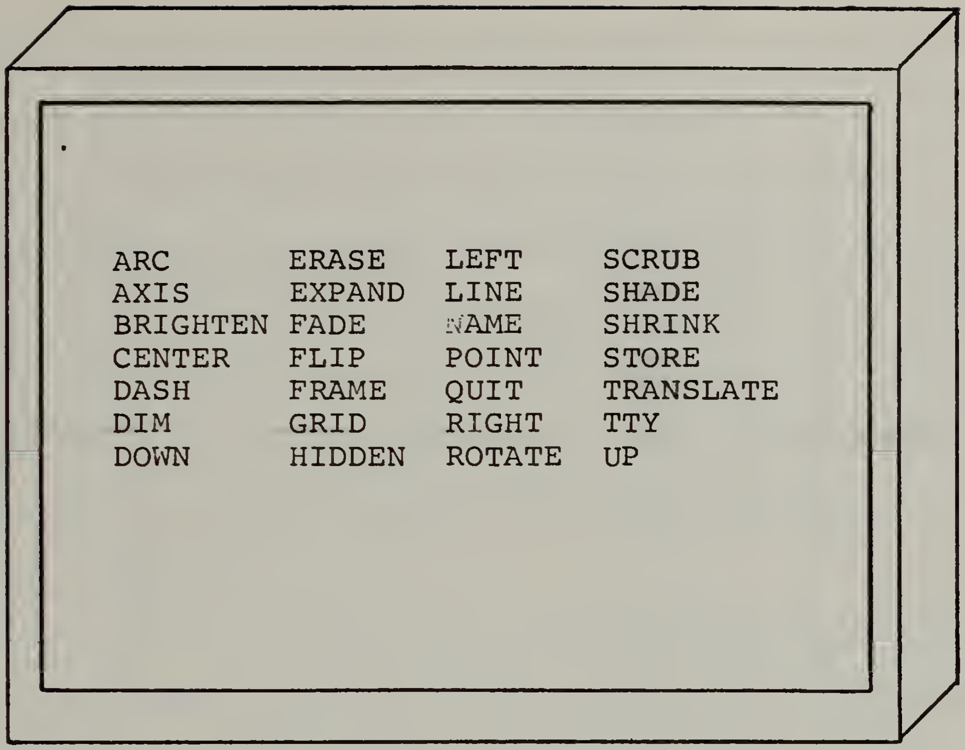


Figure 8. Large number of functions on one "menu".

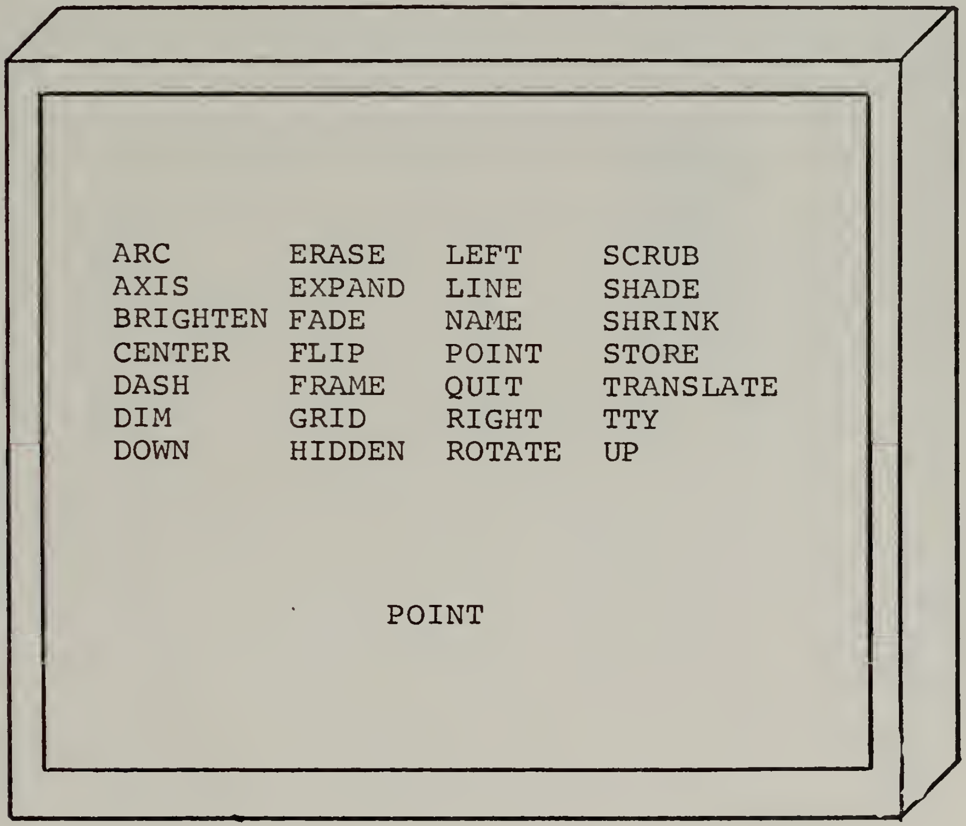


Figure 9. The function Point has been selected.

the subject would select "TWO" from a list at the bottom of the menu which contained the various page numbers. This is illustrated in Figure 10. After finding the proper page, the subject would select the required functions from that page as in Phase one. Following this, the name of the next function to be selected was read from the list and testing proceeded as indicated above. This process continued until all of the functions had been selected.

c. Phase 3

In this phase, only seven functions were displayed on the CRT. Other than this, testing was identical to that described in Phase one.

