1

AD ____________

DEVELOPMENT OF MILITARY PERFORMANCE MODELS FOR THE ASSESSMENT OF PSYCHOPHARMACOLOGICAL AGENT IMPACT

ANNUAL AND FINAL REPORT

K. Ronald Laughery
Carl Drews

November 1987

Supported by

U.S. ARMY RESEARCH AND DEVELOPMENT COMMAND
Fort Detrick, Frederick, Maryland 21701-5012

Contract No. DAMD17-84-C-4234

Micro Analysis and Design
9132 Thunderhead Drive
Boulder, CO 80302

Approved for public release;
distribution unlimited

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION
UNCLASSIFIED

1b. RESTRICTIVE MARKINGS

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION / DOWNGRADING SCHEDULE

3. DISTRIBUTION / AVAILABILITY OF REPORT
Approved for public release; distribution unlimited

4. PERFORMING ORGANIZATION REPORT NUMBER(S)

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION
Micro Analysis and Design

6b. OFFICE SYMBOL *(If applicable)*

7a. NAME OF MONITORING ORGANIZATION

6c. ADDRESS *(City, State, and ZIP Code)*
9132 Thunderhead Drive
Boulder, Colorado 80302

7b. ADDRESS *(City, State, and ZIP Code)*

8a. NAME OF FUNDING / SPONSORING ORGANIZATION U.S. Army Medical Research & Development Command

8b. OFFICE SYMBOL *(If applicable)*

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
DAMD17-84-C-4234

8c. ADDRESS *(City, State, and ZIP Code)*
Fort Detrick
Frederick, Maryland 21702-5012

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.
63764A	3M6-3764D995	AB	163

11. TITLE *(Include Security Classification)*
Development of Military Performance Models for the Assessment of Psychopharmacological Agent Impact

12. PERSONAL AUTHOR(S)
K. Ronald Laughery, Carl Drews

13a. TYPE OF REPORT
Annual and Final*

13b. TIME COVERED
FROM 9/1/84 TO 9/30/87

14. DATE OF REPORT *(Year, Month, Day)*
November 1987

15. PAGE COUNT
30

16. SUPPLEMENTARY NOTATION
* Annual covers period September1, 1985 - September 30, 1987
Final covers period September 1, 1984 - September 30, 1987

17. COSATI CODES

FIELD	GROUP	SUB-GROUP
09	02	
05	10	

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
simulation, modeling, SAINT, drug effects, human operators, task network modeling, microcomputer
RA V. ([illegible]

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

In time of war, the human operators of military systems may be exposed to harmful psychopharmacological agents. Certain pretreatment drugs are known to ward off the harmful effects of chemical agents, but these drugs have adverse side effects that may degrade a soldier's ability to perform an operation. This report describes the development of a task network modeling tool which is used to simulate the effect of drugs on human performance.

This tool is a software package known as Micro SAINT. It runs on an IBM PC or compatible microcomputer. The construction of models is entirely menu-driven, and does not require knowledge of computer programming. This report contains recommended procedures for conducting simulation analysis. Micro SAINT is presently being used in over 55 government installations.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL
Virginia M. Miller

22b. TELEPHONE *(Include Area Code)*
(301) 663-7325

22c. OFFICE SYMBOL
SGRD-RMI-S

SUMMARY

This report describes the development and use of a microcomputer-based simulation tool known as Micro SAINT. The need for this tool was foreseen by the Joint Working Group on Drug-Dependent Degradation in Military Performance (JWGD3 MILPERF). The JWGD3 is studying the effects of pretreatment drugs on soldier performance, and needed an easy-to-use tool for conducting simulation experiments.

We have developed and delivered Micro SAINT, a software package which runs on an IBM PC or compatible microcomputer. Micro SAINT is not a computer model in itself. Instead, it is a menu-driven modeling system that is used to build models of human operators. It requires no prior knowledge of computer programming. This report describes a number of real applications in which Micro SAINT was used during the course of the project, including models for an M60 tank, an LHX helicopter, and an aircraft maintenance operation. The final section gives procedures for studying drug effects, workload, and manpower requirements.

FOREWORD

Citations of commercial organizations and trade names in this report do not constitute an official Department of the Army endorsement or approval of the products or services of these organizations.

TABLE OF CONTENTS

LIST OF FIGURES

1. Introduction

This report describes work performed for the Joint Working Group on Drug-Dependent Degradation of Military Performance (JWGD3 MILPERF). The Joint Working Group, a section of the U.S. Army Medical Research and Development Command, is studying the effects of pretreatment drugs on soldier performance. A simulation tool is needed to conduct experiments involving psychopharmacological agents.

