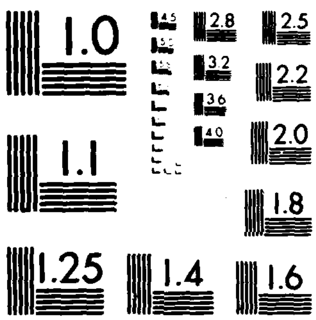
AD-A097 520   ANALYTICS INC WILLOW GROVE PA   F/G 9/2
AN INTRODUCTION TO HUMAN OPERATOR SIMULATOR.(U)
DEC 79 M I STRIEB, R J WHERRY   N62269-79-C-0235
UNCLASSIFIED TR-1400.02-D   NL

END
DATE
FILMED
5-81
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

LEVEL III

1

Technical Report
1400.02-D

AN INTRODUCTION TO THE HUMAN OPERATOR SIMULATOR

Melvin I. Strieb
Robert J. Wherry, Jr., PhD

December, 1979

Contract No. N62269-79-C-0236

ANALYTICS
2500 MARYLAND ROAD, WILLOW GROVE, PA. 19090

(14) TR — Technical Report 1460.02-D

AN INTRODUCTION TO THE HUMAN OPERATOR SIMULATOR

Melvin I. Strieb
Robert J. Wherry, Jr., PhD

December 1979 (12) 83

Contract No. N62269-79-C-0235

ANALYTICS
2500 MARYLAND ROAD, WILLOW GROVE, PA. 19090

EXECUTIVE SUMMARY

HOS is a digital computer program designed to be used in the evaluation of complex crewstations. It enables an analyst to dynamically simulate the activities of an operator (perception, anatomy movement, decision-making, etc.) and the performance of the hardware in response to the operator's actions. Like a real operator, the HOS operator must be taught how to use a system by supplying it with a description of how the equipment operates, the circumstances under which the equipment is used, and tactical strategies to be pursued in order to accomplish specific goals. The operator can then be placed in a specific tactical environment and will respond to the dynamics of the situation just as an actual operator would under similar circumstances. Models of human performance in HOS predict how long each specific operator activity will take. The HOS operator will adapt his behavior to the dynamics of the tactical situation in accordance with the rules by which he has been trained. By controlling the tactical situations, the analyst can use HOS to obtain data on human and system performance in hypothetical tactical situations--data that heretofore could only be obtained at too late a stage in the system development to have a significant impact on the system design. HOS enables different system configurations and operator strategies to be tested and studied without having to build hardware or train operators, or run actual experiments or exercises.

Although HOS was designed for the human engineering community as a design and evaluation tool, its potential usefulness extends well beyond the classical scope of human engineering design and evaluation probelms. This presentation will focus on the role that HOS can play in the system design process and on specific details of the HOS operator models. More detailed discussions on how to use HOS are presented in the HOS Study Guide (Strieb, Glenn, and Wherry, 1978).

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

LIST OF FIGURES

1. THE ROLE OF OPERATOR MODELS IN SYSTEM DESIGN AND EVALUATION

1.1 HUMAN ENGINEERING AND THE SYSTEM DESIGN PROCESS

In order to place the role of HOS in the proper perspective, it is useful to look at the system design process, at the role that human engineers play in that process and some of the tools and models that have been developed to assist in this process, particularly with respect to human performance evaluation.

One way of looking at the role of the human engineer is in terms of the types of analyses required to support each stage of the system acquisition process. Program development can be divided into four major phases, based on major decision points in the weapons system acquisition process:

(1) Program Initiation

(2) Demonstration and Validation

(3) Full Scale Engineering Development

(4) Production and Deployment

Within each of these phases, there are a variety of types of analyses required by or performed by the human engineer. During the program initiation stage, for example, some of the types of analyses are:

- *Identification of Operational Conditions*
- Requirements Analysis
- Preliminary Function Allocation

- Preliminary Manning Analysis
- Preliminary Task Analysis
- Operational Sequence Analysis

During the concept refinement and prototype development phase, some of the additional types of analyses required are:

- Crewstation Workspace Layouts
- Control/Display Design Requirements
- Man-Machine Tradeoffs
- Maintenance/Training Objectives and Requirements
- Personnel Selection Requirements

In the full scale engineering development stage, the required analyses are associated with developing training programs and operational procedures. And, finally, in the production and deployment stages, analyses include the development of solutions to operational problems, and the analysis, by similar techniques, of retrofits and upgrades.

Related to these are the human factors program requirements defined in MIL-H-46855. These include:

- Defining and allocating system functions.
- Information flow and processing analysis.
- Estimates of potential operator/maintainer processing capabilities.
- Equipment identification.
- Task analysis.
- Analysis of critical tasks.
- Loading analysis.

- Preliminary and detailed system and subsystem design.
- Studies, experiments, laboratory tests.
- Mockups and models.
- Dynamic simulation.
- Design drawings.
- Workspace environment.
- Test and evaluation.

Another way of viewing the role of the human factors engineer is in terms of the specific problems with which he is faced. In general, human factors system design efforts can be subdivided into six substantive areas:

(1) Tasks and Functions to be Allocated to Humans

(a) The type of functions that should be allocated to humans.

(b) The degree to which various functions must be automated to ensure that system operators will not be overloaded.

(c) The specific functions taht should be allocated to each equipment operator.

(2) Human Information Processing Procedures

(a) The relative priority of various assigned tasks.

(b) The specific steps that the human is expected to perform in each assigned task.

(c) Various kinds of information processing performed by each human element.

(d) The expected speed and accuracy of the operator's performance on each task.

(3) Panel and Console Design

(a) Required types and sizes of displays and controls.

(b) Their locations and arrangement on panels and consoles.

(c) Panel markings and arrangement within each crewstation.

(4) Equipment Testing and Handling

(a) Equipment handles, fasteners, connectors, and access points for test and inspection.

(b) Special support equipment to aid the loading, unloading, transporting, etc., of equipment.

(c) Special tools and equipment required for fault detection, isolation, and correction.

(5) Environmental Control, Ingress and Egress

(a) Supports and restraints required for each crew member.

(b) Personal equipment and clothing.

(c) Normal and emergency ingress and egress equipment.

(d) Crew station lighting, air conditioning, noise suppression, and other habitability considerations.

(6) Training Manuals and Job Aids

The information and procedures that the human would have difficulty memorizing and that are therefore stored as reference material.

It would be useful to many of the above types of design activities to be able to actually "place" actual operators in a proposed system to provide a realistic assessment of how well the system is suited to human capabilities and limitations. Unfortunately, it is usually not until late in the full scale engineering development stage that such assessments can be made. Before then, analyses have to be based on models, mockups, and "educated guesses."

Therefore, a variety of tools have been developed to assist in the system design and evaluation process. To better understand how the HOS model relates to some of these other tools, it is useful to review the role of models and some of the critical issues associated with model development.

1.2 THE ROLE OF MODELS

Models are developed for any of several reasons:

- Because phenomena that we wish to study, such as human performance, are too complex to be dealt with directly.
- Because systems that we wish to study have not been developed to a sufficient state that they can be studied directly.
- Because conditions that we wish to study cannot be created except at great expense or at the risk of life.

Models attempt to reduce the complexities of actual situations to simpler forms that are more amenable to study and analysis.

Because of the complexity of human performance, models of human performance may exist any any one of a number of levels of detail. At certain stages of system development, a model that describes human performance in terms of the gross tasks that the operator must perform might suffice; in later stages, it may be necessary for these tasks to be broken down into descriptions of how the operator interfaces with his displays and controls and, still further, into the actual eye and hand movements that he makes and the cognitive processes that govern his selection of actions. For still other purposes, it may be important for these processes to be broken down into the subprocesses that control movement and brain functioning and ultimately into biochemical processes. Therefore, when attempting to choose a model of human performance for a particular application, the modeler must:

(1) Determine the point at which he no longer cares to attempt to describe in any more detail the actual subprocesses, and

(2) Ensure that the subprocesses that are described are concatenated in such a way that they accurately reflect the behavior of the system being modeled.

The first point addresses the level of detail attained by any particular model -- a critical issue which must be carefully assessed when attempting to select the appropriate model for any particular problem. What information the model can be expected to supply and, conversely, the choice of a level of detail, are determined by the problems that the model is designed to address.

After reviewing the human performance literature, it becomes apparent that there are vast differences in the levels of detail addressed by the various models that have been developed as well as the problems they are designed to address. To place HOS in perspective with respect to these other operator models, it is useful to characterize models as either:

- Task Analytic Models
- Control Theory Models
- Micro-process Models

Each type of model attempts to cope with both the complexity and variability of human performance in different ways. The characteristics of these models are discussed below.

1.3 TASK ANALYTIC MODELS

Task analytic models are those in which the tasks assigned to the operator(s) are described, either implicitly or explicitly, as a network in which the timing and sequencing through each mission stage (network "node") are structured by the analyst. In these models, it is usually the user's responsibility to predetermine all the characteristics of each node, including times (or conditions) under which the node will be executed, nominal execution times, probabilities of successful completion, transition probabilities to other nodes, etc. Task analytic models can be of highly varying degrees of complexity, sophistication, and detail. For example, the methodology known as task analysis is a task analytic model with relatively few dynamic features -- it can be used to produce a nominal timeline for estimating system performance, but is only minimally responsive to the possible variations that can occur in actual operations that would impact system performance. Other analytic models (e.g., Siegel-Wolf and SAINT) are stochastic (Monte-Carlo) models in which the network is executed many times with input conditions and, possibly, nodal characteristics that vary according to distributions supplied by the modeler, resulting in outputs that predict summary performance measures such as overall mission success. Other models (e.g., the CAFES, WAM, and FAM models) perform primarily bookkeeping functions or have built-in decision algorithms that modify the task network to optimize performance.

In general, it is the modeler's responsibility when using these models to fully describe the task network and possible interactions. The accuracy of such inputs may be either good or bad, depending on the modeler's apriori knowledge and experience with the situation being modeled.

Task analytic models are human performance models in that they attempt to describe, in some sense, the performance achieved by one or more humans in combination with a system. However, they do not contain a model of the human, per se. The success of the model is, in fact, dependent on how successful the modeler is at describing the network and in assigning times, conditions, and probabilities that accurately reflect the situation being simulated.

1.4 CONTROL THEORY

In (manual and optimal) control theory models, the interactions between the operator and the system are represented by servo-control models. Unlike the task analytic models, control theory models *do* have an explicit (and general) model of human performance -- namely that an operator behaves in such a way that errors are minimized within fixed performance constraints.

Control theory models have typically been used to examine human performance in situations in which a single display/control relationship and the operator's response under a variety of environmental conditions are critical. Thus, manual and optimal control modelers have devoted extensive study to, for example, pilot landing performance under a variety of external (e.g., buffeting) conditions.

1.5 MICRO-PROCESS MODELS

Micro-process models are detailed representations of the operator in terms of the physical, psychological and phsysiological processes that are involved in carrying out a task. In order to use such a model, the analyst must describe the specific tasks that the operator must perform. The model then generates performance predictions for the tasks based upon the modeled processes. Note that there are actually two models involved -- the description of the tasks to be performed is a model (just as it was in the task analytic models) and the way in which the micro-process model translates those tasks into activities is also a model.

1.6 A COMPARISON OF MICRO-PROCESS, TASK ANALYTIC, AND CONTROL THEORY MODELS

To illustrate the differences between the various types of models, suppose that we are interested in modeling an operator's tracking behavior, perhaps as part of a larger model of pilot performance.

The simplest task analytic description for this situation would be the statement:

PILOT TRACKS TARGET

with a time charge assessed for that activity. A more complex task analytic model might subdivide the macro-level activity "tracks" into more micro-level activities -- for example:

PILOT LOOKS AT TARGET

PILOT MANIPULATES CONTROL

and again times would be assessed for each activity. A dynamic (stochastic) task analytic model might include logic that described the probability that the pilot would need to perform a corrective activity and/or probabilities that the operator would have to iterate through a control loop.

A control theory model of the same problem would typically concern itself primarily with the dynamics of the control loop. It would include a detailed mathematical description of the target's behavior and the pilot's response to that stimulus and mathematically model features such as the operator and the system response lags, motor noise, etc.

A micro-process model, on the other hand, would take the activities described by the task analytic description of the operator's tasks and the dynamic behavior of the system, as described by the control theory models, and combine them to model the behavior of the operator in response to system stimuli. For example, a micro-process model might break down the statement

PILOT LOOKS AT TARGET

into a set of activities shown in Figure 1.

There are significant differences between the task analytic model and the micro-process model for these tasks. For example, unlike the task analytic model which would have assessed a time charge whether or not the pilot was already looking at the target, the micro-process model would not. A micro-process model retains complete knowledge both of what the operator is doing and what the target is doing at any instant, thereby enabling unnecessary time charges to be eliminated and accurate time charges to be assigned only for operator activities that actually are needed. Moreover, the time charges assessed can be based on actual experimental data rather than on subjective estimates by an analyst, since the activities are at a level for which experimental data is available.

1.7 DETERMINISTIC VS STOCHASTIC MODELS

Micro-process models are basically extensions of the task analytic approach. They reduce task activities to the level of elementary human processes -- anatomy movement, information absorption, etc., -- and combine operator activities with the dynamics of the system and the external world. Thus, some

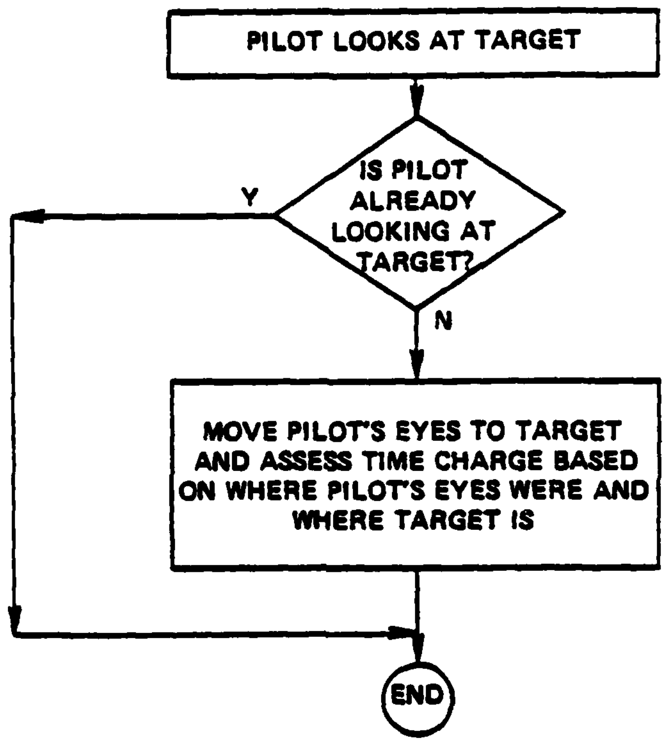


Figure 1. A Micro-process Tracking Model

of the features of micro-process models could be achieved in a dynamic task analytic model in which each task analytic node corresponded to a micro-process. However, there is a fundamental difference between the way in which micro-process models and both task-analytic and control theory models view human performance. In particular, micro-process models make the fundamental assumption that an operator's behavior is explainable and not random -- that an operator's actions and the times that those actions will take are determined fully by the state of the system and the operator's goals at any particular point of time. Thus, micro-process models are basically deterministic models (although individual micro-processes may contain random components), rather than stochastic models. However, because of the way in which various micro-models interact, the output from a micro-process model will exhibit variability, making it seem, in some cases, indistinguishable from stochastic output.