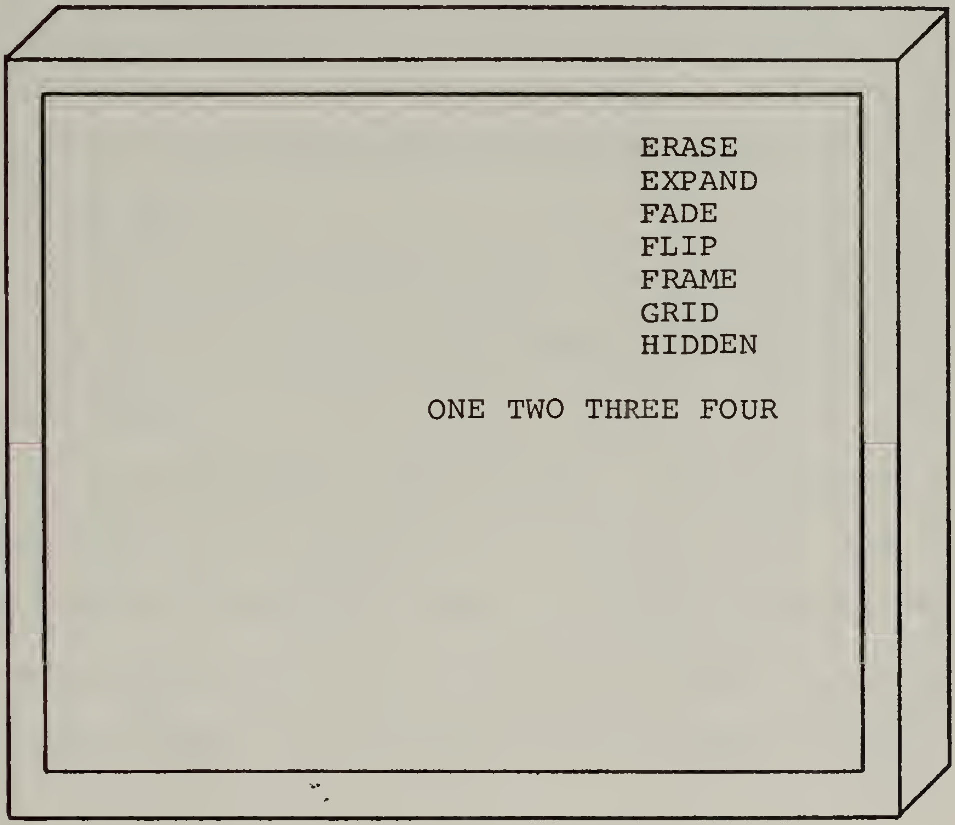


Figure 10. A multi-paged display with page two selected

d. Phase 4

This phase was similar in all respects to Phase one with the following exceptions. Functions were selected by pushing the proper button on the function box as opposed to selecting them from a menu. The particular function selected was indicated by the light which was energized. The function box is shown in Figure 11.

e. Phase 5

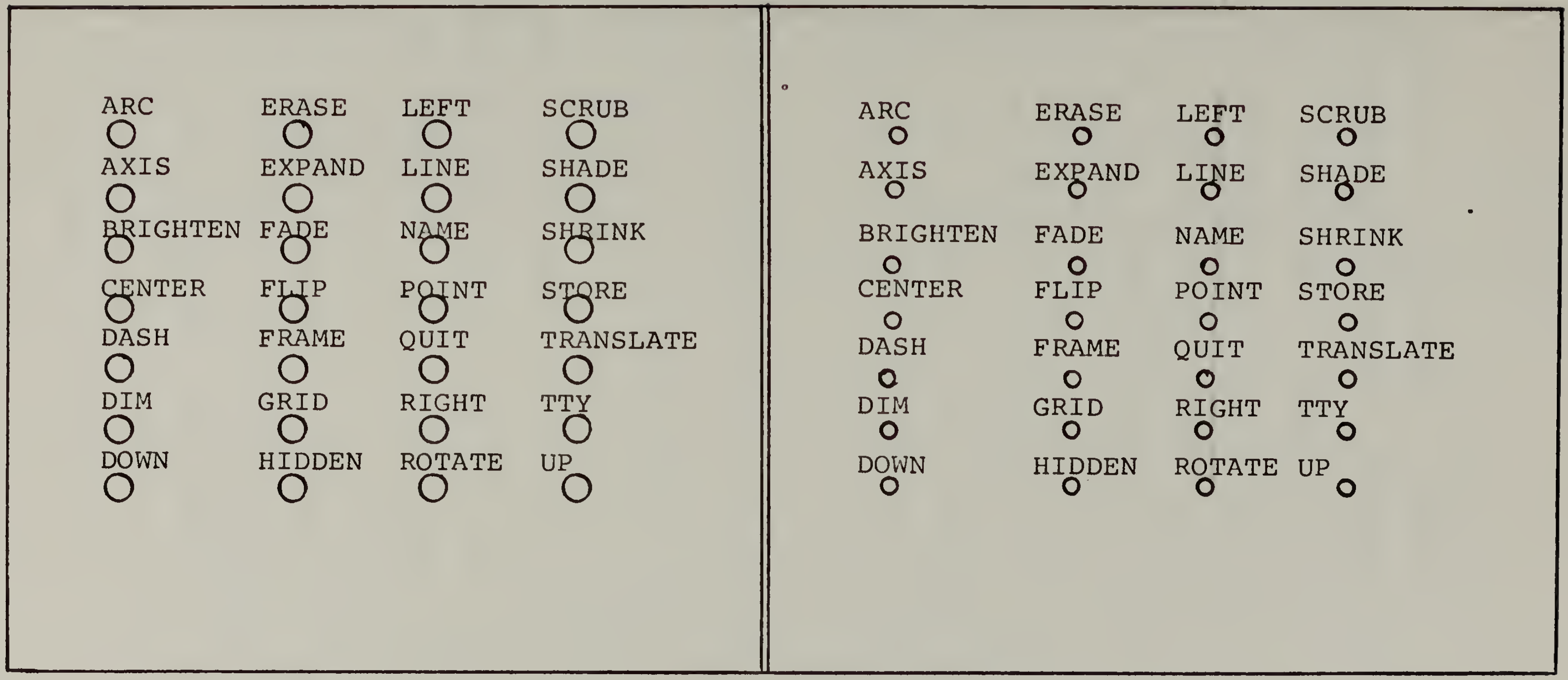
Instead of having the functions on four pages which were selected with the light pen as in Phase two, the subject was given a series of four templates with seven function names per template. To select a particular function he was required to locate the proper template, place it over the function keys, and push the button. Once more, the function selected was indicated by a light. With the above exceptions, the test proceeded as in Phase two.

f. Phase 6

This phase modeled Phase three with function switches taking the place of the light pen and menu.

g. Phase 7

In this phase, the subject was required to input the list of 15 functions to be selected via the teletype. After reading the name of the function he typed it in and followed this with a carriage return. This process continued until all of the function names had been typed in.



Function switches

Function selected indicators

Figure 11. Function box and associated light to indicate function selected

4. Test Results

The actual results obtained from the experiment are contained in Appendix A. Figure 12 compares the mean total time required to select functions from a limited number of alternatives with no penalty for error. It should be noted that while the teletype is definitely unfavorably rated in this case, the function switches and menu are close enough to require further comparison of the two before choosing one over the other. Such things as ease of use and naturalness would be considered in this testing. Figure 13 depicts the total number of errors committed for each device. Once more the teletype is definitely undesirable while the function switches and menu remain undecided.

In order to obtain a representation of the time required to perform a task taking error correction into account, the adjusted selection time was calculated. This was done by first dividing each subject's total selection time by the number of functions selected (fifteen). This yielded the average time for each function selection for that subject. This time was then multiplied by the number of mistakes encountered to get the total correction time. The total correction time was then added to the subject's selection time to obtain the adjusted selection time. As an example, consider the case where the subject selected the fifteen functions in thirty seconds with three errors. The average function selection time is thirty divided by fifteen or two seconds. The correction time is two seconds