Historically, simulation has been difficult foı researchers. Many of the classical simulation languages are based on a programming language such as FORTRAN, and they require some experience in coding and debugging. We have solved this problem by creating Micro SAINT, an easy-to-use simulation tool that is different from classical languages. Micro SAINT allows researchers to build their own models of human performance, to collect data, and to analyze the results without any prior experience wiıh computers.

Section 2 of this report is a short history of our work on the contract. It gives a brief description of each projec' in which Micro SAINT has been used. Sections 3 and 4 describe the software and documentation. Section 5 suggests potential applications for Micro SAINT, and includes recommended procedures for collecting data, building models, and running simulation experiments. Section 6 concludes this report.

2. Background

We made continual efforts to put our software into the hands of users while working on the development of Micro SAINT. The comments we received from these installations were often incorporated as features in the final product. We are confident that this feedback will ensure the acceptance of Micro SAINT throughout the Joint Working Group. This section of the report gives a chronology of these applications, and also describes the technical milestones that resulted from them.

The first application of Micro SAINT in a real situation occurred in February 1985 at Fort Knox, Kentucky. With Dr. Fred Hegge (Director, JWGD3) we showed the prototypal software to a group of subject matter experts (SMEs) who were non-commissioned officers from an Armored division. The M60 tank has a four-man crew, and the officers were able to describe to us the sequence of voice commands, responses, and actions that take place between the times a target is spotted and a shot is actually fired. We built a model of the firing sequence and returned to Fort Knox in April to check its accuracy. In a follow-up session with the tankers, we made several revisions and included some special cases (Reference 1). The Micro SAINT software and tank model were delivered to the Advanced Technology Research Division at Fort Knox, to be used in further drug studies by the staff there.

Out of this first application came two important results. First, we were greatly encouraged by how well the SMEs were able to understand the methodology of task network modeling. It is similar to a flow chart, and the tank commanders were able to correct our diagrams using the symbols we had chosen.

The second result was a needed feature: Our system had to be able to model multiple operators. Communication between crew members is a critical part of many military systems, and is subject to delays when spoken words become unintelligible due to harmful chemical agents. Most of the military systems we have looked at since then have involved more than one person, and the interaction between them has been especially important. We enhanced the execution module of the software so that it could handle multiple entities within a network.

The M60 tank model turned out to be the archetype for the entire project, in that it embodies the technique for studying the affects of psychopharmacological agents on military performance. The original model is constructed from the SMEs' knowledge of normal operating conditions. Each task in the model has certain performance parameters, such as the mean time to perform that task. These task parameters are modified to reflect estimates of drug-induced performance changes. By running the modified model, researchers are able to determine how the task-by-task changes influence the operation of the whole military system. This process is explained more fully in Section 5.

In April 1985 we began a 3-month simulation project under a contract with Texas Instruments in Dallas, Texas. The ARTI (Advanced Rotorcraft Technology Integration) program was addressed to the cockpit design for the Army's LHX helicopter. This is to be a one-man helicopter, and Texas Instruments was determining whether a single pilot could handle the same workload that was handled by a pilot and co-pilot. We simulated a combat mission that involved entrance into the enemy zone, several combat engagements, egress from the zone, and battle damage assessment. The models we built for this project were much larger than the M60 tank model mentioned earlier; the largest had 250 tasks (Reference 2).

The important result that came out of the LHX effort was our approach to operator workload. We divided the pilot's attention into four channels: visual, auditory, cognitive, and psychomotor. The attention value for a channel was stored in a variable (named v, a, c, or p). Each task increased the values of the four attention variables when it started, and decreased them again at its completion. After running the simulation, we were able to plot these variables over time to identify overload points in the mission. We hypothesize that the pilot's overload thresholds would be lower in the presence of chemical stressors, and the Analyze application allowed us to look for overload points under different threshold values. Our results showed that Micro SAINT is a viable tool for studying operator workload.

In September 1985 we began work on a contract with Universal Energy Systems at Fort Rucker, Alabama, where U.S. Army helicopter pilots are trained. The trainees go through a 6-week program that involves both classroom instruction and in-flight practice. Our task was to determine the amount of resources required throughout the course schedule, as well as to model the skill acquisition of a student pilot.

The final resource model, which was the largest model we had built up to that time (Reference 3), had about 400 tasks. The biggest influence of Fort Rucker on the evolution of Micro SAINT was in the size and complexity of our models. Increasing the maximum number of tasks to 400 took up most of the memory in an IBM PC (IBM Corporation, Boca Raton, FL) equipped with 640K of random access memory (RAM). Another new feature was the addition of variable arrays, which greatly increased the power of Micro SAINT by enabling it to deal with groups of entities.