1.8 THE MAIL SORTING PROBLEM

As a further illustration of the difference between the micro-process and the task analytic approaches, let us compare the task analytic version of a problem with the micro-process version of the same problem. The sample problem that will will use is the design of a mail sorting console for the Post Office. In designing such an operator station, a variety of design decisions must be made -- characteristics and locations of individual keys, pacing of the system, etc. Assume that it has been decided that the machine will be self-paced -- in order to have the next letter presented, the operator must depress a feed key. Once a letter is at the read station, the operator must make a decision as to whether the first three digits of the letter indicates a destination within the local area or not. If the area code is local, the operator must depress a local key and then enter the last two digits of the zip code corresponding to the city zone. If the area code is not local, then the operator must enter the three digits corresponding to the area code.

A task analytic network model for this system is shown in Figure 2. Note that the analyst is required to supply means and standard deviations at each node in the network for the times that the activities represented by the nodes will take. In addition, the analyst must supply transition probabilities at each node where a decision is made. However, even if the analyst had values for these numbers for a particular situation based on experimental data, the numbers would most likely be totally inappropriate for a new system that was in any significant sense different from the one from which the numbers came. Thus, in many respects, it would be virtually impossible to use the task analytic network with any confidence to resolve the issue of interest -- the efficacy of alternative designs.

The flowchart for a micro-process model of the same situation would be almost identical. But with a micro-process model, the analyst would not have to supply the times and transition probabilities based on the specific task damands, the characteristics of the operator and of the system. For example, when the task "DEPRESS FEED KEY" was to be executed, a micro-process model would determine the time required to move the operator's hand from wherever it was at the time the action was required to the control and the time for the control manipulation, based on the characteristics of the control. At the decision point "IS AREA CODE LOCAL," a micro-process model would cause the operator to "look" at the first three digits of the zip code, determine their value, and make the decision as to whether it was local or not.

The primary advantage of the micro-process model, besides the reduction in the number of input data values required, is that the micro-process model enables us to ask questions that could not be asked of the task analytic model. For example, what would be the effect of changing the control locations? Of making the system forced paced rather than self-paced? Of changing the control characteristics?

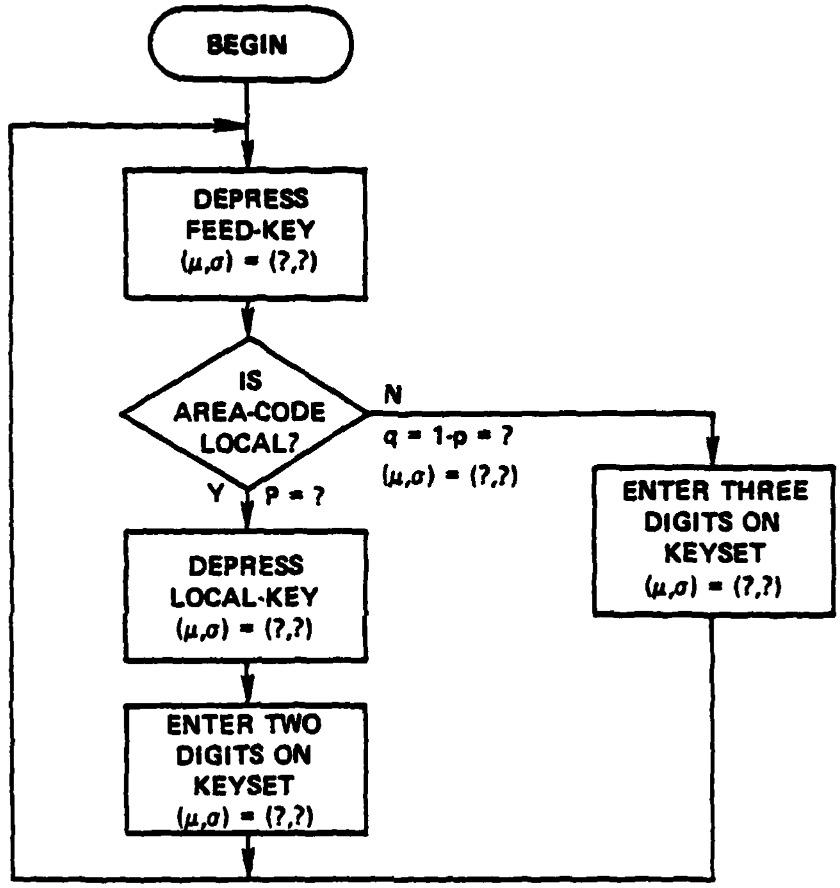


Figure 2. A Task Analytic Model for the Mail Sorting Problem

The HOS model is an example of a micro-process model and is the only micro-process model that has been developed to such an extent that it can be applied to complex system design problems. The following sections will describe the HOS model in more detail.

2. THE HOS MODEL

2.1 CHARACTERISTICS OF THE HOS OPERATOR

The HOS operator is assumed to be a highly motivated, well-trained, average operator. In addition, the HOS model assumes that the performance of the operator and the rules that an operator should use in operating a system can be explicitly described as a series of procedures and statements that describe the operator's activities and the decisions that the operator must make. Other characteristics of the HOS operator are:

(1) The position of the operator relative to the displays and controls in the crewstation must be fixed -- i.e., the operator is stationary.

(2) The operator can only process one task statement at a time. However, once a statement has been processed, the operator can begin work on the next statement, even though actions initiated in the preceding statement may still be continuing. Thus, the HOS operator can be performing several actions simultaneously -- e.g., reading a display and manipulating a control simultaneously, manipulating two controls, one with each hand, etc.

(3) The operator is a multi-processor in the sense that several procedures can be "active" simultaneously, although only one can be worked on at any given time.

(4) Operator performance variability in HOS is assumed to be the result of differences in performance capabilities coupled with differences in operator strategies. Differences in performance capabilities are represented by parametric differences in the functional relationships in the micro-models. Differences in operator strategies are representable as either different decision rules in the operator procedures or as differing prioritizations of the operator procedures.

(5) The HOS operator carries out instructions without omitting a step, making an incorrect decision (based on the decision rules specified in the instruction set) or incorrectly carrying out an instruction.

2.2 OPERATOR ERROR

The last point above refers to one of the most controversial issues associated with HOS -- its model of operator error. To understand this model, it is important to remember that the primary objective for which HOS was developed was the evaluation of the nominal performance of a system by a *well-trained, average* operator. By definition, a well-trained operator is one who carries out instructions "by the book," without omitting a step, making an incorrect decision (based on the decision rules specified in the instruction set), or incorrectly carrying out an instructions. However, this definition does not preclude all sources of operator error. For HOS, the significant sources of operator error are:

(1) Requiring the operator to perform more activities in a given period of time than possible (because of human and/or equipment limitations), thereby causing the operator to "fall behind" in the mission.

(2) Giving the operator an incorrect set of decision rules and/or operating instructions, thereby causing tactical and/or operational errors.

(3) Giving the operator poor displays and/or controls that do not permit information to be read or controlled with sufficient accuracy to permit proper operation of the system, causing errors to occur in carrying out subsequent (or concurrent) operations and/or requiring the operator to invest more time, once again causing the operator to fall behind in the mission.

These types of errors *result* in operator performance errors, but are really failures in the design of the system -- flaws which the human factors engineer must address in proposing design modifications. They are problems created when system designers fail to take into account human performance limitations. Clearly, they are not errors of the same sort as when an operator inadvertently pushes a wrong button -- such errors are either random and of low frequency (in which case it is unfair to use them to evaluate the nominal performance of the system) or caused by working the operator beyond capacity. They are, however, the types of errors that must be engineered out of the system.

2.3 THE HOPROC LANGUAGE

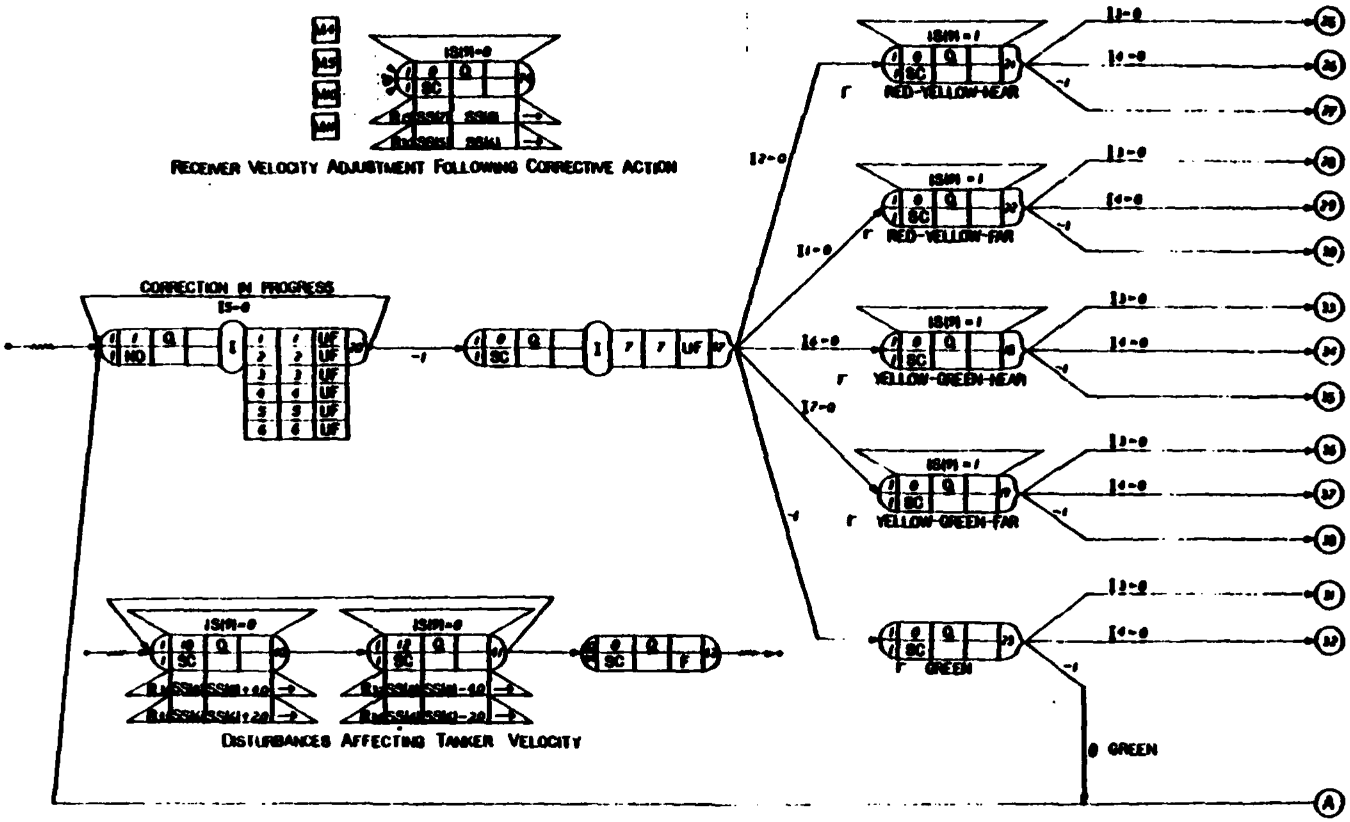
In order to be able to describe the operator activities in a task analytic or a micro-process model, it is necessary to have a formal task language. One task analytic model, the System Analysis of an Integrated Network of Tasks (SAINT) model, uses a graphical notational system (Figure 3) to represent the task network. The SAINT notational system can be readily converted to numeric strings that describe each task and input to the SAINT program.

HOS uses an English/FORTRAN-like language -- the Human Operator Procedures (HOPROC) language -- for the same purpose. However, HOPROC statements do not need to be converted into numeric entries -- HOS interprets HOPROC statements directly, just as an actual system operator can interpret English instructions.

The HOPROC language is divided into three major sections -- the title declarations section, the hardware section, and the operator section. In the title declarations section (Figure 4), the analyst identifies the names of all the displays and controls in the crewstation and their generic characteristics -- whether they are discrete or continuous, their settings (if any) and their associated scale factors (if any).

The operator section is divided into a set of *operator procedures* and a set of *operator functions*. The operator procedures are English-like statements that describe the operator's tasks. The operator functions are FORTRAN-like descriptions of the mental calculations that the operator has to perform to carry out those tasks.

Similarly, the hardware section is divided into a set of *hardware procedures* and *hardware functions*. The hardware procedures describe the hardware changes that occur as the result of the actions of the operator, as well as independent events, such as the movement of external targets, changes in the environment, etc. Like the operator procedures, the hardware procedures



SCALE FACTOR FOR FUNCTION SC = 1.0

Figure 3. Portion of a SAINT Task Network

```
SYSTEM          MAIL SORTING CASE A
SETTING SECTION
OSTATE SECTION
ARGUMENT SECTION
      AMOUNT
      COUNT
      NUMBER-OF-DIGITS
DISPLAY SECTION
      ZIP-CODE
CONTROL SECTION
      LOCAL-KEY             MOMENTARY
      FEED-KEY              MOMENTARY
      KEYSET 1,10  NUMBER   MOMENTARY
```

Figure 4. The Title Delcarations for the Mail Sorting Problem

are written as English-like statements. The hardware functions, like the operator functions, are written as FORTRAN-like statements and describe the mathematical calculations required to support the hardware procedures.

2.4 ACCESSING THE MICRO-MODELS

HOS interprets each HOPROC statement and converts it into a series of operator actions. Every action that the HOS operator performs is a combination of one or more of seven primitive functions. The seven primitive functions are:

(1) Obtaining information;

(2) Remembering information;

(3) Performing a mental computation;

(4) Making a decision;

(5) Moving a body part;

(6) Performing a control manipulation; and

(7) Relaxing.

Although an analyst can write HOPROC statements that will force the operator to perform a particular primitive at a particular point in a sequence of actions, generally, the analyst will let HOS determine the primitives required to accomplish a particular task for itself.

The primitive functions are often either imbedded in, or contain within themselves, human performance models. For example, when a situation arises in which the operator must move his hand to a particular device, there is logic that determines which hand he will use. Similarly, when the operator attempts to recall some item of information, there is a *recall model* that is automatically assessed by the program that simulates the operator's short-term memory processes.

The primitive function calls that result from two HOPROC statements -- the READ statement and the ALTER statement -- are shown in Figures 5 and 6. The READ statement (Figure 5) has the simpler sequence of calls. When the READ statement is encountered, the anatomy movement micro-model is called. Based on the type of display or control to read, the anatomy mover determines which body part is required in order to absorb information from the device. It then computes the amount of time required to bring that body part into contact with the device. The information absorption micro-model is then called to determine how long it will take to read the value of the display or control.

The ALTER statement (Figure 6) has a somewhat more complicated flow. When the statement is encountered, HOS first determines whether the current value of the device is the same as the value to which the device is to be changed. It does this by calling the recall micro-model to determine whether the operator can recall the value of the display or control. If the value cannot be recalled, the anatomy movement and the information absorption micro-models are called to obtain the device value, as with the READ statement. Once the value has been obtained, whether by recall or through the information absorption model, the current value and the desired value are compared. If they differ, then the necessary actions are taken to correct the device value. In the case of controls, the necessary actions require a single call to the anatomy mover and to the control manipulation micro-model. In the case of displays, however, a series of control manipulations may be required. These manipulations must be described in a special type of procedure, called an *adjust* procedure, associated with the display. HOS places the adjust procedure on a list of active procedures, to be executed either when the operator has time or when another procedural statement requires the completetion of the adjustment.