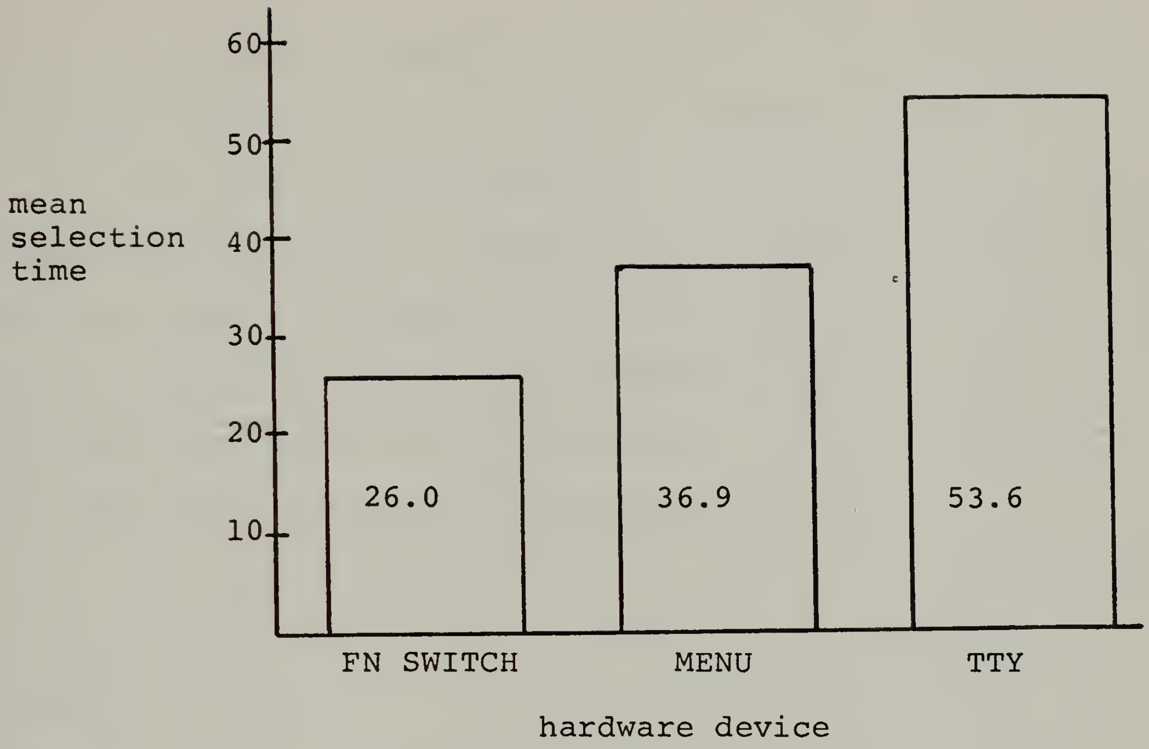


Figure 12. Mean selection time for a limited function set

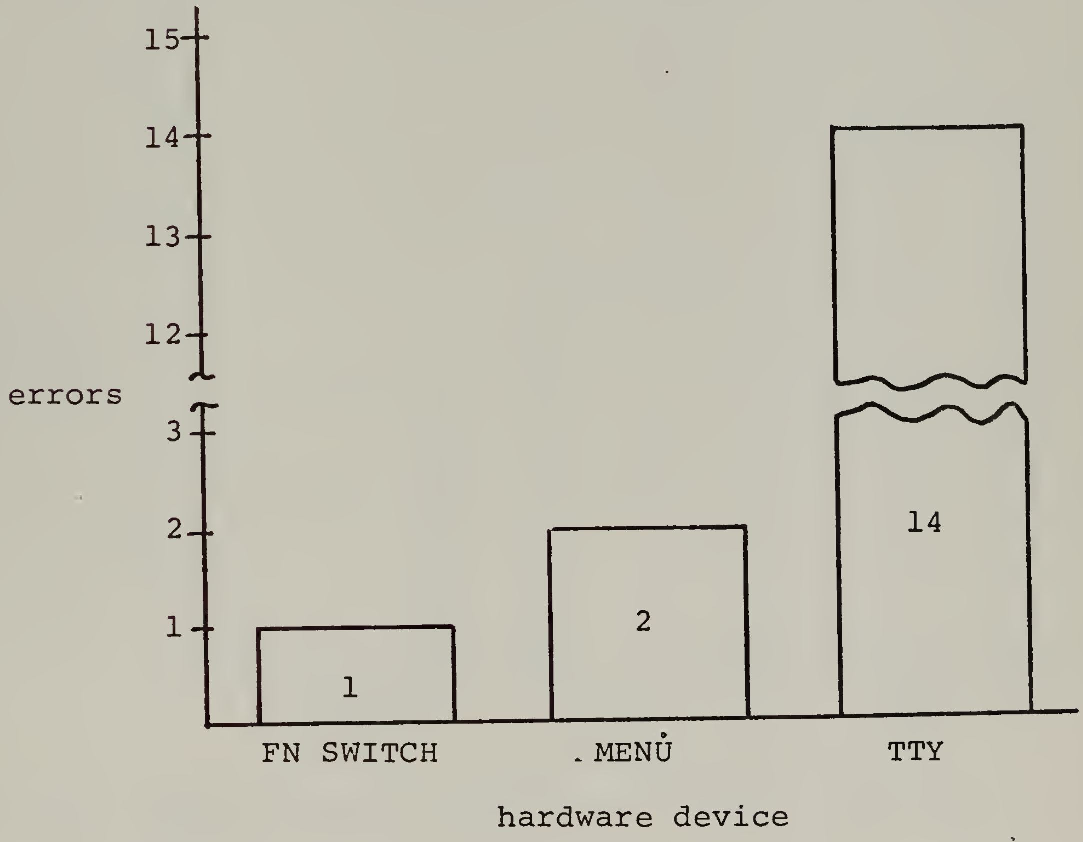


Figure 13. Total number of errors for each device

multiplied by three errors or six seconds. So, the adjusted selection time in this case would be thirty-six seconds.

In Figure 14, the adjusted selection time for each device is illustrated. Although the difference between the function switch and menu increased somewhat in this case there is still an insufficient separation between the two to justify choosing one over the other. The teletype, on the other hand, remains an unsatisfactory choice.

The mean selection times for the various devices with a large number of functions displayed simultaneously are compared in Figure 15. In this case all three devices are closely related and selecting one in particular on this basis alone would not be valid. In figure 16, the total