In January 1986 we constructed a demonstration model of a combat turn process at an Air Force base. The term "combat turn" refers to the sequence of activities that a fighter plane undergoes in the course of a single day. The planes go through a pre-flight inspection and are sent off on a mission in groups. When the planes return, they must undergo a series of maintenance tasks and inspections before they are ready to fly again. These tasks are performed by maintenance personnel, each of whom is currently trained to handle a limited set of maintenance functions. Our combat turn model examined the manpower requirements of the facility, and showed that airplanes would build up at the maintenance stations at certain times during the day. The presence of a backlog indicates that aircraft time is being wasted, and suggests that the maintenance process should be speeded up. The solution being considered was to cross-train the maintenance personnel so that they are qualified to perform more functions. With our model we were able to perform experiments based on hypothetical situations and show a dramatic improvement in the aircraft downtime and in the number of missions flown.

The combat turn model uncovered a glaring deficiency in the capabilities of Micro SAINT. We found it awkward to model a number of entities waiting to be processed at a station. While our previous problems had involved a small number of human operators performing procedural tasks, this application involved a number of planes being processed in a sequence similar to an assembly line. The planes could proceed if resources were available, but otherwise they had to wait. To handle this limitation, we decided to add a new feature to the methodology. In May 1986 we completed work on Micro SAINT "job queues," which allow entities to wait in a queue before being processed. Job queues allow the model builder to handle a large number of entities flowing through a system, whether those entities are parts, soldiers, or jobs to be performed. An important lesson we learned from this experience was that listening to our users would provide the best direction for future development efforts. With job queues, we were able to decrease the number of tasks in the combat turn model from 80 to 20.

Until this stage of development we had not used the graphics capabilities of the IBM PC. We had been using Symphony (from Lotus Development Corporation in Cambridge, Massachusetts) to plot the data collected during the execution of a Micro SAINT model. Although this was an adequate technical solution, it was also clear that the need for external tools would limit the acceptance of Micro SAINT within the military community. Interested researchers would have to procure Symphony in order to use Micro SAINT effectively. So we resolved to add graphics capabilities to the software.

In October 1986 we completed work on a special layout and routing algorithm that creates task network diagrams. The tasks in a network are displayed as ovals, and arrows between them show the sequence of execution. A sample diagram is shown in Figure 1. The diagrams can currently be printed out on a LaserJet printer (Hewlett-Packard, Palo Alto, CA) or an Epson-compatible dot matrix printer (Epson America, Torrance, CA). While developing a model, the user can view the diagram by selecting the "view" command. We have found that the diagram feature is an excellent way to check the accuracy of a model.

During this time we also worked on graphical output of Micro SAINT results, and in February 1987 we delivered Release 3.0 of the software. This release provides statistical functions for examining the data collected from a model. Researchers can create bar charts, line graphs, scatter plots, step charts, and time lines. A sample graph is shown in Figure 2. This visual presentation of data makes it much easier for researchers to communicate their conclusions to other people.

The new graphics capabilities were enthusiastically received by users, some of whom requested a further enhancement to the network diagrams. Our sponsor asked us to add a "mouse" (a hand-held pointing device) to the model development portion of Micro SAINT, so that model builders could create their diagrams by pointing and clicking if they chose to. We implemented this new interface by displaying a toolbox in the upper right-hand corner of the diagram. By clicking in the toolbox, model builders can select tools to add, modify, and move tasks or to draw connecting arrows between them. The addition of the mouse interface made the network diagrams more interactive. The mouse software was completed in April 1987.