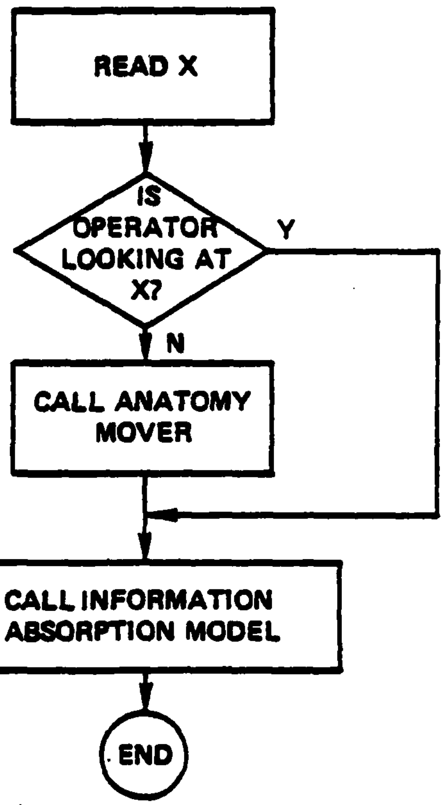


Figure 5. Processing the READ Statement

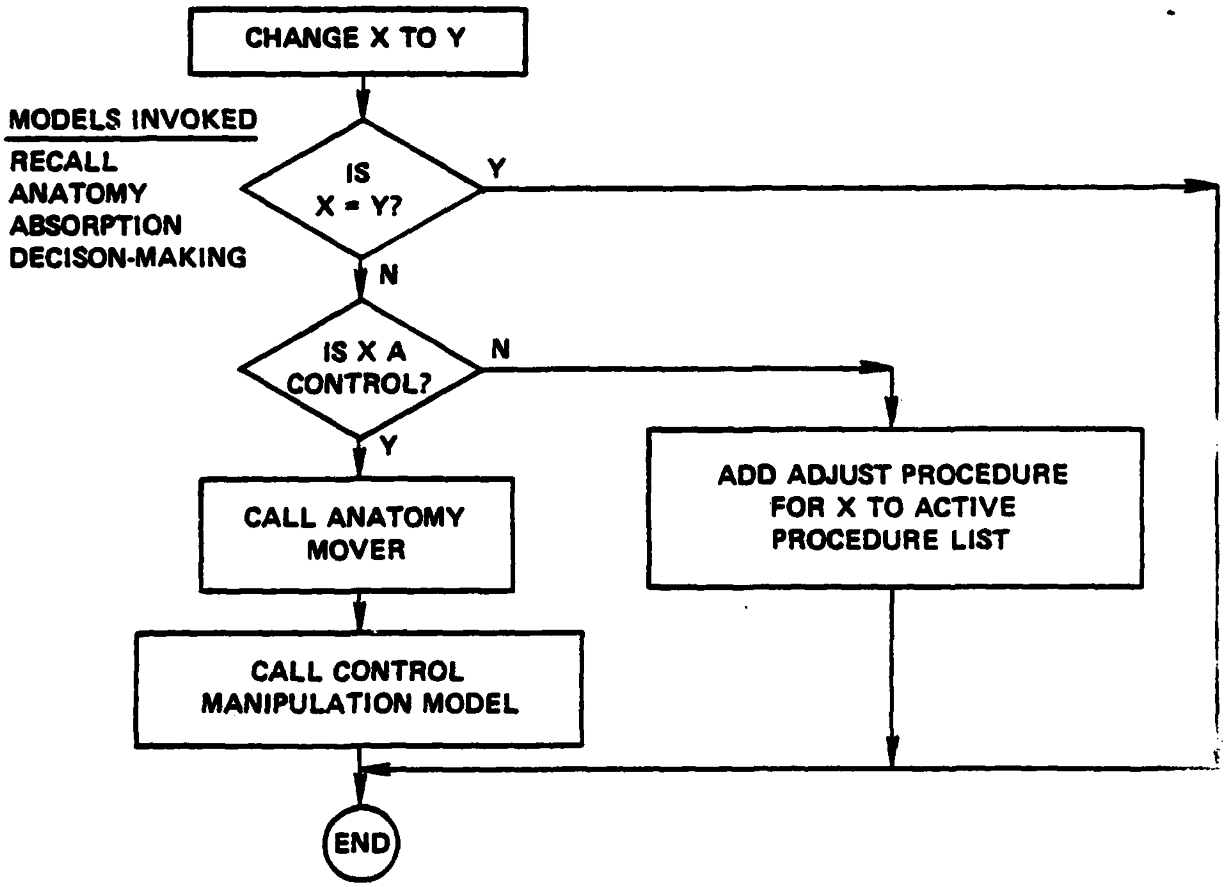


Figure 6. Processing the ALTER Statement

The other micro-models may be accessed as a consequence of other types of procedural statements or on an as-needed basis. For example, if an ALTER statement requires a mental computation to determine the desired value for a device, the mental computation micro-model can be accessed by including the name of the computation in the ALTER statement itself. Alternatively, there is a HOPROC statement (COMPUTE) that is similar to READ except that it invokes the mental computation micro-model rather than the information absorption micro-model.

One characteristic of the mental computation micro-model is that it may require calls to the other micro-models in order to carry out the necessary calculations. For example, a mental calculation may require the value of a displayed quantity. The mental calculation micro-model will automatically initiate calls to the other micro-models -- recall, anatomy movement, information absorption, etc. -- as necessary in order to obtain the required values.

The decision making micro-model is accessed whenever a task statement requires a decision or whenever the operator must make a decision about what procedure to work on next. The statement decisions are expressed as IF... THEN... constructs. The information required to make the decisions is gathered by calls to the recall, anatomy movement, information absorption and mental computation micro-models. Procedural decisions (what procedure to work on next) are based on how long it has been since each procedure was last worked on and how important each procedure is.

The relaxation micro-model functions in parallel with the other micro-models, returning body parts to relaxed locations when they are not being used to carry out procedural statements.

2.5 OPERATOR PROCEDURES FOR THE MAIL SORTING SIMULATION

The HOPROC operator procedures and functions for the mail sorting problem described in Section 1.8 are shown in Figure 7.

Several things should be noted about the code. First, the statements are in an English-like language and are similar to the instructions that one might give an operator who was learning to use the system. Exactly what is expected of the operator is reasonably clear, even without a detailed knowledge of the language. Thus, the HOPROC instructions could almost be used as the basis for training operators in the use of the system.

Second, there are some things missing from the code that might seem to be significant omissions. For example, there is no mention of where the controls are located, what their characteristics are, how many letters the operator will have to process, what the specific zip codes will be, what the actual sequence of button depressions will be, or what any other operating characteristics of the mail sorting machine or the operator are. These factors have been left out of the procedural descriptions because they are not significant to the description of the procedures that the operator must follow. However, since they do drive the results that would be obtained in an actual situation, they are entered later as input data to the simulation through the hardware procedures and functions (Figure 8) and through other direct inputs to the simulation (Figure 9).

2.6 THE OUTPUT FROM HOS

The primary output from HOS is a time history of operator activities. An example of this output for the mail sorting problem is shown in Figure 10. The left hand column lists the simulation time, in seconds, for each of the operator activities listed in the next column. The body parts in use and the hardware procedures being executed are listed in succeeding columns. Although this example is much simpler than the average HOS simulation, it is clear that HOS can provide data to a level of detail unmatched by any other operator simulation technique, including dynamic simulation.

```
OPERATOR PROCEDURES
       DEFINE MISSION.
NEXT:      DEPRESS THE FEED-KEY.
           IF THE AREA-CODE IS 191 THEN
                  DEPRESS THE LOCAL-KEY;
                  KEYSET-ENTER USING THE CITY-ZONE, 2;
                  GO TO NEXT.
           KEYSET-ENTER USING THE AREA-CODE, 3.
           GO TO NEXT.
       DEFINE THE PROCEDURE TO KEYSET-ENTER USING AN AMOUNT,
               NUMBER-OF-DIGITS.
           INITIALIZE THE COUNT.
NEXT:      ADD 1 TO THE COUNT.
           IF THE COUNT IS GREATER THAN THE
                     NUMBER-OF-DIGITS THEN END.
           DETERMINE THE NEXT-DIGIT.
           DESIGNATE IT AS THE KEYSET-NUMBER.
           DEPRESS THE KEYSET-NUMBER.
           GO TO NEXT.


OPERATOR FUNCTIONS
       GO TO 10000
 9000  CONTINUE
       I='NUMBER-OF-DIGITS'-'COUNT'
       I='AMOUNT'/(10**I)
       J=I/10
       J=J*10
       K=I-J
       'NEXT DIGIT'=K
C
       I='ZIP-CODE'/100
       'AREA-CODE'=I
C
       I='ZIP-CODE'/100
       I='ZIP-CODE'-100*I
       'CITY-ZONE'=I
```

Figure 7. Operator Procedures and Functions for the Mail Sorting Problem

```
HARDWARE PROCEDURES
      DEFINE PROCEDURE TO SIMULATE FEED-KEY.
START:    READ A ZIP.


HARDWARE FUNCTIONS
      GO TO 10000
 9000  CONTINUE
      READ(7,100) IZIP
      IF(EOF(7).NE.0) STOP 5
      ACTUAL(<ZIP-CODE>)=IZIP
 100  FORMAT(I6)
      'ZIP'=IZIP
```

Figure 8. Hardware Procedures and Functions for the Mail Sorting Problem

```
SYSTEM                                        MAIL SORTING -- CASE A
METRIC                    0 25 -50
DISPLAY SECTION
AREA CODE                 1 0 1 1 -.7 20 10 0
CITY ZONE                 3 0 1 1 .7 20 10 0
CONTROL SECTION
LOCAL KEY                 2 0 1 1  5 -3 0 0
FEED KEY                  2 0 1 1 -5 -3 0 0
KEYSET 1 NUMBER           2 0 1 1 -2 -2 0 1
KEYSET 2 NUMBER           2 0 1 1  0 -2 0 2
KEYSET 3 NUMBER           2 0 1 1  2 -2 0 3
KEYSET 4 NUMBER           2 0 1 1 -2  0 0 4
KEYSET 5 NUMBER           2 0 1 1  0  0 0 5
KEYSET 6 NUMBER           2 0 1 1  2  0 0 6
KEYSET 7 NUMBER           2 0 1 1 -2  2 0 7
KEYSET 8 NUMBER           2 0 1 1  0  2 0 8
KEYSET 9 NUMBER           2 0 1 1  2  2 0 9
KEYSET 10 NUMBER          2 0 1 1  0 -4 0 0
OPERATOR FUNCTIONS
NUMBER OF DIGITS          7 0 1 1 .04 0
NEXT DIGIT                7 0 1 1 .04 0
MODEL SPECIFICATIONS
 1.AREA CODE DISPLAY          5 1 0 .04 2 3
 2.MOMENTARY CONTROL      7 3 0 .04 1 3 0 .1
 3.CITY ZONE DISPLAY      5 1 0 .03 1.5 3
END OF MODEL SPEX
HUMAN OPERATOR SPEX
EYES                          0  20  10  .03  84  122
HANDS                        -5  -3   0  -20  -3    0
SHOULDERS                  15.25 0 -15.25  -15.25  0  -15.25
END OF HUMAN SPEX
/EOR
 19126
 45321
 89674
 15222
 80231
 12345
 67529
```

Figure 9. Crewstation Input Data for the Mail Sorting Problem

```
                                              MAIL SORTING -- CASE A       BODY
                                                                     RH LH RF LF  E     HARDWARE
OPERATOR                                                             --------------     --------
--------

      .00 MISSION
      .00      STEP NEXT
      .00      ALTER    2                                            X
      .00      MANIPULATE FEED KEY                                                      $FEED KEY

      .00      IF    3
      .00      COMPUTE AREA CODE                                                  X
      .00      ABSORB ZIP CODE                                                           19126.0
      .24          ZIP CODE            =  19126.0
      .28          AREA CODE           =    191.0
      .29      ALTER    6                                          X
      .67      MANIPULATE LO KEY
      .67 KEYSET ENTER
      .67      STEP NEXT
      .67      IF   29
      .68      COMPUTE NEXT DIGIT
      .68      COMPUTE CITY ZONE                                                  X
      .68      ABSORB ZIP CODE                                                           19126.0
      .72          ZIP CODE            =  19126.0
      .76          CITY ZONE           =     26.0
      .76      COMPUTE NEXT DIGIT
 DEFAULT HARDWARE PROCEDURE EXERCISED FOR LOCAL KEY
      .80          NEXT DIGIT          =      2.0
      .80      ALTER   36                                            X
     1.08      MANIPULATE KEYSET 2 NUMBER
 DEFAULT HARDWARE PROCEDURE EXERCISED FOR KEYSET 2 NUMBER
     1.18      STEP NEXT
     1.18      IF   29
     1.19      COMPUTE NEXT DIGIT
     1.19      COMPUTE CITY ZONE                                                  X
     1.19      ABSORB ZIP CODE                                                           19126.0
     1.23          ZIP CODE            =  19126.0
     1.27          CITY ZONE           =     26.0
     1.27      COMPUTE NEXT DIGIT
     1.31          NEXT DIGIT          =      6.0
     1.31      ALTER   36                                            X
     1.54      MANIPULATE KEYSET 6 NUMBER
 DEFAULT HARDWARE PROCEDURE EXERCISED FOR KEYSET 6 NUMBER
     1.64      STEP NEXT
     1.64      IF   29
     1.65 END KEYSET ENTER
     1.65 MISSION
     1.65      STEP NEXT
     1.65      ALTER    2                                            X
     1.99      MANIPULATE FEED KEY                                                      $FEED KEY

     1.99      IF    3
     1.99      COMPUTE AREA CODE                                                  X
     1.99      ABSORB ZIP CODE                                                           45321.0
     2.03          ZIP CODE            =  45321.0
     2.07          AREA CODE           =    453.0
     2.08 KEYSET ENTER
```

Figure 10. The Mail Sorting Problem

Finally, in addition to the raw timeline data produced by HOS, statistical analysis routines developed specifically for use with HOS can produce a variety of composite statistics. Examples of the timeline and channel loading analysis are shown in Figures 11 and 12. Other statistical analyses available include device usage statistics and link analyses.