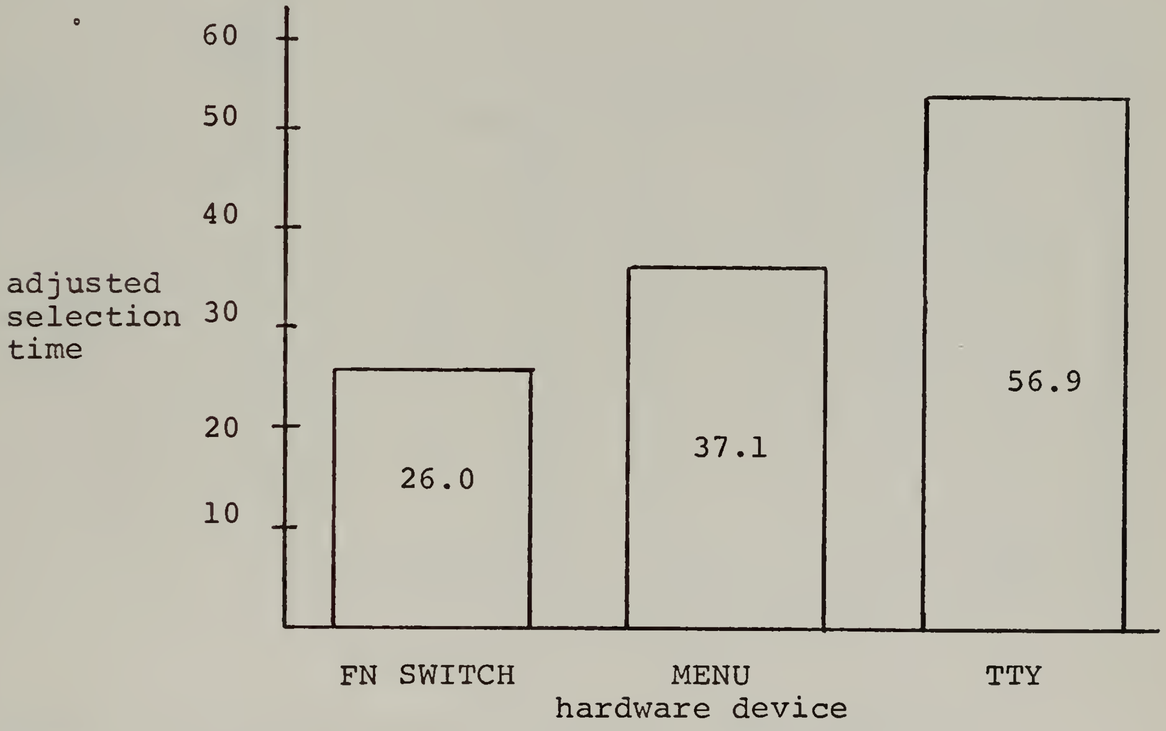


Figure 14. Mean adjusted selection time for a limited number of functions

number of errors committed in this situation is shown for each device. The error rates for function switches and the menu are comparable while the teletype is unsatisfactory in comparison. An analysis of the adjusted selection times illustrated in Figure 17 supports the hypothesis that the devices are comparable in this instance in spite of the error rates and further testing would be required in order to decide which was preferable.

The mean selection times for each device when a large number of functions are paged is illustrated in Figure 18.

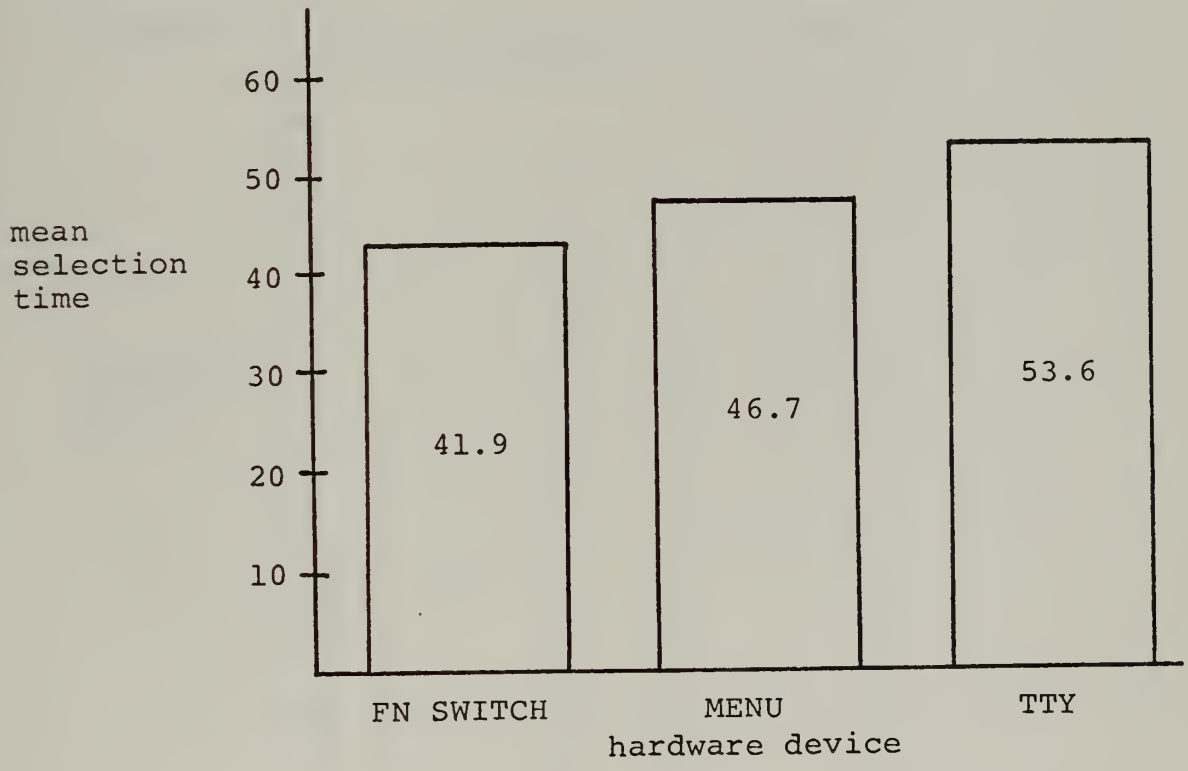


Figure 15. Mean selection time for large number of functions displayed simultaneously