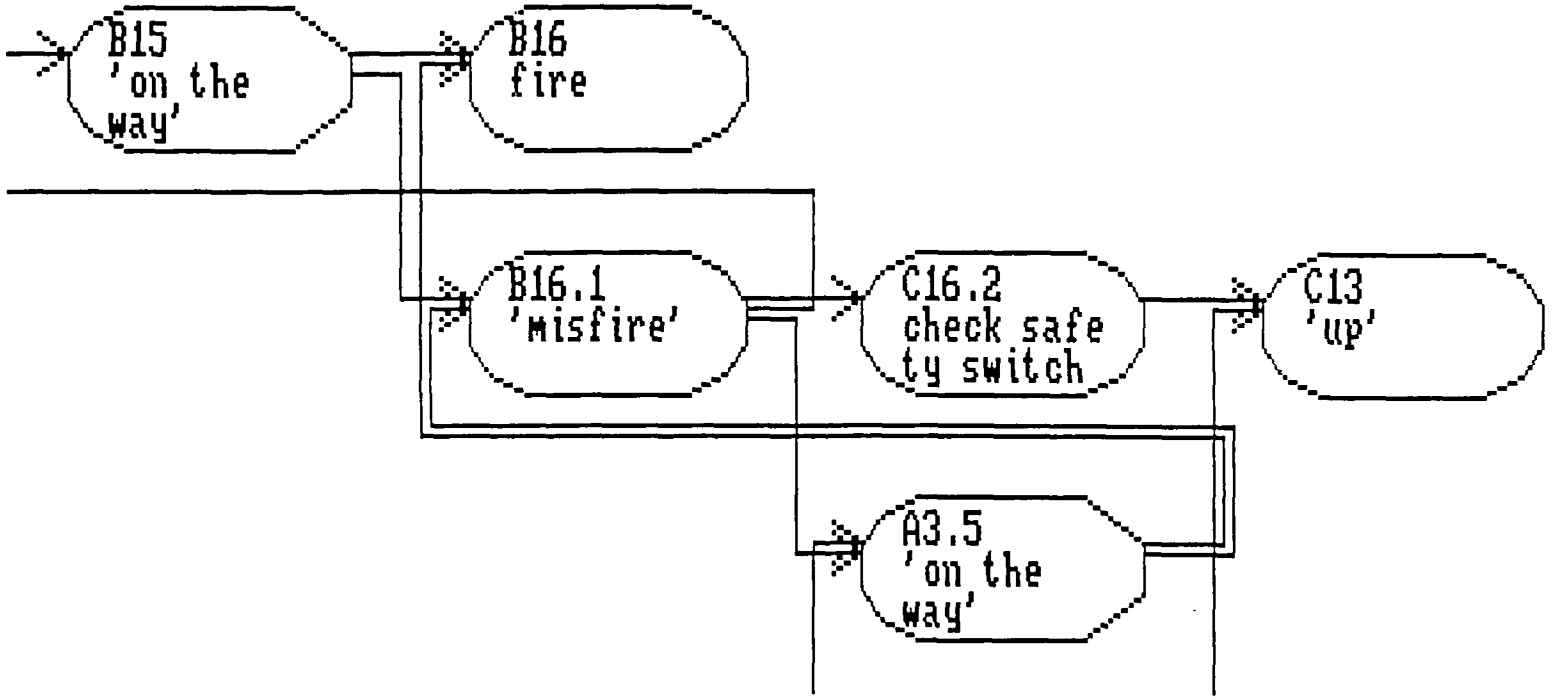


Figure 1. Task Network Diagram

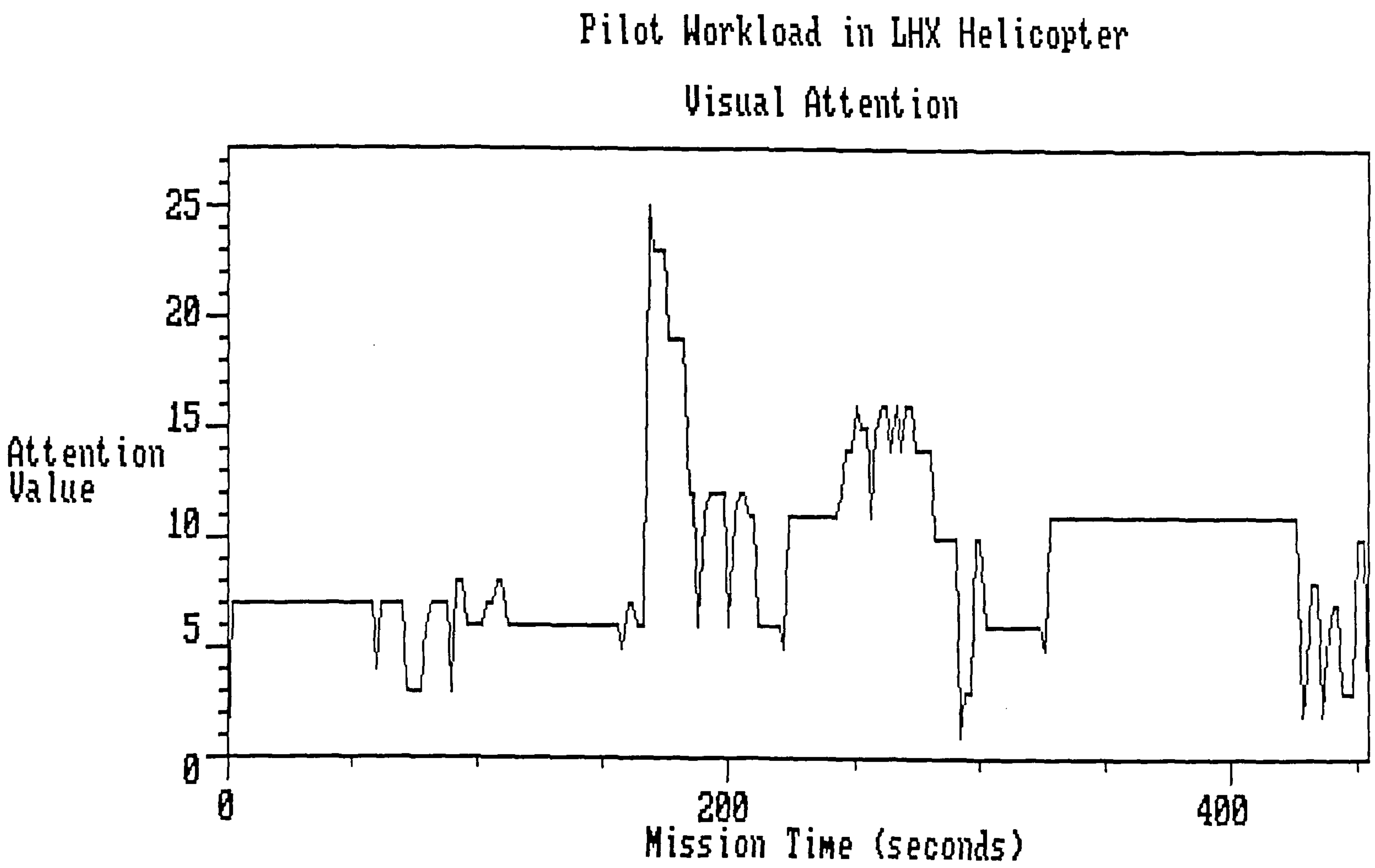


Figure 2. Sample Graph

3. The Software

The Micro SAINT software consists of 32,880 lines of source code. It is written in the programming language C. We did not refer to the original mainframe SAINT code, and did not use any of it in developing Micro SAINT. The system runs on an IBM PC microcomputer equipped with a minimum of 512K of RAM and two floppy disk drives. The recommended configuration is 640K of RAM and a hard disk drive. Micro SAINT has four main application programs: Develop, Execute, Analyze, and Utilities. Each