Figure 11. HODAC BODY PART T' LINE ANALYSIS (.5 SECOND SNAPSHOTS)

TIME	EXECUTING	EYES ARE	RIGHT HAND IS	LEFT HAND IS	RIGHT FOOT IS	LEFT FOOT IS
.0	MISSION	ABSORBING FROM ZIP CODE	MOVING TO LOCAL KEY			
.5	KEYSET ENTER	. . . .	MOVING TO KEYSET 2 NUMBER			
1.0	. . . .	. . . .	MOVING TO KEYSET 6 NUMBER			
1.5	MISSION	. . . .	MOVING TO FEED KEY			
2.0	KEYSET ENTER	. . . .	MOVING TO KEYSET 4 NUMBER			
2.5	. . . .	. . . .	MANIPULATING KEYSET 5 NUMBER			
3.0	. . . .	. . . .	MOVING TO KEYSET 3 NUMBER			
3.5	MISSION	. . . .	MOVING TO FEED KEY			
4.0	KEYSET ENTER	. . . .	MOVING TO KEYSET 8 NUMBER			
4.5	. . . .	. . . .	MANIPULATING KEYSET 9 NUMBER			
5.0	. . . .		MOVING TO FEED KEY			
5.5	. . . .	ABSORBING FROM ZIP CODE	MOVING TO KEYSET 1 NUMBER			
6.0	. . . .	. . . .	MOVING TO KEYSET 5 NUMBER			
6.5	. . . .	. . . .	MANIPULATING KEYSET 5 NUMBER			
7.0	MISSION	. . . .	MOVING TO FEED KEY			
7.5	KEYSET ENTER	. . . .	MOVING TO [illegible]			

MAIL SORTING -- CASE A PAGE 1

MODAC CHANNEL LOADING REPORT (.5 SECOND SNAPSHOTS)

TIME	EYES		MENTAL		RIGHT HAND		LEFT HAND	RIGHT FOOT	LEFT FOOT
.50	48.0	*****	48.0	*****	62.0	******	.0	.0	.0
1.00	8.0	*	16.0	**	64.8	******	.0	.0	.0
1.50	8.0	*	16.0	**	42.3	****	.0	.0	.0
2.00	2.2		2.2		67.0	*******	.0	.0	.0
2.50	13.8	*	21.8	**	53.0	*****	.0	.0	.0
3.00	8.0	*	16.0	**	51.0	*****	.0	.0	.0
3.50	8.0	*	16.0	**	49.0	*****	.0	.0	.0
4.00	16.0	**	20.1	**	56.9	******	.0	.0	.0
4.50	8.0	*	11.9	*	62.8	******	.0	.0	.0
5.00	8.0	*	22.6	**	36.7	****	.0	.0	.0
5.50	.0		1.4		73.6	*******	.0	.0	.0
6.00	16.0	**	24.0	**	52.2	*****	.0	.0	.0
6.50	8.0	*	16.0	**	42.3	****	.0	.0	.0
7.00	8.0	*	16.0	**	37.5	****	.0	.0	.0
7.50	9.2	*	9.2	*	73.8	*******	.0	.0	.0
8.00	6.8	*	14.8	*	54.2	*****	.0	.0	.0
8.50	8.0	*	16.0	**	51.0	*****	.0	.0	.0
9.00	8.0	*	16.0	**	51.0	*****	.0	.0	.0

Figure 12. Channel Loading Analysis

3. THE HOS OPERATOR MODELS

3.1 INFORMATION ABSORPTION

3.1.1 Absorption Modalities

The HOS operator has three modalities by which he can obtain information -- his eyes, hands, and feet. Currently, neither hearing nor speech nor any kinesthetic cues, such as vibration, balance, or the perception of external motions are simulated because of the unavailability of any satisfactory models for the effects that these factors have on an operator's performance that could be used in HOS.

When describing the displays and controls in the operator's crewstation, the analyst must identify the modality (eyes, hands, or feet) that the operator is to use when obtaining information from each device. Thus, if the analyst were describing the displays and controls in an automobile, he would indicate to HOS that the operator is to use his eyes to read the fuel guage, his hands to "read" the steering wheel, and his foot to "read" the accelerator.

The process by which the operator obtains information is the same for each modality and consists of a series of *micro-absorptions*. Each micro-absorption requires time. As the operator spends more time (more micro-absorptions) reading a device, both his knowledge of the device's value and his confidence in that knowledge increase until his confidence exceeds a threshold, at which time the absorption process is terminated.*

*Several other conditions may cuase an absorption to be terminated, as will be discussed below.

3.1.2 Absorption Hab Strengths

The quantity that represents the operator's confidence in his knowledge of the value of a device is termed *hab strength*, after the learning theory concept called "habit strength" by Clark Hull. Each device has an associated hab strength that builds up during absorption. As the operator spends more time absorbing information, the hab strength associated with that information increases until it exceeds a threshold value, at which point absorption is terminated.

As an example, assume that the operator is attempting to read a device (e.g., a warning light) that has two discrete settings -- on and off. Successive micro-absorptions will cause the hab strength to increase as shown in the upper dashed curve in Figure 13. Within 3-4 micro-absorptions, the operator would have established to his satisfaction whether the display is on or off and the absorption process would be terminated. Using a micro-absorption time charge of .04 such a read operation would require .12-.16 seconds* For a display that is more difficult to read, more micro-absorptions are required to reach a comparable hab strength, as in the second dashed curve for which micro-absorption time charge was .12. Similarly, displays that have more potential values -- i.e., displays with more settings or continuous displays -- require still more micro-absorptions in order to reach a comparable hab strength. The bottom two curves shown in Figure 13, for example, represent the increase in hab strength for two continuous displays with micro-absorption time charges equal to those of the two displays in the upper curves. It should be noted that the equations used for continuous displays are the same as those for discrete displays with seven or more settings -- i.e., discrete displays with more than seven settings are treated as if they were continuous.

Figure 14 shows the effect that the micro-absorption time charge can have on the amount of time spent in a single, complete (macro-) absorption. The four curves represent the same four displays as in the preceding figure.

*As compared to an average rate for reading words of .17-.24 seconds per word based on a reading rate of 250-350 wpm.

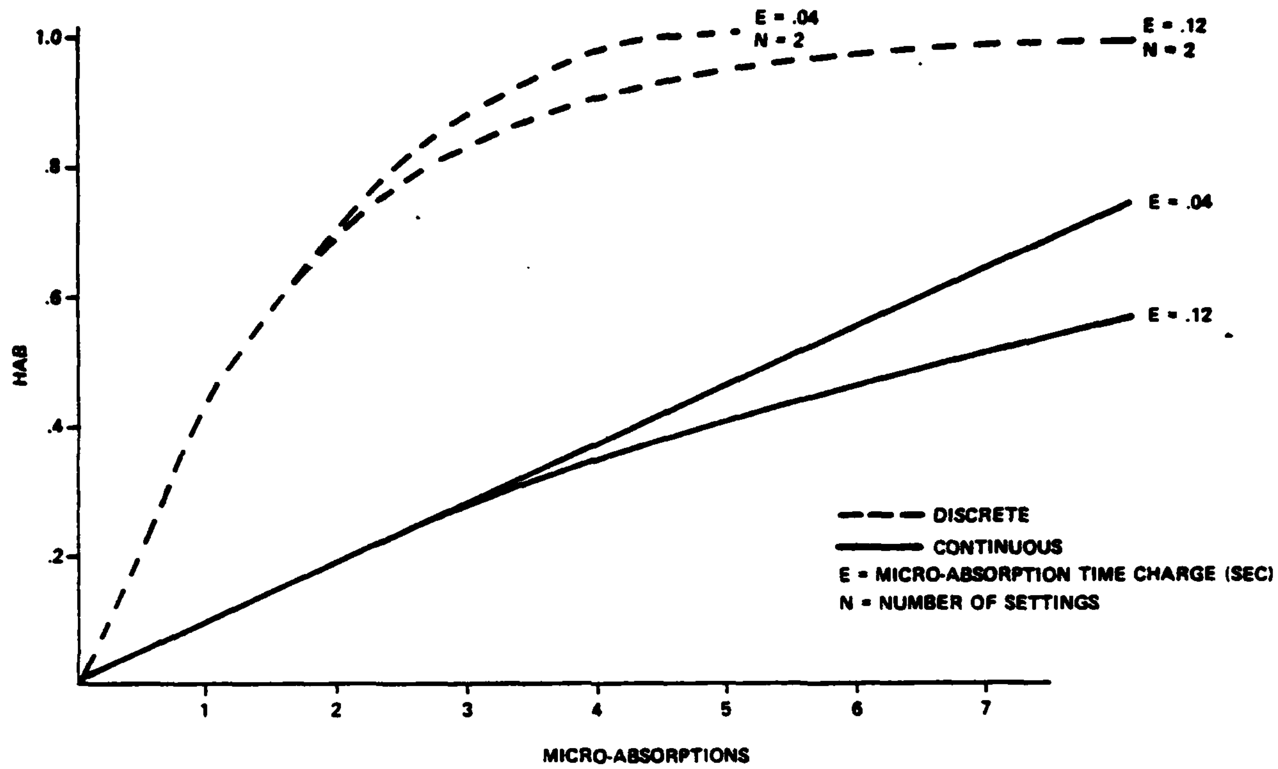


Figure 13. Hab strength as a function of the number of micro absorptions.

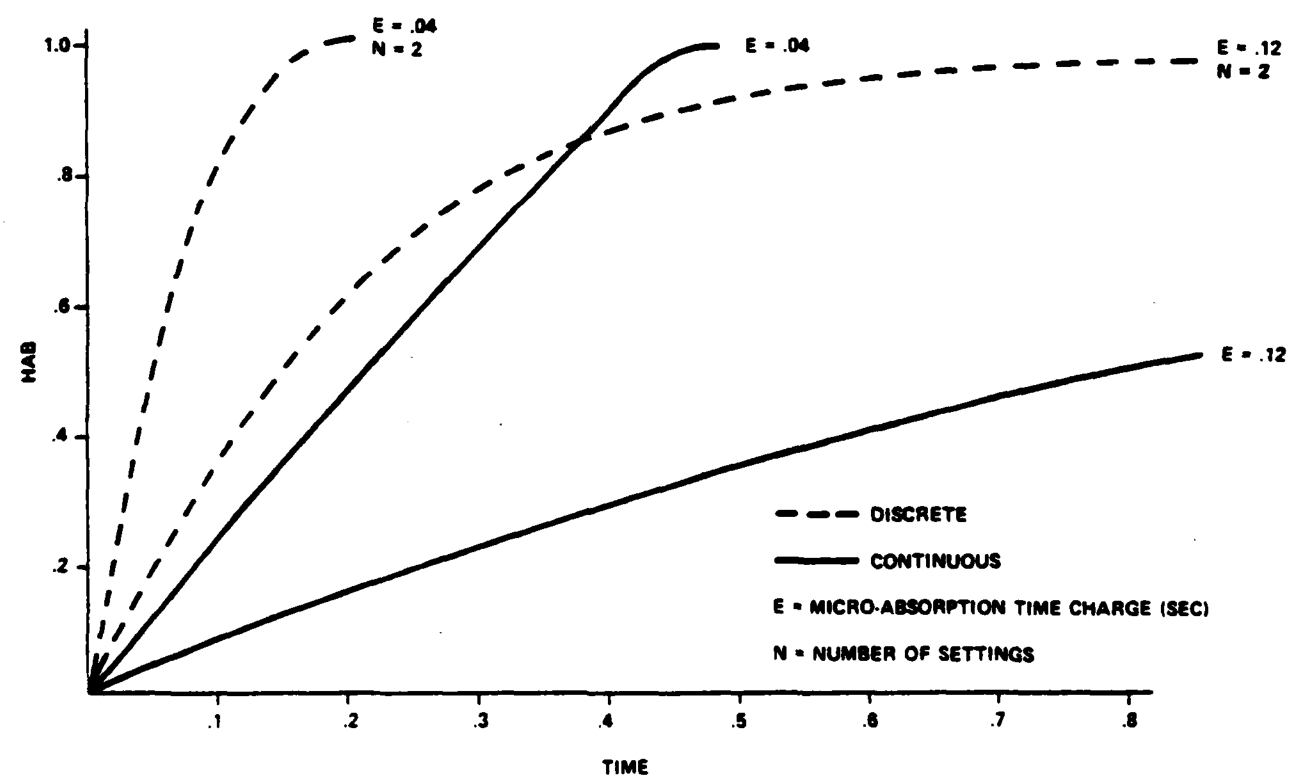


Figure 14. Hab strength as a function of absorption time.

In Figure 14, however, it can be seen that if the operator spends as much as .4 seconds in the absorption process, the hab strength for the "easy-to-read" continuous display will exceed the hab strength for the difficult discrete display.

The primary criterion for terminating an absorption process is for the hab strength of the device value being absorbed to exceed a threshold value, but there are several other conditions that the analyst can impose as input parameters that will terminate absorption. These conditions are:

- A maximum amount of time to be spent absorbing.
- A maximum number of micro-absorptions.
- A tolerance value that specifies that the hab strength has reached an asymptote.
- A tolerance on the accuracy to which the operator is requried to read any device -- after which he is considered to "know" the value of the device.

The interaction between these termination conditions are discussed in more detail in Strieb, Glenn, and Wherry (1978).

3.1.3 Absorption Estimates and Errors

During the absorption process, the operator acquires knowledge about the value of a device and confidence in his knowledge of that value. The value that the operator believes a continuous device to have (the *estimated* value of the device) is determined from the *actual* value of the device by adding an error term that is normally distributed about the actual value and whose magnitude is dependent upon an *accuracy* for the device as supplied by the analyst. Thus, if the analyst has specified that a particular device can be read to an accuracy of two percent, then two percent of the actual value of the device is used as the standard deviation when computing the value that the operator believes the device to have on any specific absorption.

3.2 INFORMATION RECALL

3.2.1 Long-Term and Short-Term Memory

The HOS information recall model consists of two submodels -- a model for short-term memory and one for long-term memory. The long-term memory model is currently limited to the recall of certain types of predetermined information. Specifically, the HOS operator is assumed to have a completely *accurate* and *instantaneous* recall of the *locations* of all the displays and controls in his crewstation, most of their *characteristics*, the *procedures* that must be followed in carrying out a job, and the *calculation processes* for any mental computations that must be performed. These assumptions are consonant with the basic assumption of the HOS model -- namely, that the operator being simulated is a *trained* operator who performs routine operations automatically.

The short-term memory model is more elaborate. Short-term memory is considered to be linked to perception through the hab strength concept. As explained above, during the process of absorbing information, the operator's confidence in his knowledge of the value of a device increases until it exceeds a threshold at which point the absorption process is terminated. The ultimate hab strength associated with the device, a value between zero and one, constitutes a measure of the operator's confidence in his knowledge of the device value.

During recall, the hab strength is used to determine the probability that the operator will recall information absorbed from a device. The probability of successful recall is given by:

$$P = H^{\sqrt{t}}$$

where H is the hab strength and t is the time in seconds since the last absorption. Since H is a value between zero and one, the probability of recall is one at time zero -- i.e., the operator has an instantaneous memory of the value of a device that is perfect, to the extent that he has learned the information in the first place. One second after the completion of an absorption, the probability of recall is exactly equal to the hab strength. As soon as absorption is complete, the probability of recall begins to decay exponentially as shown in Figure 15 . Thus, within 60 seconds after an absorption that had raised the hab strength to .7, the probability of successful recall would be less than .1. If, however, the hab strength had been raised to .9, the probability of recall would stay above the level .1 for approximately seven minutes. Figure 15 shows recall probabilities from some of the available experimental data on short-term memory and how these data correspond to various hab strength values. Based on these data, we have chosen .8 as the default value for the hab strength threshold -- the value that is used to determine when the absorption process is terminated.

The value of P from the probability of recall equation:

$$P = H^{\sqrt{t}}$$

is compared against a number drawn at random from a uniform distribution. If the randomly drawn number is less than P, then the information is "remembered." If the randomly drawn number is much larger than P, then the information is "forgotten." If, however, the randomly drawn number is close to P, then the model assumes that the operator is in a region of "near-recall," where given a little more time, he might remember. A second random number is therefore drawn and compared with P to determine whether the information is remembered, forgotten, or in the near-recall region. Usually a second draw will suffice -- the random number will either be in the remembered or forgotten region. But the process could theoretically go on for three or more tries. Each try results in the addition of a small amount of time, the *short-term memory cycle time*, to the total time for retrieval from short-term memory (Figure 16).