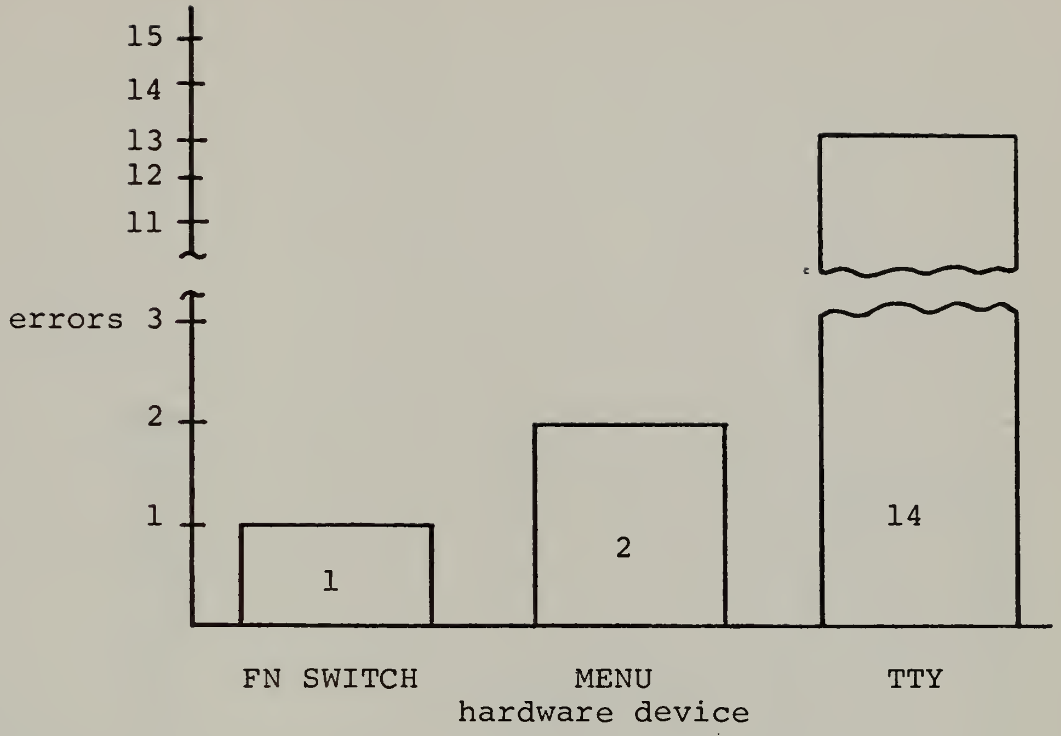


Figure 16. Total number of errors committed for each hardware device

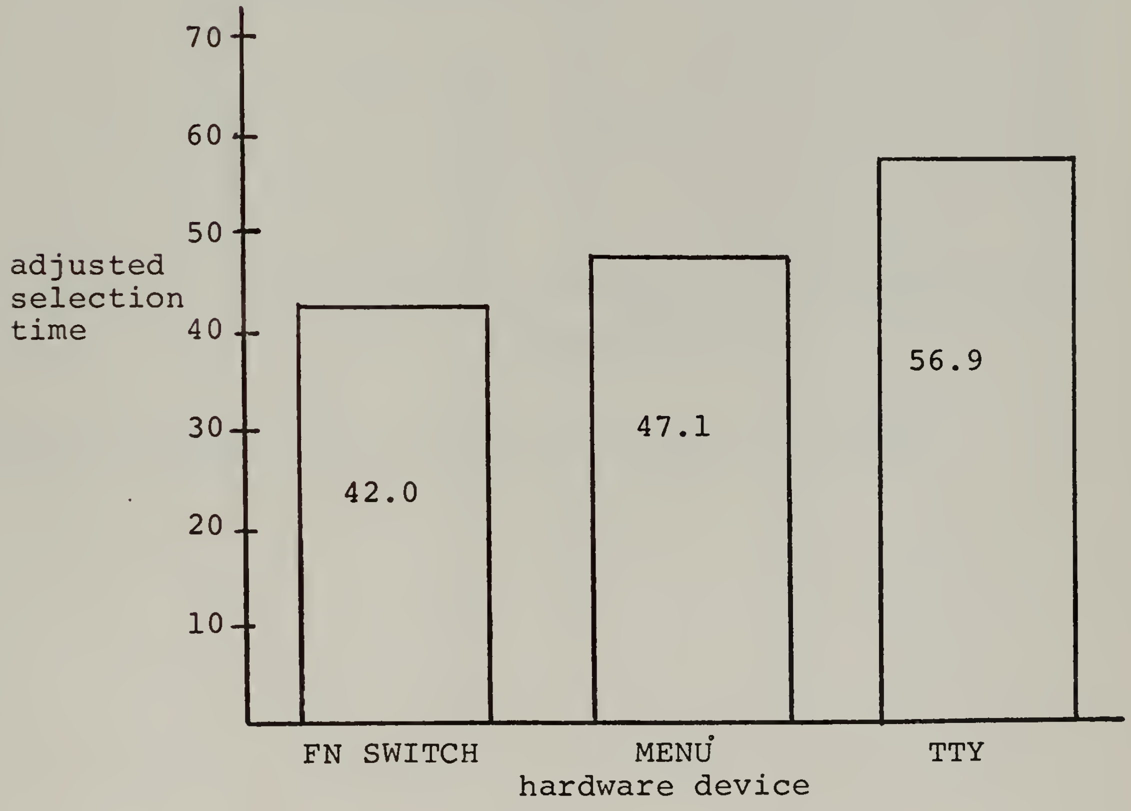


Figure 17. Mean adjust selection time for a large number of functions

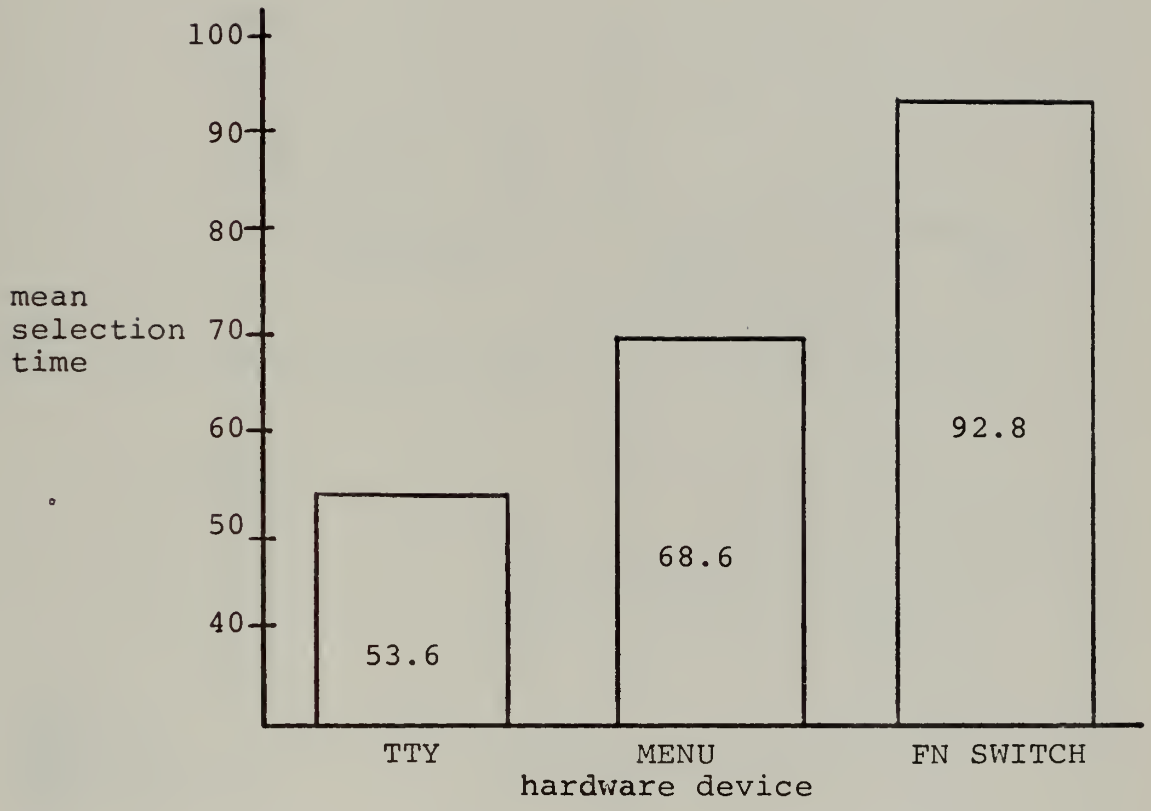


Figure 18. Mean selection time for paged functions

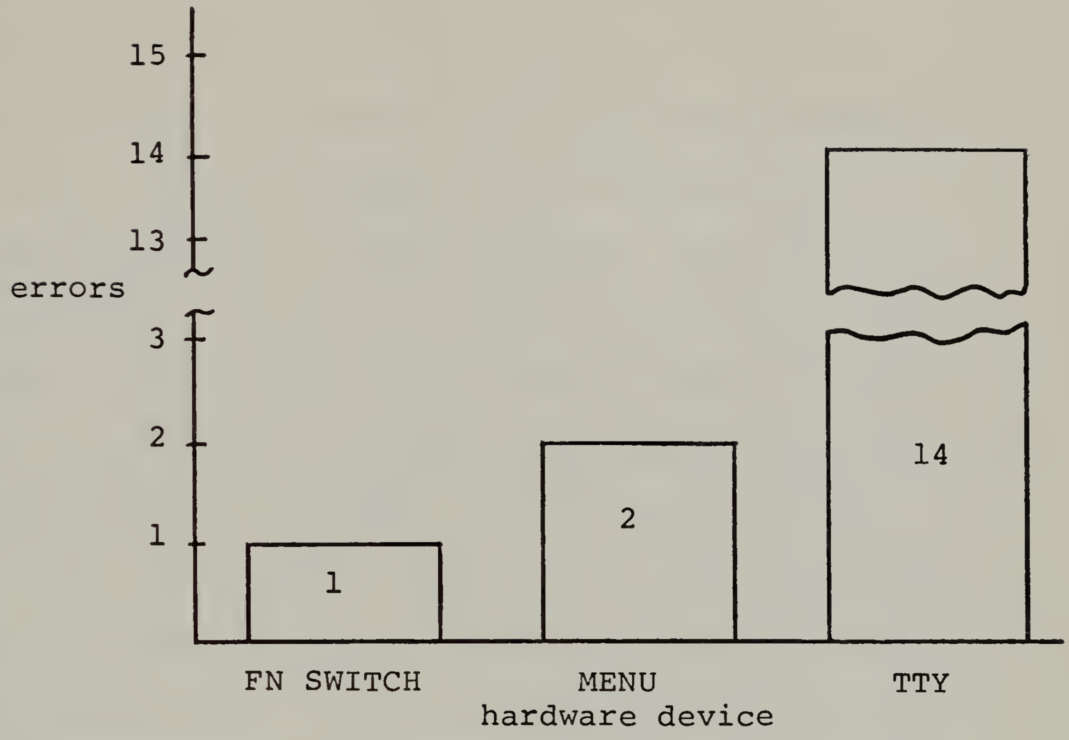
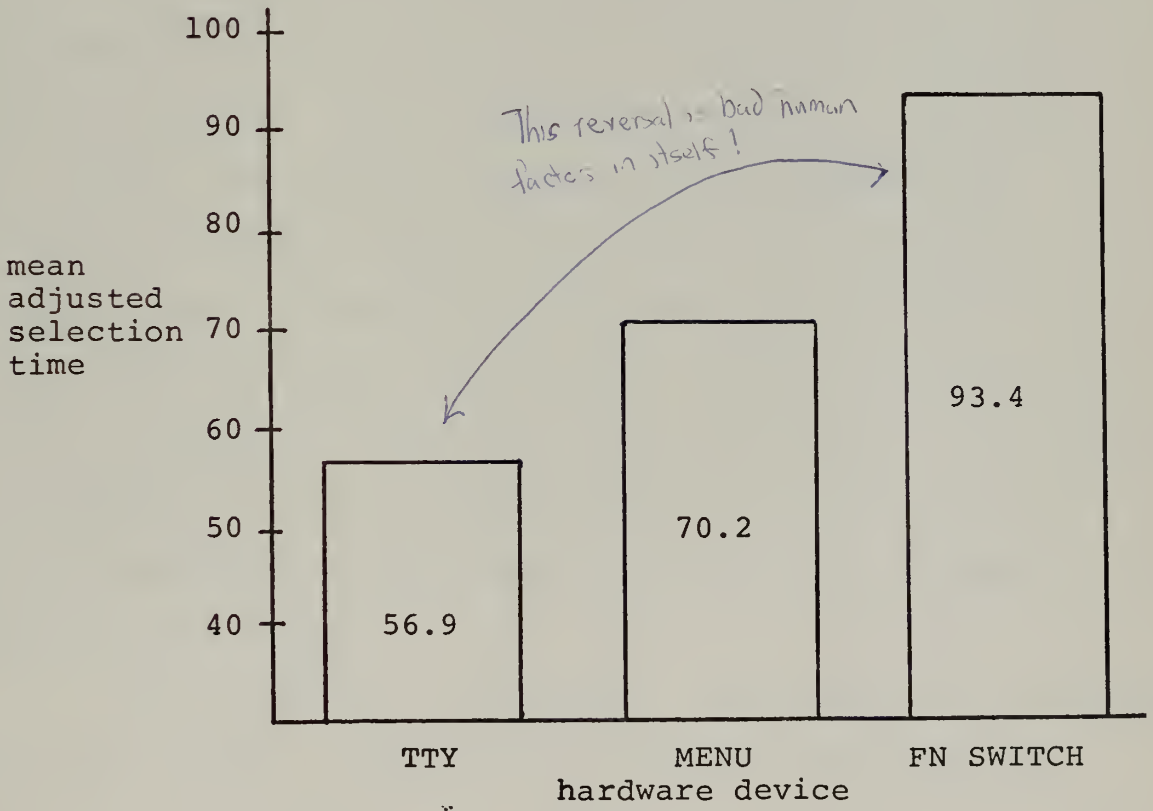


Figure 19. Total errors committed for paged functions.



This reversal is bad human factors in itself!

Figure 20. Mean adjusted selection time for paged functions.

Investigation reveals that the teletype is the most preferable and the function switches the least preferable in this case. The poor showing on the part of the function switches was caused by the requirement that the subject manually "page" through the templates to obtain the one desired. An examination of the error rates as shown in Figure 19 reveals the function switches and the menu to be the most accurate as before. The teletype, however, remained very error prone. The adjusted selection times showed the teletype to be the most favorable in spite of its high error rate. This is illustrated in Figure 20.

5. Conclusions

From the test results, several general conclusions are possible. First, it is best to implement a limited set of functions. This enables the analyst to scan the list quickly and with a minimum of frustration. This conclusion is based upon the fact that selecting from a limited number of functions was clearly faster and more error-free than either of the other choices in the experiment. It should be noted that the choice of seven functions to represent a limited number was strictly arbitrary and further testing would be required to establish an upper bound on this figure.

With regard to the hardware that should be associated with this choice, the function switches or menu would be preferable to the teletype. Since these alternatives rated so close together in the tests conducted, further investigation

concerning such things as ease of use and naturalness would be required to select between the two.