program handles one phase of the modeling process. Accompanying the programs are 150 help screens and some sample models. The entire package resides on five floppy diskettes and occupies 1 Megabyte of storage on a hard disk drive.

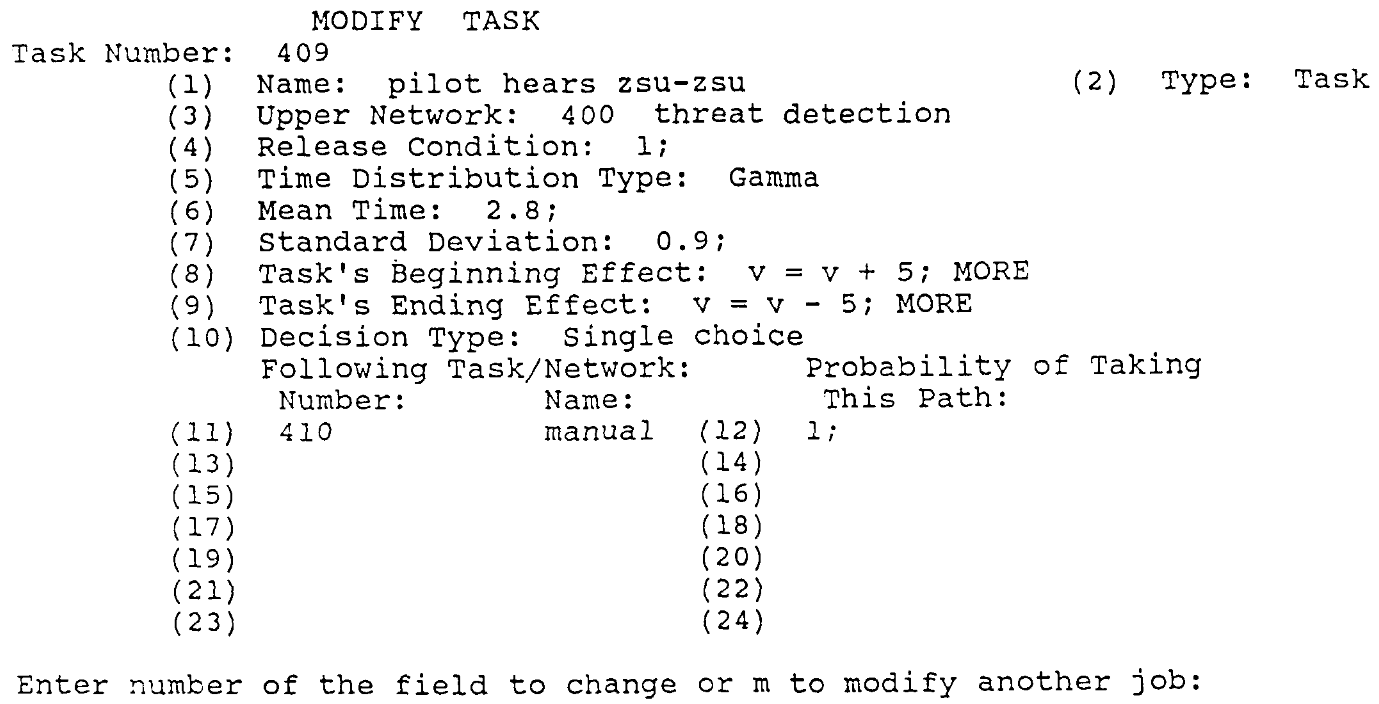
Micro SAINT embodies a new direction in simulation technology. The system differs from traditional simulation languages in the way that models are constructed. Most traditional languages were based around a standard programming language, such as FORTRAN. Simulation models were built by writing a computer program in the language, compiling it, and running the final executable model.