When the operator recalls a value, the hab strength associated with that value is changed in order to simulate the effects of *rehearsal*. The

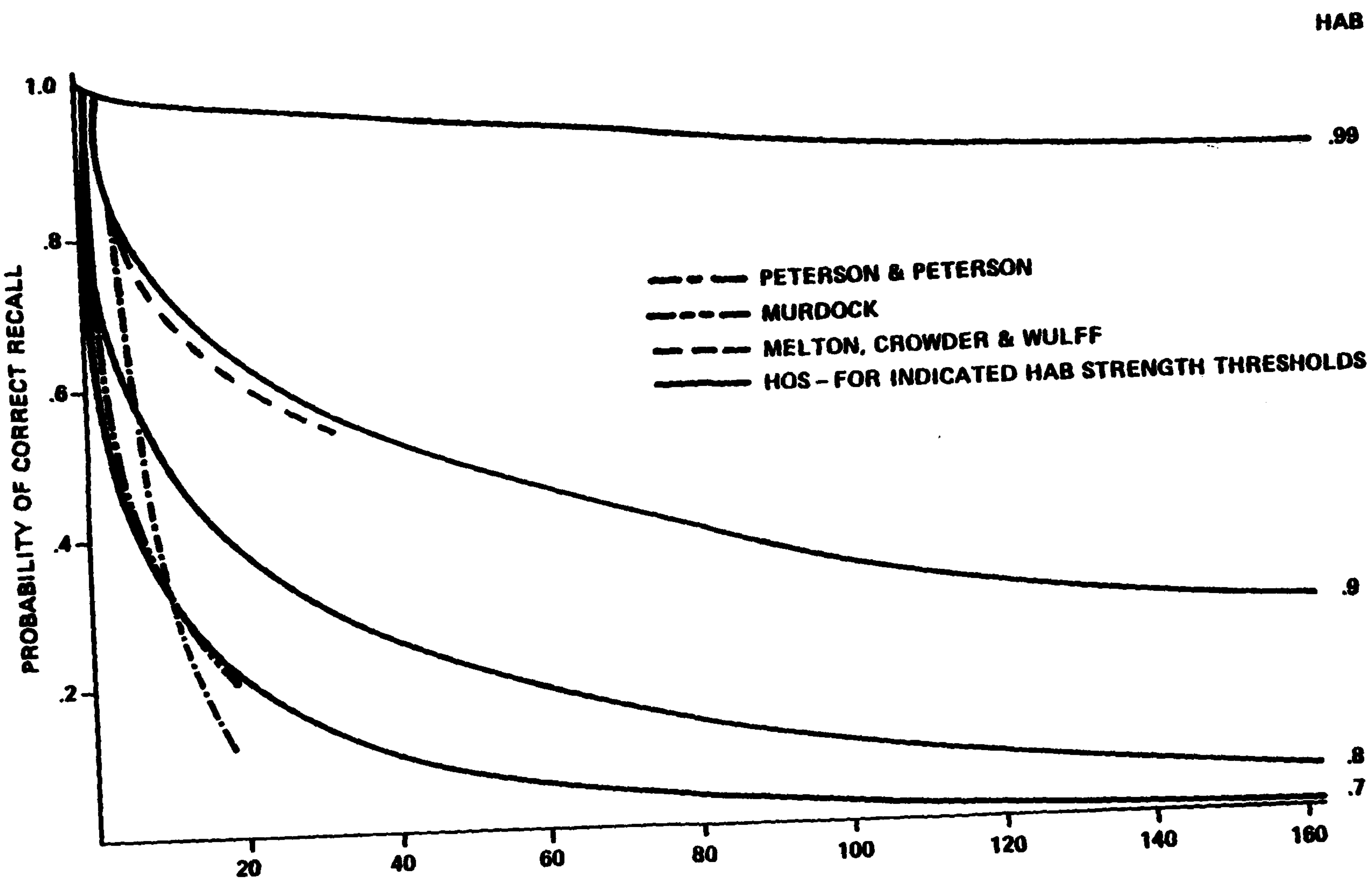


Figure 16. Experimental Data on Short-term Memory Compared with HOS Hab Strength Recall Probabilities

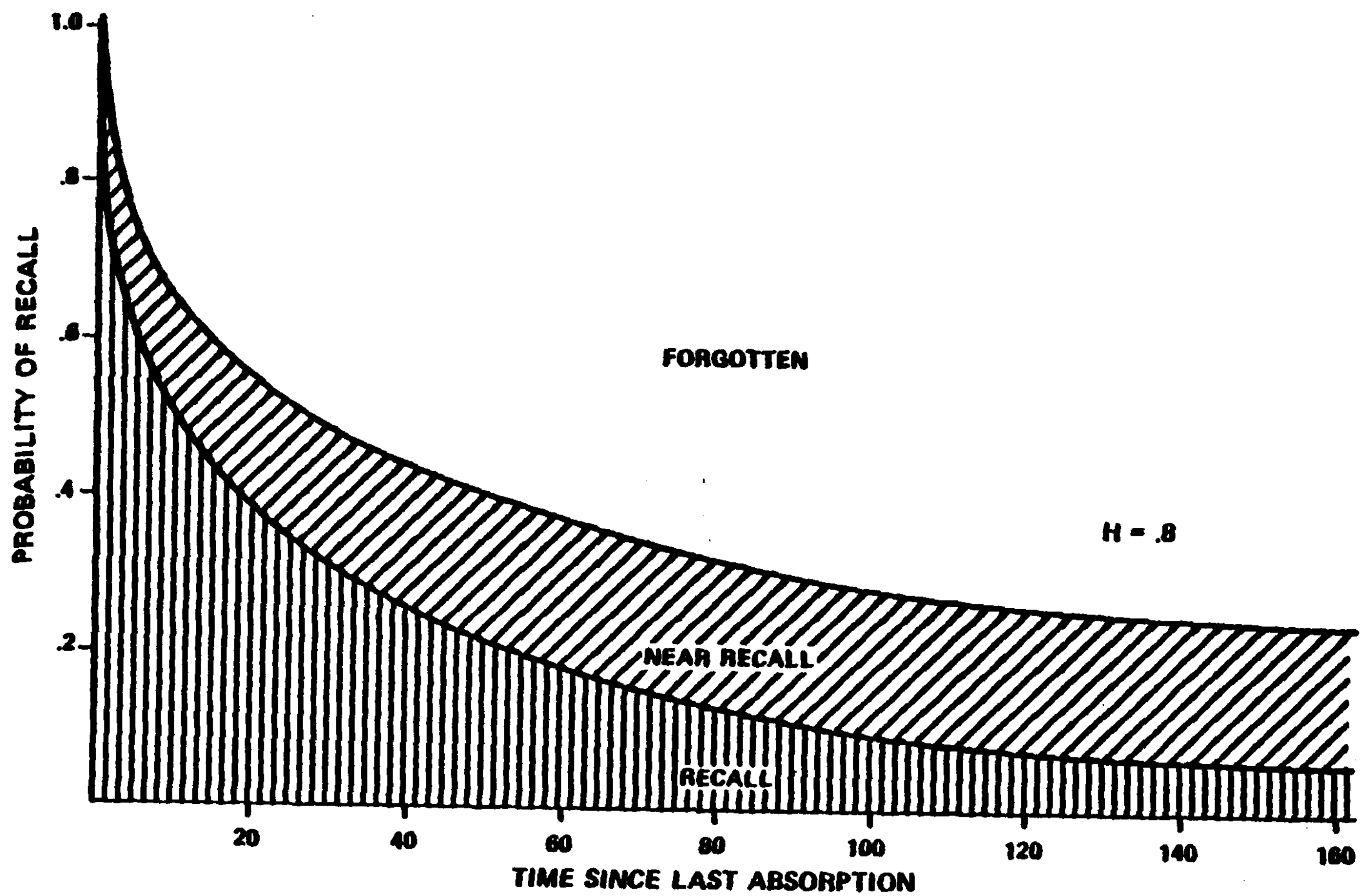


Figure 18. Short Term Recall Regions

remembered value is given a hab strength that is lower than if the information had been absorbed again, but higher than it would have been had the normal decay curve been followed.

There are several features of this recall model that deserve some comment (and probably some future work). First, the process by which the hab strength associated with any item of information is increased and recalled is independent of the value of information to the operator -- the threshold value is the same for all information and consequently all items of information follow essentially the same curves for the increase and decrease in hab strength. This is clearly unrealistic -- information that is of greater value to the operator should decay less rapidly and should be learned to a higher level of confidence than less important information. Secondly, the recall model has no explicit provision for allowing information to transfer from short-term memory to long-term memory, though there is an effective transfer that results from rehearsal for the real human operator. Third, there is no linkage between items of information -- if, for example, the operator depresses a switch that changes the value of a display, that action will normally not affect the value that the simulated operator will recall for the display, whereas a true operator would certainly know that the displayed value had changed.* And fourth, there are no external cues that impact the perceived or recalled value of a device, as the view out the window might cue the recall of the altimeter value for an aircraft pilot.

3.2.2 Errors During Recall

For continuous devices, there is a portion of the recall model that simulates the decreased accuracy associated with the recalled value. The basic premise behind this feature of the recall model is that as confidence (i.e., hab strength) in the value of a device decreases, the precision of the value that the operator recalls for the device will also decrease. Thus, if at some later time, the operator is asked for the value of the device, then the operator will be able to supply fewer "significant digits" as the

*Although the analyst can, in fact, specify that such linkages exist when coding the procedures.

time from the last absorption of the value of the device increases. We term this process *modular decay*. The modular decay function is such that given an initial device value of, for example, 123456 and an initial hab strength of .8, the modularly decayed values would be as shown in Figure 17.

3.2.3 Extrapolation of Values

If the operator can recall the value that a continuous device had the last time he read it, then HOS enables the operator to extrapolate its value to the current time. The extrapolation is linear and based on the two preceding absorbed values and the times when those values were obtained. It is the responsibility of the HOS user to declare whether or not extrapolation is to be permitted for each device.

3.2.4 Scope of the Information Absorption and Recall Models

The estimated value of a device is the only characteristic of a device that is either recalled or read by the HOS operator. The operator *does*, however, maintain other information on other device characteristics -- desired values, upper limits, lower limits, etc. -- but these quantities (termed device *parameters*) are considered to be resident in the operator's long-term memory and therefore are not subject to the information absorption/recall processes. The various device parameters are listed in Figure 18.

3.3 MENTAL COMPUTATION

The mental calculations performed by the HOS operator are termed *operator functions*, or simply *functions*.

The mental calculation micro-model uses the hab strength construct in much the same way as the information absorption and information recall micro-models. The result of a mental computation has an associated hab strength that represents the operator's confidence in the computed data. As the operator spends more time on the computation, his confidence in his estimate increases until either:

TIME SINCE ABSORPTION (T)	VALUE RECALLED IF RECALL SUCCEEDS	PROBABILITY THAT RECALL WILL SUCCEED (P)	TIME SINCE ABSORPTION (T)	VALUE RECALLED IF RECALL SUCCEEDS	PROBABILITY THAT RECALL WILL SUCCEED (P)
0	123456	1	0	123.456	1.0
1	123455	.8	1	123.455	.8
20	123000	.37	20	123.000	.37
40	125000	.24	40	125.000	.24
60	125000	.18	60	125.000	.18
80	125000	.14	80	125.000	.14
100	125000	.11	100	125.000	.11
120	125000	.09	120	125.000	.09
140	125000	.07	140	125.000	.07
160	125000	.06	160	125.000	.06

Figure 17. Modular Decay Examples

- DESIRED VALUE
- RATE OF CHANGE
- TIME (OF LAST ESTIMATE)
- X AND Y COMPONENTS
- UPPER AND LOWER LIMITS
- CRITICALITY
- STATE (ACTIVE OR INACTIVE)
- ESTIMATED VALUE

Figure 18. Device Parameters

- The hab strength threshold is exceeded.

- The hab strength has asymptoted.

- The maximum number of interactions through the hab strength incrementing process has been exceeded.

- The amount of time spent in computation exceeds a maximum allowable computation time.

The recall model for mentally computed data is identical to the model used for any other type of data.

The basic difference between the mental computation model and the information absorption model is that, in the latter, information is absorbed from a display or control in the crewstation, whereas in the former, displayed information is used to determine a value that is not displayed anywhere in the crewstation. For example, a typical mental computation when driving an automobile is determining how much farther one can go on a tank of gas. The computation requires the absorption of an item of information (the amount of fuel remaining) coupled with some prior knowledge (the number of miles per gallon).

When a mental calculation is required, HOS will determine what information is needed in order to perform the calculation. If the HOS operator can remember the information, the calculation is performed at once. If he cannot remember the information, an appropriate sequence of actions is initiated to enable the operator to obtain the data. In the above example, the displayed information required is the amount of fuel remaining. If the operator cannot remember this, HOS would cause him to look at the fuel gauge and read its value.

Each mental calculation can require as many as ten different data items. These may be the values of displays or controls or the results of other mental calculations. An unlimited number of parametric values are also allowed. The amount of time required for a mental calculation is considered to be the amount of time required to gather all the items of information needed for the calculation plus some additional time to "put it all together." Because of the high potential variability in a function calculation, the analyst is required to supply a function computation time for each function -- HOS itself will supply the times required to gather all the items of information needed for the calculation.

A second difference between the mental computation and information absorption models is that in the information absorption model, the minimum hab strength associated with a device is dependent on the number of settings associated with the device. In the case of mental computations, the hab strength associated with the operator function is the minimum hab strength associated with any of the components in the function claculation.

Errors in mental computation are assumed to be the result of errors associated with the data that goes into the calculation itself. The calculation process itself is considered to be error-free. Thus, if the operator makes an error or obtains an inaccurate data value when either recalling the data or reading the data needed for a calculation, then the result of the calculation will be incorrect, or inaccurate, according to the incorrectness or inaccuracy of the incoming data. If the data values are correct and accurate, then the result of the calculation will be accurate. It should be noted, however, that, as a result of the way in which mental calculations are described to HOS, the analyst has the ability to inject errors into the function calculation if he so chooses.

3.4 MAKING A DECISION

HOS decision-making takes place at two levels -- the inter-procedural and the intra-procedural levels. A *procedure* is an *operator task consisting of any number of steps*, any step of which can invoke the execution of another procedure or any other operator action. For example, the operator's *mission* in any particular simulation is a procedure that invokes other procedures -- a pilot's mission may invoke a procedure for takeoff, another for cruise, another for landing, etc. Within these procedures (or any procedures that they invoke) there are steps that describe operator actions -- reading a display, adjusting a control, etc. Decision-making can therefore operate at two different levels -- deciding what procedure to perform from a set of available *active* procedures, or deciding what to do *next* within any particular procedure.

HOS gives the analyst the option of both limited and total control over these decisions. The analyst can opt for *total control* in the sense that simulations can be constructed that *force* the operator to follow a specific sequence of steps and procedures. The analyst can, instead, choose *limited control* in the sense that the exact sequence of task and subtask operations that an operator will use is unknown -- the simulation can be constructed so that the HOS operator is allowed to make decisions for himself in accordance with a flexible task structure. Such a flexible structure is appropriate because, like a real operator, HOS can adapt its actions to situations. The power of HOS lies in its ability to adapt its performance to situations in a natural and realistic fashion.

Decisions about what to do next *within* a procedure are fairly simple. HOS will attempt to execute each step in a procedure in sequence until it can go no further, for whatever reason. If it finds itself blocked, it will attempt to "unblock" itself. If it can, it will continue with the next procedural step; if it can't, it will look for some other procedure to work on, at which point the decision-making logic for selecting a procedure is invoked.

As it executes the steps in a procedure, HOS may encounter a *statement* that requires a decision, i.e., an IF statement. The IF statement requires the operator to make a decision about the current status of information or events in the simulation. If the condition(s) tested is (are) satisfied, then it proscribes a set of actions to be taken. If the condition(s) is (are) not satisfied, the actions are not performed. A small time charge is assessed for this decision-making function over and above the time charges associated with gathering the information needed for the decision.

There are basically three types of events that will block the operator:

(1) An action is required that the operator cannot perform because the action requires body resources that are busy doing something else,

(2) The operator requires information that is currently unavailable because a device is inactive (not enabled), or

(3) The operator must perform a control action that cannot be performed because the control is inactive (not enabled).

Of these situations, the latter two are the more common. When they occur, HOS will automatically invoke a special type of procedure -- an *enable* procedure -- whose function is to activate the device that is inactive. When the first situation occurs, HOS will simply go off and work on another procedure until the required body part is free.