The second conclusion is that if it is not possible to work with a limited number of functions the next best alternative is to display all of the functions at once. In this case, further investigation of the three input devices considered would be required to choose one in particular since all rated closely in the testing.

Finally, if neither of the above alternatives is possible the use of the teletype would be advocated for inputting the various functions as opposed to paging with the function switch or menu. It should be emphasized that this alternative clearly rated below the other choices and should be avoided.

IV. SUMMARY

It has been shown that by combining the features of graphical communication with the traditional facilities offered by the computer it is possible to enhance the man-machine relationship.

A highly efficient interaction in which the assets of both the computer and the man are properly utilized has not been achieved. Therefore, the potential users of interactive graphical systems are finding more traditional means such as batch processing to solve their problems in spite of the delays. This failure to achieve a satisfactory uniting of man and machine has been caused by the failure of present-day systems to develop a sound communications link between the two.

An analysis of this communications link has revealed two basic components: the method by which the user controls program flow and the hardware device used to communicate commands to the computer.

Upon further investigation of the control of program flow, it was discovered that the traditional pattern of inputting information, performing calculations, and outputting the results was not satisfactory for an interactive graphical environment. Instead, a system in which the machine executes an instruction and then idles while awaiting further direction from the user is preferred. In

addition, the basic step taken by the machine prior to receiving further direction would be a function. This would provide the user with a maximum amount of control over the program without being awkward.