With Micro SAINT, simulation models are constructed by responding to interactive "menus" (lists of choices on a screen). Modelers do not have to write any computer programs; in fact, no code is generated or compiled. Modeling a system is as easy as using a spreadsheet. The benefits of this easy-to-use system are twofold: If a researcher is new to simulation, he or she will learn Micro SAINT and produce useful results within a few weeks. If a researcher is experienced in simulation, he or she will complete modeling projects much more quickly than with a traditional language. Model maintenance will be greatly simplified as well.

The Develop application is used to construct the various parts of a Micro SAINT model. Tasks are entered by filling in the fields of a menu, as shown in Figure 3. At any time the model builder can type "help," which produces a help screen that explains what is going on and suggests what to do. Micro SAINT models can have up to 400 tasks. Each task can be a single task by itself, or a network of sub-tasks. This hierarchical structure allows an analyst to perform top-down decomposition of the system under consideration. Variables are used to control the model and to keep track of values. At any stage of model development, the software will draw diagrams of the task networks and either display them on the screen or print them out.

Micro SAINT lets analysts use mathematical and logical expressions to establish and control parameters that drive the model (Reference 4). They can use the mathematical functions that are supplied or create their own functions that are stored in a library.

The Execute application runs a completed Micro SAINT model. Five run modes are provided, which range from a fast "batch" mode to a "debug" mode, which shows the event queue and the values of variables. The user may pause a running model, change the values of any variables, resume execution, and watch what happens. The pause feature makes it easy to ask hypothetical questions.



```
                 MODIFY  TASK
Task Number:  409
        (1)  Name:  pilot hears zsu-zsu                  (2)  Type:  Task
        (3)  Upper Network:  400  threat detection
        (4)  Release Condition:  1;
        (5)  Time Distribution Type:  Gamma
        (6)  Mean Time:  2.8;
        (7)  Standard Deviation:  0.9;
        (8)  Task's Beginning Effect:  v = v + 5; MORE
        (9)  Task's Ending Effect:  v = v - 5; MORE
        (10) Decision Type:  Single choice
             Following Task/Network:      Probability of Taking
              Number:       Name:          This Path:
        (11)  410           manual  (12)  1;
        (13)                        (14)
        (15)                        (16)
        (17)                        (18)
        (19)                        (20)
        (21)                        (22)
        (23)                        (24)

Enter number of the field to change or m to modify another job:
```

Figure 3. Micro SAINT Menu

The Analyze application enables the user to create graphs of the data that were collected during model execution. It also calculates and displays statistical information in tabular format.

The Utilities application is used for handling models. The user can copy, delete, or print out models and their complete diagrams. An especially valuable feature is "merge models," which merges two completed models into one. This allows several people to build the parts of a large simulation project, test them independently, and then combine them into the final model.

The most important thing to emphasize about the software is that Micro SAINT is a finished, final product. There are no "features that don't work yet." Micro SAINT is already being used in over 55 U.S. government installations. We have thoroughly tested the software in our own development labs and at user sites. For further information on the operation and capabilities of Micro SAINT, please refer to the **Micro SAINT User's Guide**.

4. The Documentation

With this polished software comes professional documentation. The **Micro SAINT User's Guide** (Reference 4) is a 400-page manual that comes in a three-ring binder. It begins with a tutorial that provides a friendly introduction to the software and to simulation in general. After the tutorial there are sections describing each of the four applications. These sections were written with the beginning user in mind. Each section lists the features of that application alphabetically, and gives examples of their use via screen illustrations taken from the software.