One of the actions that can be performed within a procedure is the invocation of another procedure. When a procedure is invoked, the analyst can specify either that:

(1) The procedure is to be executed immediately and no more steps in the current procedure are to be executed until the invoked procedure has been completed, or

(2) The invoked procedure is to be placed on an *active procedure list* and is to be executed as soon as appropriate, or

(3) The invoked procedure is to be executed periodically until removed from the active procedure list.

In situation (1), control transfers immediately to the invoked procedure and no more steps in the invoking procedure are executed until the invoked procedure is completed. The active procedure list, formed by invoking procedures by methods (2) and (3), is the list of procedures available to the operator when he finds himself blocked in his current procedure. Procedures placed on the active procedure list by method (3) are called *monitor* procedures in that they are usually used to cause the operator to periodically monitor a particular display or control.

Finally, the analyst can force a procedure to be selected from the active procedure by using special forms of the IF and GO TO statements.

When a procedure is to be selected from the active procedure list, a model that represents the operator's procedural selection process is invoked. This model considers two factors:

(1) The criticality (priority) of the procedure, and

(2) How long it has been since the procedure was last executed.

A detailed discussion of the interaction of these factors is presented in Strieb, Glenn, and Wherry (1978). Briefly, as the length of time since the procedure was last executed increases, the *effective criticality* of the procedure increases (over an initial criticality that can be set by the analyst), as shown in Figure 19. In addition, for monitor procedures, the effective criticality is further modified by a factor that is dependent on how close the device being monitored is to a defined set of limits. As the estimated value of the device approaches its limits, the effective criticality of the device increases. When it exceeds the defined limits, the effective criticality increases very rapidly, as shown in Figure 20. The computed effective criticalities for each procedure on the active procedure list are compared and the procedure with the highest effective criticality is chosen as the next procedure to work on.

3.5 ANATOMY MOVEMENT

The anatomy movement micro-model is almost always accessed implicitly -- i.e., the analyst will rarely issue a command that will force a body movement. Rather, HOS itself will determine whether a body movement is required in order to accomplish the objective of an instruction. If it decides that a body movement is required, HOS will automatically select the appropriate body part, move it to the required location, and add to the simulation time a computed estimate of the amount of time the action would have taken a real operator. For example, suppose a procedural statement says:

TURN SWITCH-A ON.

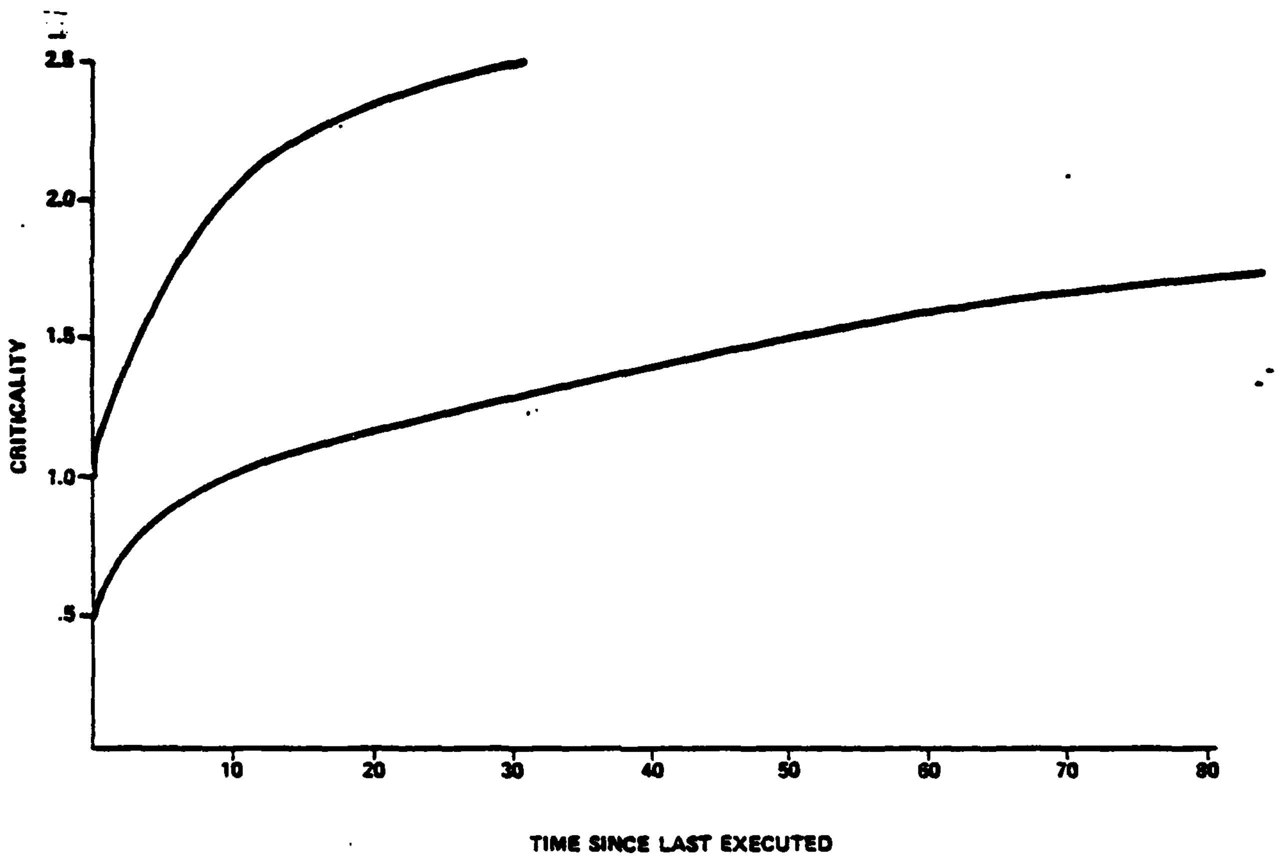


Figure 19. Increase in Procedural Criticality with Time

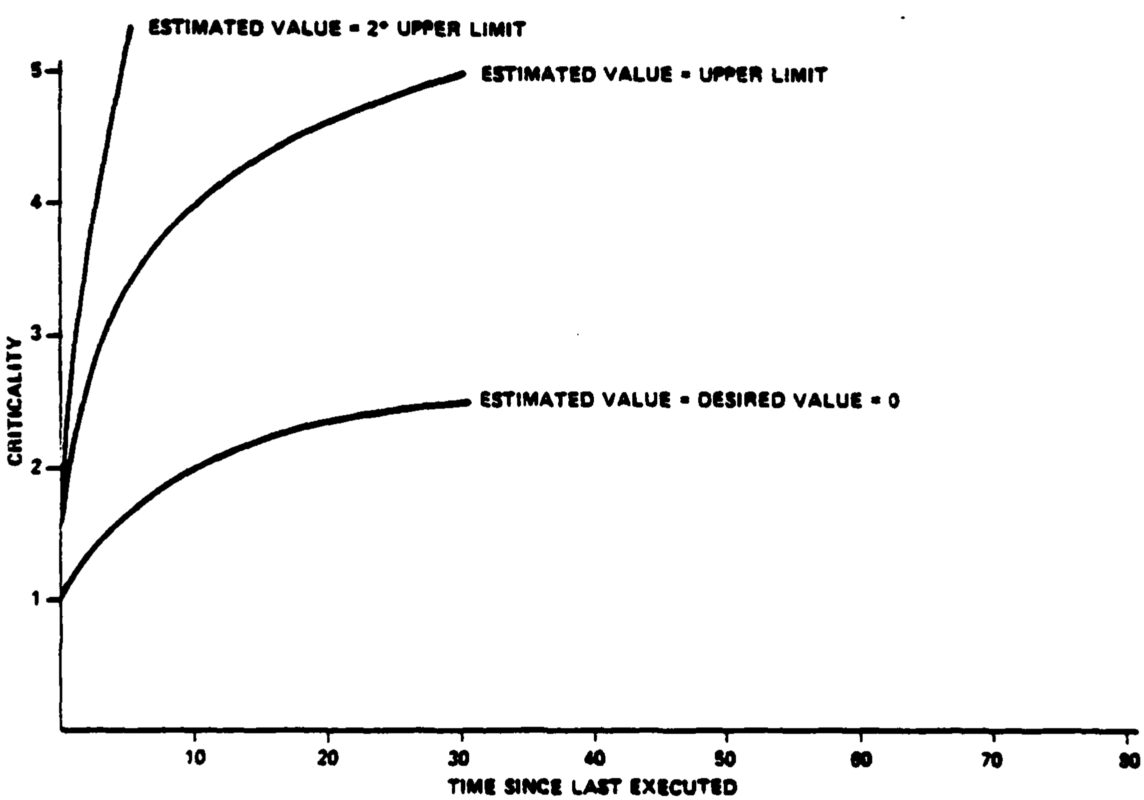


Figure 20. Increase in Criticality for Monitor Procedures

If HOS decides that this action is necessary,* and if one of the operator's hands is not already on SWITCH-A,** HOS will select a hand, "move" it to SWITCH-A, and charge and amount of time equal to the time that a real operator would have taken to move that hand to SWITCH-A from wherever the hand was at the time the instruction was issued.

Thus, the moving and grasping primitive function consists of two micro-models -- one to determine which body part to use for a particular action, the other to assign a time charge for the movement. The body part selection micro-model is based on several common-sense principles. The first is that the body part to be used is determined by the function to be performed and the device being referenced. Thus, if the operator is going to be reading data from a device, the eyes are usually the appropriate body part to use. However, there may be some devices whose value cannot be determined visually -- touching them with a hand or foot may be more appropriate. Some devices may use two modalities -- the eyes are used to absorb information while the hands are used when the device is to be altered. HOS permits the analyst to specify for each device the most appropriate modality for each function (reading and/or altering).

If the operator's eyes are to be used for a specific function, there is no problem -- the HOS operator has only one pair of eyes which are immediately moved to the device. The time charge assigned for an eye movement is computed from an equation that was developed by fitting the data from an experiement that involved lateral eye movements (Dodge and Cline, 1901) and from an unpublished experiment by Wherry and Bittner that involved both lateral and convergence movements.

---

*SWITCH-A may be ON. A real operator, if he remembered this, would not perform the action. Similarly, HOS would decide whether the simulated operator remembered whether the device was on and, if he did, would not initiate the body movement.

**Assuming that SWITCH-A is a device that is turned on by hand.

The equations used are:

$$T = .14324\,A + .0175$$

where

$$A = \max(\Delta\theta, \Delta\phi) + .2 \min(\Delta\theta, \Delta\phi)$$

and

$$\Delta\phi = \left| \tan^{-1}\left(\frac{P_1}{1.275}\right) - \tan^{-1}\left(\frac{P_2}{1.275}\right) \right|$$

$$\Delta\theta = \left| \cos^{-1}\left(\frac{P_1 P_2}{|P_1||P_2|}\right) \right|$$

P_1 = vector from design eye point to fixation point 1

P_2 = vector from design eye point to fixation point 2

These equations assume that both the lateral and convergence movements can proceed concurrently with the total movement time being dependent on which movement takes the greater amount of time.

When one of the operator's hands is needed, HOS must decide which hand to use for the action. The logic that HOS uses is as follows:

(1) It will attempt to use the hand that is currently closer to the device, unless that hand is currently busy doing something else.

(2) If the preferred hand is busy, but will be free "soon," where "soon" is the amount of time that can be set by the analyst, then HOS will "wait" until the preferred hand is free and will then "move" the operator's hand to the device.

(3) If the preferred hand will not be free soon, then the operator's *other* hand is used -- assuming that it is free and can reach the device.

(4) If the operator's other hand is *not* free, but will be soon, HOS will again wait until that hand is free and then use it.

(5) If, however, the operator's other hand cannot reach the device, then a determination is made as to whether a *hand swap* should be initiated. In a hand swap, the less preferred hand takes over the function being performed by the preferred hand so that the operator can move the preferred hand to the device.

(6) If both hands are busy and won't be free for some time, or if a hand swap cannot be performed, HOS will decide that the instruction is unexecutable and will delay the execution of the procedure in which that statement is found until one of the operator's hands is free.

Similar logic pertains to the use of the operator's feet with the exception that "swaps" cannot take place.

The time required for a hand or foot movement is assumed to depend on both the magnitude and the precision of the movement. The equations that determine how long a hand movement will take are a combination of the results of experiments by Fitts (1954) and by Topmiller and Sharp (1965) are discussed in detail in Strieb, Glenn, and Wherry (1978). These data are shown in Figure 21 where they are compared with other hand movement studies. The same equations are also currently being used for foot movements, but with different basic parameter values.

Some key characteristics of the anatomy movement micro-model that should be noted here are:

(1) Movements, like the instructions that initiate them, are executed serially for each body part.

(2) Movements are ballistic -- once initiated they cannot be stopped nor can another action be initiated while the movement is taking place.

(3) Movement times are fully deterministic, based on where a body part is and where it is being moved to -- there is no variability.

(4) If a movement cannot be performed, an interrupt will be generated enabling the operator to select another procedure from the active procedure list for execution.

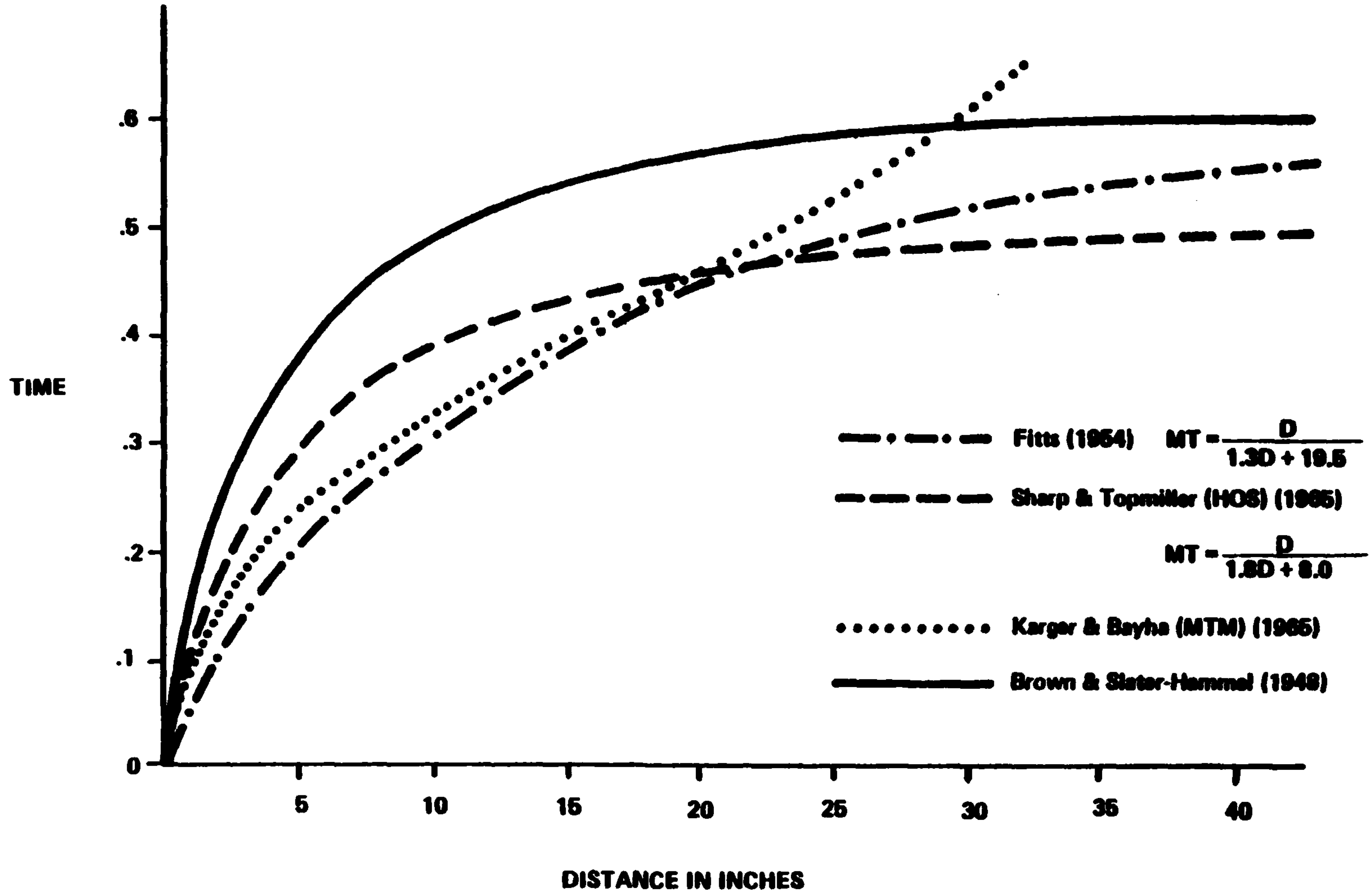


Figure 21. Hand Movement Times as a Function of Distance

3.6 PERFORMING A CONTROL MANIPULATION

Times associated with control manipulations are highly variable because of the diverse types of controls used in different operator stations. Consequently, HOS allows the user to describe the characteristics of a control which are used to determine a set of equations that describe the time associated with a control manipulation. In addition, there are a set of "packaged" calculations that compute control manipulation times for two basic control types -- discrete controls and continuous rotary knobs.