The functions available to a user from a particular area (who would be known as the analyst) would be prepared by a programmer and were referred to as the discipline set. A subset of these which would be common to all disciplines were referred to as basic functions.

A generalized method for problem solution and program control which would increase the capabilities of the analyst by allowing him to interact with the machine in a natural manner and with a minimum of learning was described. This consisted of six basic phases of activity as follows:

Phase A: Selecting primitives

Phase B: Defining any new primitives required

Phase C: Constructing a model of the problem

Phase D: Selecting the analysis functions to be applied

Phase E. Defining any new analysis functions required

Phase F: Outputting results

An investigation of hardware control devices revealed that for the most part there is no evidence that human factors engineering techniques were applied when selecting a particular piece of equipment over another. As a result, users are frequently frustrated in their attempts to input commands to the machine.

A function classification system was proposed in which the functions were given a number to indicate the amount of information which the user would be required to supply to completely define that particular function. It was noted that the particular hardware device used could effectively lower the class of a function.

As an example of the type of analysis that should be performed when analyzing various hardware devices tests were designed and conducted to measure the relative merits of the light pen and menu, the function switch, and the teletype. In addition, the tests measured the effectiveness of displaying a large number of functions simultaneously, displaying a limited number of functions, and displaying a large number of functions with only a limited number shown at any instant.

APPENDIX A

The following is a tabulation of the actual test results. The times listed are in seconds.

LARGE NUMBER OF FUNCTIONS

fn switch																
time	34	31	35	42	31	40	44	43	60	52	67	36	36	34	28	47
errors	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
menu																
time	38	61	54	39	48	66	35	76	34	42	50	37	36	38	44	50
errors	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0

LIMITED NUMBER OF FUNCTIONS

fn switch																
time	30	37	44	26	20	26	25	31	21	23	20	23	21	17	22	28
errors	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
menu																
time	36	44	36	41	33	37	31	40	23	42	35	28	28	63	33	41
errors	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0

PAGED FUNCTIONS

fn switch															
time	97	124	78	72	81	109	105	104	89	89	102	77	78	94	90
errors	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

menu																
time	80	73	70	78	65	62	61	93	50	74	98	46	60	50	66	72
errors	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0

TELETYPE

teletype																
time	55	66	72	50	47	53	69	45	53	41	28	48	52	71	59	49
errors	1	2	0	0	1	0	1	3	0	1	0	2	1	1	0	1

LIST OF REFERENCES

1. Parslow, R. D., Prowse, R. W., and Green, R. E., Computer Graphics Techniques and Applications, Plenum Press, 1969.

2. Thomas, E. M., "Display Programming - 1968," Proceedings of the Society for Information Display, v. 10, Winter, 1969.

3. Siders, R. A., Computer Graphics A Revolution in Design, American Management Association, 1966.

4. Sutherland, I. E., SKETCHPAD: A Man-Machine Communication System, paper presented at the Spring Joint Computer Conference, May, 1963.

5. Jacks, E. L., and others, The GM DAC-1 System, Design Augmented by Computer, paper presented at the Fall Joint Computer Conference, 1964.

6. Streit, E., VIP: A Conversational System for Computer-Aided Graphics, paper presented at the Conference on Pertinent Concepts in Computer Graphics, 1969.

7. Ninke, W. H., Interactive Computer Graphics: Some Interpolations and Extrapolations, paper presented at the Conference on Pertinent Concepts in Computer Graphics, 1969.

8. Lingren, N., "Human Factors in Engineering - Part II," I.E.E.E. Spectrum, April, 1966.

9. System Development Corporation Report TM-LO-790/000/00, An Experimental Study of 465L Message Processing, Watson, W., Clark, D., and Sasmore, R., 1962.

10. Corbin, H. S., and Frank, W. L., Display Oriented Computer Usage System, paper presented at the Twenty-first National Conference of the ACM, 1966.

11. Sutherland, I. E., SKETCHPAD: A Man-Machine Communications System, paper presented at the Spring Joint Computer Conference, 1963.

12. Lingren, N., "Human Factors in Engineering - Part II," I.E.E.E. Spectrum, April, 1966

13. Keast, D. N., "Survey of Graphic Input Devices," Machine Design, August, 1967.

14. Baker, J. D., From the Diet of Worms to the Bucket of Worms: A Protest Concerning Existing Display Dogma for Information Systems, paper presented at the Second Congress on the Information System Sciences, 1964.

15. Kinney, G. C., "Symbols, Sense and Sin," Proceedings of the Society for Information Display, v. 10, Winter, 1969.

16. Fitts, P. M., Human Engineering in the National Defense, paper presented at the Panel on Human Engineering and Psychophysiology, June, 1953.

17. Lingren, N., "Human Factors in Engineering - Part I," I.E.E.E. Spectrum, March, 1966.

BIBLIOGRAPHY

1. Abzug, I., "Graphic Data Processing," *Datamation*, v.11, January, 1965.

2. Allen, T. H., and Foote, J. E., *Input/Output Software Capability for a Man-Machine Communication and Image Processing System*, paper presented at the Fall Joint Computer Conference, 1965.

3. Baker, J. D., *From the Diet of Worms to the Bucket of Worms: A Protest Concerning Existing Display Dogma for Information Systems*, paper presented at the Second Congress on the Information System Sciences, 1964.

4. Carlson, R. D., "Computer Graphics and Manufacturing," *Information Display*, v. 6, January, 1969.

5. Chapanis, A. R., *Research Techniques in Human Engineering*, John Hopkins Press, 1962.

6. Chasen, S. H., *Experience in the Application of Interactive Graphics*, paper presented at the University of Illinois Conference, 1967.

7. Chasen, S. H., *The Introduction of Man-Computer Graphics into the Aerospace Agency*, paper presented at the Fall Joint Computer Conference, 1965.

8. Christensen, C., and Pinson, E. M., *Multi-Function Graphics for a Large Computer System*, paper presented at the Fall Joint Computer Conference, 1967.

9. Compton, R. D., "Information Displays," *EDN The Electronics Engineers Design Magazine*, v. 11, April, 1966.

10. Coons, S. A., *An Outline of the Requirements of a Computer-Aided Design System*, paper presented at the Spring Joint Computer Conference, May, 1963.

11. Davis, M. R., and Ellis, T. O., *The RAND Tablet: A Man-Machine Graphical Communication Device*, paper presented at the Fall Joint Computer Conference, 1964.

12. Dertouzous, M. L., and Graham, J. L., *A Parametric Graphic Display Technique for On-Line Usc*, paper presented at the Fall Joint Computer Conference, 1966.

13. Diebold, J., Beyond Automation, McGraw Hill, 1960.

14. Electronic Systems Technical Report ESD-TDR-63-545, Study of Computer Manual Input Devices, Pollack, W. T. and Gildner, G. G., September, 1963.

15. English, W. K., Engelbart, D. C., and Berman, W. L., "Display Selection Techniques for Text Manipulation," I.E.E.E. Transactions on Human Factors in Electronics, v. 8, March, 1967.

16. Fetter, W. A., Computer Graphics in Communication, McGraw Hill, 1965.

17. Gruenberger, F. J., Computer Graphics, Thompson, 1967.

18. Hargraves, B., Image Processing Hardware for a Man-Machine Graphical Communication System, paper presented at the Fall Joint Computer Conference, 1964.