The last two sections of the **User's Guide** have proven to be of great value in helping people build and debug their models. One section explains all the messages that appear in Micro SAINT. Each explanation tells what the message means, and what to do about it. The final section is a collection of eight sample models. These models are (somewhat simplified) examples of how Micro SAINT is used to handle different situations. We have found that these are the best way to convey simulation concepts. Often we get calls from one of our users inquiring how to build a model for a certain problem, and it is easy to direct him or her to a sample model that is closely related.

We have developed a 2-day training seminar entitled, "Introduction to Network Simulation with Micro SAINT." This seminar teaches the skills needed to develop models without requiring previous experience in computer programming or simulation. At the conclusion of the seminar, participants are able to build, execute, and analyze network simulation models that incorporate virtually all of the features of Micro SAINT. We have presented this seminar at a number of government sites over the course of the contract, most recently in July 1987 to 10 people at the Naval Biodynamics Research Laboratory in New Orleans.

5. Potential Applications

In this section we will provide several illustrative examples of Micro SAINT applications that we see as being most relevant to the sponsoring agency. We emphasize that these are examples, and in no way should they be construed as all-encompassing. Micro SAINT is truly a general purpose modeling tool. Its applications are limited only by the creativity of the users.

Drug Effects

Micro SAINT's forte is research in human engineering. The task network methodology that it employs is ideally suited to modeling the activities of a human operator. Micro SAINT was originally designed to provide the Army with a methodology for evaluating the effects of pretreatment drugs or other stressors on human performance. Although we have not conducted studies of this specific type, our experience in related areas leads us to suggest the following procedure for conducting simulation experiments with Micro SAINT.

1. Collect data on the operator's activities. The best way of performing the initial task analysis is to view a videotape of the operation. If a tape is not available, you should interview an SME to construct a task network diagram of the operation that you are studying. Begin with the question, "What is the first thing that you do?" We have found that people can easily understand and correct a Micro SAINT diagram, even though they may have had no prior experience with simulation.

2. Time the videotape with a stopwatch, or ask the SME to estimate times for all the tasks in the model. Also obtain an estimate for the total time. If there are tasks in the operation that can fail, add enough tasks to show the contingency plan and obtain an estimate of the failure probability.

3. Verify the model. When you have built the model (which should take no more than a day or two), ask the SME to check the sequence of tasks and make any necessary corrections. Then run the model a number of times. Micro SAINT stores the simulated time for each run of the model in a data file, and you can examine these times with the Analyze application. The average execution time should be equal to the SME estimate for the total time. If not, you should ask the SME to reconcile the difference.

4. Now your model is ready to be run or exercised with new data obtained from information concerning the effects of a drug or other variable upon some aspects of performance contained within the model. You will need to refer to the Performance Assessment Methodologies (PAM) developed by the JWGD3. The PAM contain data taken from test batteries. These tests indicate how much the time and accuracy of certain tasks are affected by psychopharmacological agents. For each task in your model, pick the task from the PAM it most closely resembles. Each task in your model has a mean time and standard deviation. Replace these two numbers with data from the PAM, preferably with an algebraic expression that gives performance time as a function of drug dosage. For example, the SME estimate of standard deviation

1.2;   (seconds)

might change as a mathematical function of the dosage. If the PAM indicate that the standard deviation increases .053 seconds for every milligram, then the new standard deviation expression would be:

1.2 + dosage * .053;   (seconds)

For those tasks that have a probability of failure, replace the failure probabilities with functions of drug dosage as well.

5. Run the model with various levels of drug dosage. You can give the variable "dosage" an initial value in Micro SAINT's simulation scenario. If the mission is long and the drug is absorbed over time, you can use the continuous variable changes to decrease the value of the variable "dosage" as the model is running.

6. To obtain valid statistical results, you should run the model many times. We have found in our experiments with Micro SAINT that one hundred times is barely adequate to give a smooth distribution; five hundred runs is much better. The Analyze application allows you to create bar charts of the frequency distribution of execution times. Figure 4 shows the data from 500 runs of the M60 tank model. You should pay special attention to the outliers and investigate them individually. Outliers may indicate a change in the mechanism by which the operation is performed. For example, in the M60 model we found that sometimes the tank commander takes over when the gunner cannot see the target, which causes a different set of execution times.

Workload

A second potential application of Micro SAINT is in studying workload. This was the chief focus of the LHX simulation project, and the techniques we developed there can be applied to a wide range of other situations. We have used the following modeling procedure. (For further details, see Reference 2.)

1. Decide how you want to measure workload. In the LHX project we measured workload in four attention channels (visual, auditory, cognitive, and psychomotor). This method is precise, but it is hard to obtain accurate estimates of the attention values required by each task. You may simply want to get a measure of the number of tasks being performed simultaneously over the course of the mission. Or you may want to see how consistently the "background" tasks are performed and how often they are interrupted by other tasks.

2. View a videotape, or interview an SME to construct and validate a Micro SAINT model as explained in the previous section.