For discrete controls, the analyst is required to supply a time that represents the time required to move the control through a single setting. If a control manipulation results in a movement through several settings, the time assigned will be the time required for a single setting multiplied by the number of settings.

The formula for the manipulation time for a continuous rotary control was derived by fitting a quadratic to a table of data presented by Karger and Bayha (1966). The resultant formula is:

$$T = .0482 + .0050F + .0084\ FA$$

where F is the force in pounds required to turn the control and A is the angle through which the control is to be turned, in radians.

Unlike some of the other actions that we have discussed -- information absorption, recall, anatory movement, etc. -- once initiated, control manipulations can proceed in parallel with other actions. Thus the operator *can* be performing manipulations concurrently with both his right and left hands.

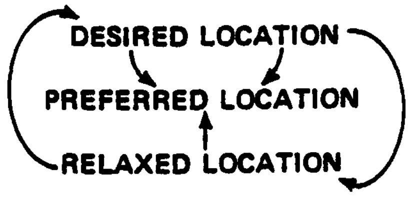
3.7 RELAXATION

The HOS relaxation micro-model interfaces with all the other action micro-models. Though fatigue itself is not currently modeled, HOS does exhibit one related characteristic that a real operator tends to exhibit -- when body parts are not busy doing anything else, the operator will move them to a comfortable, *relaxed location*. The analyst can override this default location by specifying a *grasp location* -- a location at which some action is expected. But after the operator has performed an action at the grasp location, the appropriate body part will automatically return to its relaxed location.

This logic is summarized in Figure 22. Any action establishes a location to which the operator must move in order to carry out the action. After the action has been carried out (and after a specified interval of time has elapsed), the body part will return to the grasp location (if one has been established) or to the relaxed location, if no other actions require that body part. *After an action occurs at the grasp location, that location is eliminated*, and body parts return to the relaxed location.

3.8 OPERATOR VARIABILITY

As described above, most of the equations that govern the operator micro-models in HOS are fully deterministic. This is consistent with two of the premises in the HOS model -- that the operator is a trained operator and that performance variations observed in experiments on individual operators are largely the result of situational differences, as opposed to differences in basic performance parameters. However, there are clearly differences in operator performance -- both between operators and for the same operator under different operational conditions. Some of the HOS operator parameters mentioned above enable the analyst to examine the effects of such differences -- the short-term memory cycle time, hab strength threshold, etc. In addition, by modifying the equations described above, one can readily describe an operator with a different performance profile.



BODY PARTS RETURN TO A RELAXED LOCATION WHEN NOT IN USE.

A "GRASP" LOCATION CAN BE ASSIGNED THAT TEMPORARILY OVER-RIDES THE RELAXED LOCATION

Figure 22. Relaxation Logic

Finally, there is a HOS construct that was a part of the original concept of HOS that was intended to model such performance differences under differing internal and external states. However, the *operator states* (o-states) concept has not yet been implemented because of the challenge that has so far confronted us in modeling average performance when no special stresses are influencing the operator.

4. SUMMARY

4.1 VALIDATION

Validation of any complex model (and particularly a Monte Carlo simulation model like HOS) is fraught with difficulties. One can argue that models can *never* be fully validated -- the best one can hope for is that in specific situations, given well-defined sets of inputs, the model can be shown to produce the outputs that match expectations, experience and available data. The problem is even more complex with a model like HOS because, unlike simulation systems that manipulate the user's model of a situation (i.e., the inputs) according to incontrovertible mathematical formulae, in HOS there is both the HOS model of the operator *and* the user's model of how the system functions and how the operator will utilize it. Both models must be valid for the results of any particular simulation to be valid. But since human behavior is so complex, one can never be sure that all possible circumstances have been fully described and all possible alternatives foreseen. It is therefore almost impossible to validate any specific model.

Notwithstanding these difficulties, efforts have been made to ensure both the validity of the HOS operator model and the reasonableness of the outputs obtained from specific user models. Tests of the validity of the HOS model have involved simulations of specific experiments drawn from the human factors and experimental psychological literature (Strieb, et.al., 1975, Glenn, Strieb and Wherry, 1977; Lane, Strieb and Wherry, 1977). User model validations have included simulations of specific Navy crewstations (Strieb, et.al., 1976; Strieb, et.al., 1976; Strieb, et.al., 1978; Strieb and Harris, 1978; Lane, Strieb, and Leyland, 1979; Lane, Leyland and Strieb, 1978). Both types of simulations have confirmed the general validity of HOS.

Although comparing model results with experimental data has generally been straightforward, validation of the model in complex military situations has been problematical because of the difficulties associated with attempting to capture all the potentially significant variables in the simulation. The converse of this problem is also true -- one can establish a scenario that can be run through HOS, but it is difficult (if not impossible) to set up real-world situations (e.g., at-sea exercises) that will conform to the hypothetical situations modeled in the simulations. Further confirmation of the HOS model is expected as the result of a series of HOS simulations coupled with laboratory experiments that are currently in the planning stages. These simulations will attempt to ensure the validity of the model (and will determine the values of certain input data quantities needed by the model) for a range of situations of varying complexity commonly experienced in Naval weapons systems. In addition, an effort is currently underway with NASA Langley that will test a HOS pilot model through its conformance with visual performance data collected by Spady and Kurbjun (1978).

4.2 ADVANTAGES OF THE HOS METHOD

Since HOS is basically an elaboration and formalization of task analytic techniques, it can be used for everything that task analysis can be used for. But HOS has several advantages over task analysis. First, it ensures a consistent level of description for all the tasks to be performed by the operator -- or at the least, it makes clear situations in which the task descriptions are not at the same level of description. Inconsistencies in the level of detail can be a significant problem with standard task analytic techniques because task analysis in general does not have a sufficiently well defined structure to ensure a consistent level of description. Second, unlike task analyses, HOS enables the dynamics of the situation to be simulated. Simulation permits the critical factors in system performance to be examined under controlled conditions. The analyst can completely control not only the rules under which the operator performs, but also the behavior of the external world. Furthermore, unlike experimental techniques, the results of HOS simulations are fully replicable so that the effects of modifications to the operator tactics or crewstation configurations can be thoroughly evaluated.

4.3 POTENTIAL USES OF THE HOS MODEL

Perhaps the best way to demonstrate the potential uses of HOS is to indicate some of the functions it can perform at various stages in the system design process.

In the *system definition* phase, HOS can be used to help define the functional requirements of the system. These include the identification of these functions to be assigned to man versus machine, as well as the functional requirements that the system itself must meet in order to satisfy its mission goals.

In the *system design* phase, HOS can serve both as the repository for data on the proposed system design specifications as well as a means of evaluating proposed designs.

In the *system development* phase, HOS can be used to develop standard procedures for the employment of the system and to evaluate proposed tactics for the utilization of system capabilities.

In the *system test* phase, HOS can be used to define the operator training requirements associated with the use of the system. It can also provide objective standards against which achieved operator performance can be measured.

Finally, in the *system evaluation* phase, HOS can help to provide insight into the types of decision and performance aids that would help to improve operator performance. It can be used to evaluate the performance improvements that would be obtained with such aids, thereby facilitating re-designs and re-definition of existing system and the design and definition of new systems.

HOS is not an easy tool to use. It demands a disciplined approach to the identification and characterization of system parameters. And it requires a heavy investment of time, money and analytical talents. But it is not just a tool for human factors engineers. Rather, it is a tool that can benefit a variety of users at various stages of system design. It provides a method for coordinating their efforts, yielding results that could not be obtained by any other means, and thereby justifying the effort required.

4.4. AN EXAMPLE OF THE USE OF HOS IN EVALUATING SYSTEM DESIGN

So far, HOS has been applied primarily to the evaluation of current systems. Some of these efforts have, however, indicated how valuable the timely application of the HOS model would be in evaluating proposed system design modifications. In particular, one recent effort (Strieb, et.al, 1978; Strieb and Harris, 1978; Lane, Leyland and Strieb, 1978; Lane, Strieb and Leyland, 1979) studied several alternative configurations of the non-acoustic sensor operator's station on the Navy's P-3C Anti-Submarine Warfare (ASW) aircraft. The simulations modeled the activities of the non-acoustic operator during a reconnaissance mission similar to those currently flown in the Mediterranean. In one simulation, the crewstation was configured as a "baseline" P-3C aircraft. In a second simulation, the operator was given an additional capability -- a manually controlled forward looking infrared (FLIR) system. In the third simulation, the operator was given an automated FLIR system. Comparisons of the simulation results indicated the extent to which the operator's performance on other tasks was degraded when the manual FLIR system was used. The simulations clearly showed that not only was the operator unable to maintain satisfactory performance on his other tasks, but also that the operator was unable to use the manual FLIR system to obtain the additional data for which the system had been intended. The automated FLIR rectified these problems.

The simulations demonstrated that, had HOS been used to evaluate each configuration before it was introduced to the fleet, an unacceptable configuration could have been avoided, thereby saving substantial sums of money. The conclusions reached from these simulations were confirmed by reports from the fleet.

4.5 CONCLUSIONS

HOS is a powerful technique for evaluating proposed designs for complex systems. The method has been shown to produce realistic assessments of operator taskload demands and overall system performance. The technique is one that should be used throughout the system design process to ensure the practicality of proposed designs in time-critical mission situations.

APPENDIX A

A BRIEF HISTORICAL PERSPECTIVE OF HOS

(1967-1978)

APPENDIX A
A BRIEF HISTORICAL PERSPECTIVE OF HOS
(1967-1978)

Robert J. Wherry, Jr., Ph.D

It is a truism that necessity is often the mother of invention -- and this is certainly true with regard to HOS. It was conceived out of feelings of frustration and disappointments with the impotency of human engineering technology of the mid-Sixties. The concept of a Human Operator Simulator (HOS) did not suddenly appear to me one day, but was, I believe, the inevitable outcome of consciously searching for a better approach to solving human engineering problems. The concept of modeling human behavior had attracted me for a number of years, however, prior to those months in 1967 when HOS was ultimately conceived. I am certain that the prior work with which I had been involved in the area of vigilance behavior, information processing under stressful conditions, and predictions of student pilot success or failure were instrumental in directing the ultimate conception of how humans processed information and did various tasks. Factor analytic studies I had done in Pensacola, Florida on a rather wide variety of pilot tasks had left on me an indelible appreciation (or belief, at any rate) that, perhaps, only a few independent factors really accounted for goodness of performance in what, at first, had appeared to be very diverse tasks. Finally, the experience which I had gained since 1959 in programming computers for complex applications in aviation psychology, medicine, and biophysics had made a believer of me with regard to the potential power of computer simulation for solving all sorts of problems.

Thus, HOS not only developed from a specific need, but it also grew out of what I consider to be an unusual and fortuitous series of experiences to which I had been exposed. To better appreciate the specific purposes for which HOS was initially conceived and developed, I must take you back to late 1966 when I was transferred from Pensacola, Florida to the Naval Missile Center (NMC) at Point Mugu, California to head up the human engineering branch. Our mission there was to accomplish the tests and evaluations of new Naval airborne weapons systems.

To perform a test and evaluation one must, of course, first decide what one desires to test. It became obvious that two different approaches were possible. The first I shall refer to as "comparison with specs and standards" and the second I shall call "performance evaluation." The first approach dealt with testing whether various aircraft displays, controls, labels, panels, etc., conformed to Human Engineering guides, standards and specifications. It may be recalled that MIL STD 1472 and MIL SPEC 46855 were first issued in 1966. Because of this we had in our possession, at that time, the latest documents containing data on what the Navy (and the other services as well) deemed to be "acceptable" HE design standards. On the other hand, because of the newness of those documents, no system arriving for test and evaluation at NMC for several years thereafter would have required a contractor to meet those standards and specifications. Thus, those documents did offer a standard of comparison by which at least some aspects of the crewstations could be evaluated even though it might be difficult or impossible to force an air frame contractor comply with those standards. A second drawback in using MIL STD 1472 was that no guidance on the impact on operator or system performance was provided in cases where various aspects of crewstation design failed to meet the new standards. I found that it was virtually impossible to get the Navy interested in correcting any single deficiency, because no single deficiency was ever so bad as to be able to say that it alone made the aircraft either unsafe or that it alone would be the cause of unsuccessful or aborted mission performance. It was obvious to our human engineering team that *the cumulative effect of a series of minor deficiencies could and would have a major impact on system safety and mission success.* To be able to convince others of this point of view, however, would require a fairly detailed model of the impact of various display and control features on human information absorption, processing, and transmission in a task sequencing framework to illustrate such cumulative effects! Unfortunately, such models were not available at that time.

The second approach to the test and evaluation of the crewstation dealt with attempting to determine (regardless of conformance or non-conformance to various MIL STDs) if the operators were able to adequately perform the various functions which had been allocated to them. In attempting to determine precisely what was expected of a given operator, we had occasion to examine a wide variety of task analyses and timelines which had been prepared by a variety of different contractors. Without exception, these rather costly items, when they had, in fact, been prepared, were extremely disappointing in terms of adequately expressing what was actually expected of a given operator. All too often task analysis blocks had been prepared at a very macro level (e.g., "Pilot acquires and locks on target") and times assigned to such activities were, obviously, merely "educated guesses." It was my personal experience that, at least by the time a weapon system was delivered to NMC, no task analysis or timeline indicated that the operator would be too busy to perform all the functions he had been assigned. The task analyses which we reviewed in those days also failed to give the reader a good appreciation of the often necessary simultaneity of various different task demands facing a particular operator during crucial segments of a mission. It became obvious that a more stringent set of rules were needed in guiding whoever prepared task analyses so that (a) an appropriate level of detail would be included, and (b) a given statement made by a task analyst could be interpreted without ambiguity as to what the operator's responsibilities were. (From this concept, the Human Operator Procedures (HOPROC) language ultimately arose.) Further, it was felt that a successful accomplishment of any task analysis really involved two distinct efforts, the first of which was expressing what was expected of the operator (in terms of what actions he must take) and the second was (given the displays, controls, and layout of the crewstation) to determine if the operator could, in fact, accomplish all those assigned tasks within the requisite time. This implied that the tasks themselves ought to be able to be described independently of the particular crewstation layout, *and, if a sophisticated human performance model were available,* then the impact of different crewstation designs could be reliably and objectively evaluated without relying on "educated guesses" by contractor personnel who had a personal interest in making their own aircraft appear to be good in the Navy's eyes.