19. Hicks, C. R., Fundamental Concepts in the Design of Experiments, Holt Rinehard and Winston, 1964.

20. Hobbs, L. C., "Display Applications and Technology," Proceedings of the I.E.E.E., v. 54, December, 1966.

21. Hornbuckle, M. D., "Man-Computer Graphics for Computer Aided Design," Proceedings of the I.E.E.E., v. 54, March, 1967.

22. Hurwitz, A., Citron, J. P., and Yeaton, Y. B., GRAF: Graphic Additions to FORTRAN, paper presented at the Spring Joint Computer Conference, 1967.

23. Jacks, E. L., and others, The GM DAC-1 System, Design Augmented by Computers, paper presented at the Fall Joint Computer Conference, 1964.

24. Jacks, E. L., A Laboratory for the Study of Graphical Man-Machine Communication, paper presented at the Fall Joint Computer Conference, 1964.

25. Johnson, T. E., SKETCHPAD-III: A Computer Program for Drawing in Three Dimensions, paper presented at the Spring Joint Computer Conference, 1963.

26. Johnson, W. L., "The Ford Computer Graphics Project," Computers and Automation, November, 1967.

27. Joyce, J. D., and Ciamciolo, M. J., Reactive Displays: Improving Man-Machine Communication, paper presented at the Fall Joint Computer Conference, 1967.

28. Karplus, W. J., On-Line Computing, McGraw Hill, 1967.

29. Kennedy, J. R., A System for Time-Sharing Graphics Consoles, paper presented at the Fall Joint Computer Conference, 1966.

30. Kulsrud, H. E., "A General Purpose Graphical Language," Communications of the ACM, v. 11, April, 1968.

31. Lewin, M. H., "An Introduction to Computer Graphic Terminals," Proceedings of the I.E.E.E., v. 55, June, 1967.

32 Lichtenberger, W. W., and Pirtle, M. W., A Facility for Experimentation in Man-Machine Interaction, paper presented at the Spring Joing Computer Conference, 1962.

33. Licklider, J. C., and Clark, W. E., On-Line Man-Computer Communication, paper presented at the Spring Joint Computer Conference, 1962.

34. Loewe, R. T., Sesson, R. L., and Horowitz, P., "Computer Generated Displays," Proceedings of the I.E.E.E., v. 49, January, 1961.

35. Loewe, R. T., and Horowitz, P., Display System Design Considerations, paper presented at the Eastern Joint Computer Conference, 1961.

36. Machover, C., Graphic CRT Terminals - Characteristics of Commercially Available Equipment, paper presented at the Fall Joint Computer Conference, 1967.

37. Major, J. H., and Sutherland, I. E., "On the Design of Display Processors," Communications of the ACM, v. 11, June, 1968.

38. Morrison, R. A., Graphic Language Translation with a Language Independent Processor, paper presented at the Fall Joint Computer Conference, 1967.

39. Newman, W. M., A System for Interactive Graphical Programming, paper presented at the Spring Joint Computer Conference, 1968.

40. Ninke, W. H., GRAPHIC-1 A Remote Graphical Display Console System, paper presented at the Fall Joint Computer Conference, 1965.

41. Parslow, R. D., Prowse, R. W., and Green, R. E., Computer Graphics Techniques and Applications, Plenum Press, 1969.

42. Poole, H. H., Fundamentals of Diaplay Systems, Spartan Books, 1966.

43. Prince, M. D., "Man-Computer Graphics for Computer-Aided Design," Proceedings of the I.E.E.E., v. 54, December, 1966.

44. Rippy, D. E., and Humphries, D. E., MAGIC - A Machine for Automated Graphics Interface to a Computer, paper presented at the Fall Joint Computer Conference, 1965.

45. Roberts, L. G., Graphical Communication and Control Languages, paper presented at the Second Congress of Informational Sciences, 1964.

46. Ross, D., and Rodriques, J., Theoretical Foundations for the Computer-Aided Design System, paper presented at the Spring Joint Computer Conference, 1963.

47. Schwinn, P. M., A Problem Oriented Graphic Language, paper presented at the National Conference of the ACM, 1967.

48. Seitz, R. M., and Chasen, S. H., "On-Line Systems and Man-Computer Graphics," Computers and Automation, November, 1967.

49. Siders, R. A., "Computer Aided Design," I.E.E.E. Spectrum, November, 1967.

50. Skinner, F. D., "Computer Graphics - Where are We?," Datamation, v. 11, May, 1966.

51. Stotz, R., Man-Machine Console Facilities for Computer-Aided Design, paper presented at the Spring Joint Computer Conference, 1963.

52. Sutherland, I. E., SKETCHPAD: A Man-Machine Communication System, paper presented at the Spring Joint Computer Conference, 1963.

53. System Development Corporation Report TM-LO-790/000/00, An Experimental Study of 465L Message Processing, Watson, W., Clark, D., and Sasmore, R., 1962.

54. Teixeira, J. F., and Sallen, R. P., Sylvania Data Tablet: A New Approach to Graphic Data Input, paper presented at the Spring Joint Computer Conference, 1968.

55. Thomas, E. M., "Display Programming - 1968," Proceedings of the Society for Information Display, v. 10, 1969.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	LT R. D. DeLaura, USNR (Code 52Dl) Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
4.	LT Robert W. Cassell Jr., USN 8 Oakview Drive Northfield, New Jersey 08225	1

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)*

Naval Postgraduate School
Monterey, California 93940

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

3. REPORT TITLE

AN ANALYSIS OF MAN-MACHINE COMMUNICATION IN AN INTERACTIVE GRAPHICAL ENVIRONMENT

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Master's Thesis: (December 1971)

5. AUTHOR(S) *(First name, middle initial, last name)*

Robert Wayne Cassell, Jr.

6. REPORT DATE December 1971	7a. TOTAL NO. OF PAGES 61	7b. NO. OF REFS 17

8a. CONTRACT OR GRANT NO

b. PROJECT NO.

c.

d.

9a. ORIGINATOR'S REPORT NUMBER(S)

9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

11. SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY

Naval Postgraduate School
Monterey, California 93940

13. ABSTRACT

The failure of present-day interactive graphical systems to provide for an effective relationship between man and machine is examined. An inefficient communications link is shown to be the basis of the problem. The manner in which the user controls program flow and the hardware utilized in inputting commands are revealed as the basic components of communication.

Program control is investigated. A revised control method is described which consists of a series of function executions and six basic phases of activity for program control.

With respect to input hardware, more human factors analysis is advocated. To assist in this analysis, a function classification system is described. An example test is conducted to compare the relative merits of the light pen with menu, the teletype, and function switches.

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Interaction						
Graphics						
Man-computer communications						
Light pen						
Function switches						
Keyboard						

Thesis 133806
C2775 Cassell
c.1 An analysis of man-machine communication in an interactive graphical environment.

6 SEP 73	21335
5 MAY 80	26795
[illegible] APR 82	27455
3 NOV 82	28126
JUL [illegible] 85	30150
NOV 10 85	30612
[illegible]	[illegible]
11 JUN 87	32238

Thesis 133806
C2775 Cassell
c.1 An analysis of man-machine communication in an interactive graphical environment.