In a very real sense, the first concept of HOS was never intended to simulate all types of human performance, but it did set out to quantify performance times of various types of anatomy movement (head and eyes, hands and arms, feet, etc.) and the effect of various features of displays and controls (e.g., size, contrast, shape, etc.). Actually, my own feeling by late 1967, was that we were a long way from being able to predict the times various mediated mental processes might take, but that at least those observable events, such as anatomy movements, absorption of information from displays, and manipulation of controls, should be able to be accurately predicted. In this respect, I was especially encouraged by the work of Topmiller and Sharp (1965) which had indicated that arm-hand reach time was very predictable. Also several informal studies (which, I deeply regret, have never been published) on eye movement and fixation times and on numeral and dial reading times which were conducted by Alvah Bittner and myself at Pt. Mugu that greatly supported the concept that any task could be broken down into sequences of various *micro* processes and the sum of the micro process times would, in fact, yield the total task times. Many people rejected such a hypothesis and predicted that there would be tremendous interactions among many if not all of the micro processes which would make the analytical "additive" approach I was advocating doomed to failure. Such discussions and arguments, I might add, were very philosophical, since neither I nor my opponents had sufficient data to support our contentions in those days.

I suppose I stuck with the belief that each micro process was independent primarily because, if it turned out not to be true, there would be little hope for a "scientific" approach to human engineering in the, then, foreseeable future.

In addition to the above-mentioned reasons for the development of a HOS, there was yet another reason. In those days, we were conducting some open-loop simulations of various missile and missile launch systems. In one conducted by Chuck Hutchins, it was discovered that operators in the laboratory simulation were getting very good scores on locking onto and launching a simulated missile at a simulated target. It was also discovered that the operators were waiting until minimum range to launch their simulated weapons. The simulated targets were capable of maneuvering, but the maneuvers were "canned" and had nothing to do with the maneuvering our pilots were doing. Further, the simulated targets never fired back, which might well account for the willingness of our pilots to wait until minimum range to release their missiles. Thus, the concept of a simulated human operator to be used as an intelligent adversary was also one of the original planned uses of a HOS (although, to date, HOS has never been used for this purpose).

In formulating the philosophy of how one could simulate a human operator's behavior, one major concern I had in 1967 was whether a human being could be considered to be a discrete or continous information processor. In those days, many people held to the concept that man was indeed a continuous processor. If this were true, it might be more appropriate to use an analog rather than a digital computer. However, by reanalyzing some data collected much earlier by John Senders, I came to the conclusion that even in a continuous tracking task, the human appeared to be sampling the available displayed information only about 13 times per second. Thus, man appeared, at least to me, not to be a continuous sampler, but a discrete one who could relatively easily be simulated with a digital computer.

Another major philosophical point was whether man should be considered to be a single- or multi-channel processor. This is more than a question of whether an operator can be responsible for carrying out more than one task at a time, for this he might appear to do even if he were a single-channel operator capable of very rapid interlacing among more than one task. The single-channel vs. multi-channel question really involves on the issue of whether the operator can *simultaneously* be thinking about two different things. After much introspection (as well as considering the writings of various experts on this question), I chose essentially to conceive of man as being a single-channel processor who is capable of rapidly multiplexing among several tasks.

This, in turn, led to the concept in HOS of permitting the simulated operator to have many different procedures going on at the same time. In HOS, we call these the *active* procedures, while those which are not currently of concern to the operator are known as *inactive* ones. However, while many tasks may be active, HOS only works on one at a time.

One of the earliest studies I did (long before HOS or the HOPROC language existed) was a relatively simple computer simulation to determine what would happen under various strategy algorithms for deciding which displays to pay attention to when the simulated operator was responsible for monitoring several different ones at the same time. These early studies led to the concepts of a *monitoring procedure* for a display as well as the concepts of a procedure's *criticality* and the idea that criticality could dynamically change as a function of the disparity between a display's *desired position*, its *allowable limits*, and its *estimated position*. These concepts have been retained in HOS since its beginning stages back in early 1968. Such algorithms provide the basis for the adaptiveness of the behavior exhibited by HOS.

Another very early consideration (which has changed very little over the years) was how to handle "short-term" memory of the simulated operator. The concept of *hab* strength (which is discussed elsewhere) and the probability of successful recall of an item of information which had been recently absorbed was a concept which I adapted from Hull's and Thorndike's theories of learning. The concept of modeling short-term memory was felt to be necessary to determine how often the human would feel a necessity to update his current information about some parameter by actually looking at a display.

By 1968, the basic concepts of HOS which included micro-process handlers, adaptiveness algorithms, short-term memory, with the operator as a single-channel processor capable of rapid multiplexing among the "active" procedures had been formulated in detail as well as the earliest version of the HOPROC language by which the user would specify what it was that the simulated operator was expected to do. These concepts were reported in the proceedings of a two-day meeting jointly hosted by the Office of Naval Research and North American Aviation in Columbus, Ohio in November, 1968. It is surprising and somewhat rewarding to see how little the basic concepts formulated 10 years ago have changed during its development. It is also interesting to note that I and other participants at that meeting estimated that it might take 10 years to develop HOS.

By 1969, I was able to get some Independent Research (6.1) funds to pay for a programmer (Mr. Don Kennerly -- then a member of our HE branch) to start programming both the earliest versions of HOS and HAL (the HOPROC Assembler/Loader program which was to decipher the HOPROC statements for input to the HOS program). These earliest programs did not include all the specifications of HOS mentioned above and HAL was written in COBOL. More than anything else, they proved, at least to my own satisfaction, that it would be feasible to write a digital computer program for a full-blown HOS. It was also in 1969 that Bittner and I did the experiments mentioned above which also were very encouraging regarding the concept of the additivity of micro-process times.

In August of 1970, I was transferred to the Naval Air Development Center in Warminster, Pennsylvania and it was immediately obvious that a HOS would be even more valuable during early system development than during later test and evaluation phases of system design.

Paul Chatelier, who was at that time stationed at NADC, was in the throes of formulating CAFES which also dealt with computerized approaches to improving human factors engineering technology. (Later funding for HOS was formally included in the CAFES program element number, but for two years they stayed as separate development efforts.)

By December 1970, Analytics became interested in the HOS comcept and submitted a proposal to work on its further development. Prior to that, I had discovered that although I had brought the HOS and HAL programs (written by Kennerly) with me, NADC did not have a version of COBOL which could compile the HAL program as it then existed. It was decided that it would be better to have all the future programs written in FORTRAN for subsequent ease in transferring them about the country. The first contract to Analytics for work on HOS was let in 1971 and out of that effort what I might call HOS II and HAL II were developed.

As more work was accomplished on HOS, it became obvious that various additional statements in the HOPROC language would be desirable as well as a greater flexibility in how one could express various statements. For a while, these additions were added as patches to the program until it became obvious that it was time to go back and incorporate all these changes as well as some additional new concepts into the HAL and HOS programs. Thus, what had been available since late 1975 actually is what we might call HOS III and HAL III. Since that time, we have almost exclusively been involved in validity testing of HOS III and little or no additional development has taken place.

This does not mean to imply that HOS is considered to be in its final or ultimate stage of development, for there are many additional features which should and could be added to HOS. However, HOS III does represent what I consider to be a highly useful technique for the initial assessment of how well a trained operator will be able to perform his tasks in a specified crewstation under varying situational demands.

One concept which was definitely added to HOS in 1974 which was a rather marked departure from original plans for HOS, was the concept of simulating the system hardware and software as well as targets using the HOPROC language and the HAL and HOS programs. Originally, HOS was only to be the Human Operator Simulator and it was anticipated that it would be interfaced to hardware simulators in some fashion. It was found, however, that it would be extremely difficult to modify hardware simulators written by others so that HOS could easily interface with them. After much soul searching, it was decided to expand the HOPROC language, HAL and HOS, to include the ability to simulate hardware as well as the human components. These changes were also incorporated and indeed necessitated the rewriting for HOS III and HAL III.

The concept of a HODAC (Human Operator Data Analyzer/Collator) program to analyze the human operator data emminating from a HOS run was included in the very early stages of HOS planning. The first HODAC, however, was not available until 1974. It has proven to be less useful than I originally thought it might, but this may be due, in part, to the fact that we have to date been most interested in seeing if HOS behaves like real operators in systems which have already actually been built (i.e., our validating studies) rather than in systems which are actually under development. It may be that many of the routines available in HODAC will turn out to be very useful in deciding potential changes to procedures and crewstation design when we try HOS on a developing system and we determine that unless something is changed, it will be impossible for the operator to successfully do all his allocated functions.

While HAL III and HOS III both now contain the additional capability for simulating hardware and target systems, there is no automatic logging of their behavior as there is with the simulated human behavior. In part, this is due to the fact that HOS does not contain a general purpose hardware system model which is made system specific by the hardware procedures and hardware functions. Lacking an overall scheme for a general hardware system means that hardware systems are not automatically reducible to a specified number of micro-processors which can then be automatically logged out whenever they are used. This necessitates some amount of cleverness on the part of the HOS user to either log out and/or accumulate data of interest ot overall system performance.

Earlier, I mentioned that HOS should not be considered to be fully developed. Areas where HOS might be expanded to include the addition of a fatigue modes, the capability to pick up and move objects from one place to another, the capability to walk (or run) from one place to another, the ability to talk to another operator, the capability to perform visual target recognition in a complex visual field, etc. I am convinced that each of the above concepts can be added to HOS and I have, at least, rudimentary models or schemes for handling all of the above concepts as well as several others. With the rather successful validation studies which have been conducted on HOS III, it is probably now time to start the development of HOS IV and HOS V which would be versions to include one or more of the above concepts.

Finally, I would be remiss if I did not discuss the concept of operator error as it is treated in HOS. More than any other single item, the way operator error is treated in HOS has been criticized. With a few exceptions (which are discussed elsewhere), HOS does not make errors, Some people claim that this is unrealistic, but I maintain that as long as the human operator is given the requisite amount of time to do a task, then he does not make errors. He may not finish all the tasks we would like him to do, and he may not do them as well as we would like him to do them, but as long as he works at a reasonable pace, he will not make an error. The fact that he doesn't get all his tasks accomplished when he works at a reasonable pace merely indicates that we allocated too many tasks to him. Thus, if HOS indicates the simulated operator spends too little time on a given task, we may either assign a higher criticality to that task and rerun the simulation to see if this alleviates the problem, or we may reduce the number of tasks which were originally allocated to the operator to see if this solves the problem.

I am certain that real systems do exist in which operators make mistakes. On the other hand, this is a clear indication that we have, in those systems, asked the operator to do too many and/or too complicated tasks and therefore those systems are not properly human engineered by definition, since successful performance of the tasks are not within the capabilities of the operator. What HOS does is essentially to "instruct" the operator not to attempt to work at a pace at which errors and mistakes will occur because we hold to the concept that *errors are the result of being under time stress* and that error-free performance can be maintained provided the operator does not attempt to do too many things in too short of a time period. The human errors that are observed in existing systems are the result of real operators attempting, unsuccessfully, to perform at a higher level than their capabilities permit.

In closing this brief historical perspective, I should also mention that working on the development of HOS has been both fun and exciting. The associations I have formed with the many people who have participated in this development program has been very rewarding professionally over the years. Finally, working on HOS has also often been a humbling proposition as we have discovered how little we actually know about how humans behave.

REFERENCES

1. Baker, C.C., Johnson, J.H., Malone, M.T., and Malone, T.B. 1979. Human Factors Engineering for Navy Weapons System Acquisition. Essex Corporation, Alexandria, Va.

2. Dodge, R. and Cline, T.S. 1901. The Angle Velocity of Eye Movements. *Psychology Review*, 8:145-157.

3. Fitts, P.M. 1954. The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement. *Journal of Experimental Psychology*, 67:103-112.

4. Glenn, F.A., Strieb, M.I., and Wherry, R.J. 1977. *Applications to Assessment of Operator Loading*, The Human Operator Simulator, Vol. 8, Technical Report 1233-A, Analytics, Willow Grove, Pa.

5. Karger, D.W., and Bayha, F.H. 1966. *Engineered Work Management*. Industrial Press, Inc., New York.

6. Lane, N.E., Leyland, W., and Strieb, M.I. 1978. *Prediction of System Performance and Cost Effectiveness Using Human Operator Modeling*. Proceedings of the 1978 Science and Engineering Symposium. Naval Amphibious Base, Coronado, Ca.

7. Lane, N.E., Strieb, M.I., and Leyland, W. 1979. *Modeling the Human Operator: Applications to System Cost Effectiveness*. NATO/AGARD Conference Proceedings No. 268, Avionics Panel, Modeling and Simulation of Avionics Systems and Command, Control and Communications Systems, Paris, France.

8. Lane, N.E., Strieb, M.I., and Wherry, R.J. 1977. *The Human Operator Simulator: Workload Estimation Using a Simulated Secondary Task*. NATO/AGARD Conference Proceedings No. 216, Aerospace Medical Specialists Panel, Methods to Assess Workload, Cologne, Germany.

9. Melton, A.W., Crowder, R.G., and Wulff, D. Short-Term Memory for Individual Items with Varying Numbers of Elements. Cited in Melton, A.W. 1963. Implications of Short-Term Memory for a General Theory of Memory. *Journal of Verbal Learning and Verbal Behavior*, 2:1-21.

10. Murdock, B.B. 1961. The Retention of Individual Items. *Journal of Experimental Psychology*, 62:618-625.

11. Peterson, L.R. and Peterson, M.J. 1959. Short-Term Memory of Individual Verbal Items. *Journal of Experimental Psychology*, 58:193-198.

12. Spady, A.A. and Kurbjun, M.C. 1978. Flight Management Research Using an Oculometer, SAE International Air Transportation Meeting, Boston, Ma.

13. Strieb, M.I., Glenn, F.A., and Wherry, R.J., 1978. *HOS Study Guide*. The Human Operator Simulator, Vol. IX. Technical Report 1320. Analytics, Willow Grove, Pa.

14. Strieb, M.I. and Harris, R. 1978. *Sensitivity Analyses for Cost Effectiveness of P-3C SS-3 Operator Station*. The Human Operator Simulaton, Vol. 10 Part 2. Technical Report 1330, Analytics, Willow Grove, Pa.

15. Strieb, et.al. 1975. *HOS Simulation Descriptions*. The Human Operator Simulator, Vol. 6. Technical Report 1181-B. Analytics, Willow Grove, Pa.

16. --- 1976. *LAMPS Air Tactical Officer Simulation*. The Human Operator Simulator, Vol. 7. Technical Report 1200. Analytics, Willow Grove, Pa.

17. --- 1978. *P-3C SS-3 Operator Station Cost Effectiveness Evaluation*. The Human Operator Simulator, Vol. 10. Technical Report 1289. Analytics, Willow Grove, Pa.

18. Topmiller, D.A. and Sharp, E.D. 1965. *Effects of Visual Fixation and Uncertainty on Control Panel Layout*. Technical Report AMRL-TR-65-149. Aerospace Medical Research Laboratory, Wright-Patterson Air Force Base, Ohio.

19. Wortman, D.B., Sigal, C.E., Pritsker, A.A.B., and Seifert, D.J. 1975. *New SAINT Concepts and the SAINT II Simulation Program*. Technical Report AMRL-TR-74-119. Aerospace Medical Research Laboratory, Wright-Patterson Air Force Base, Ohio.

DATE
ILMED
-8