3. Each task that represents workload will have a corresponding workload variable to indicate when it is active and when it is not being performed. For example, the tasks that involve navigating should have a variable named something like "navigating."

4. For each task that represents the activity of navigating, add the following algebraic expression to its Beginning Effect.

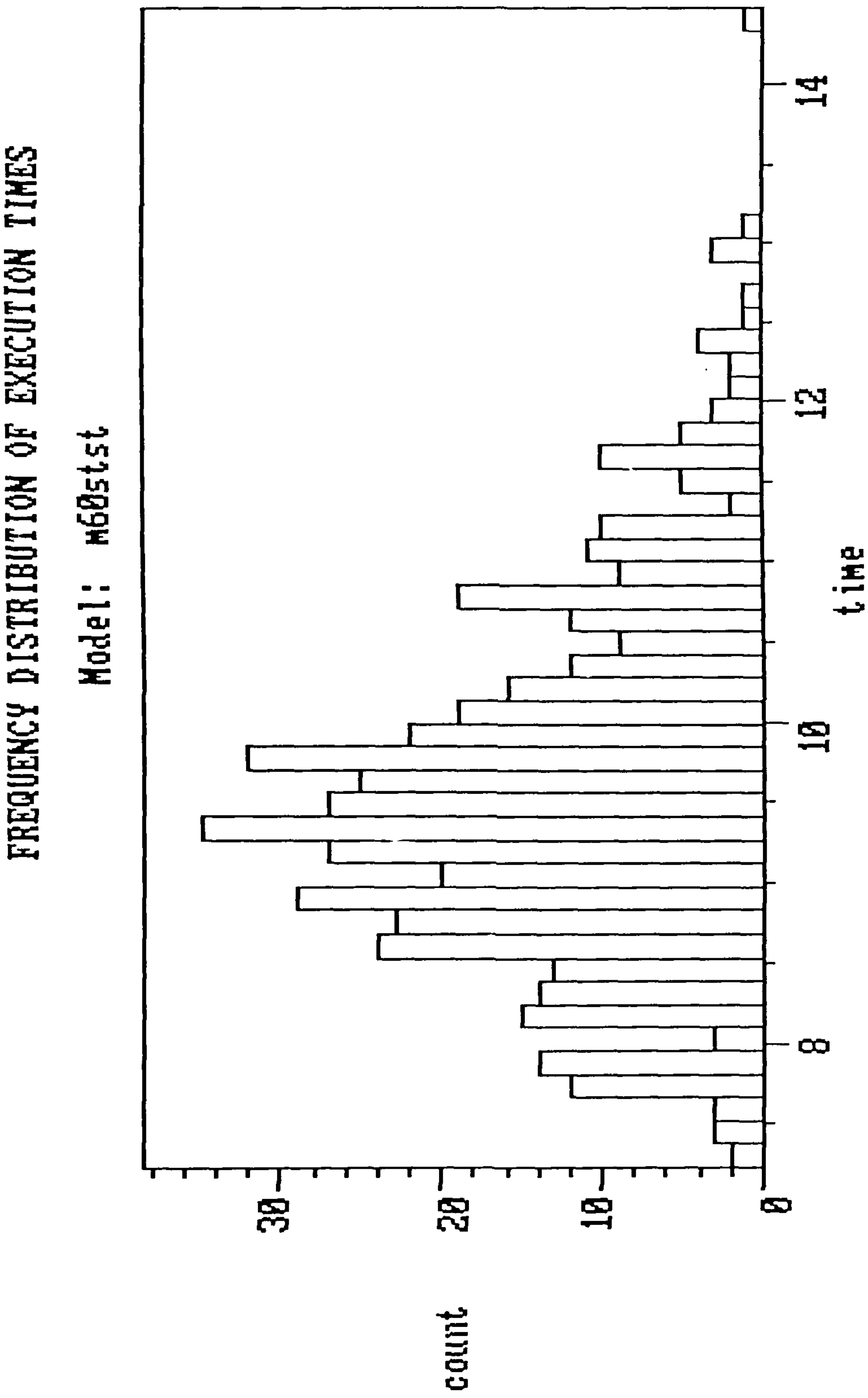


Figure 4. Bar Chart

```
navigating = navigating + 1;
```

Add the following expression to its Ending Effect.

```
navigating = navigating - 1;
```

These two expressions will raise the value of the variable by 1 when the task begins, and lower it when the task ends.

5. Take snapshots of the workload variables at regular intervals over the course of the mission. After executing the model, make a graph of the workload variables with the Analyze application of Micro SAINT. You will be able to see points in the mission at which workload is unacceptably high.

Recently we have been working on a project with Armstrong Aerospace Medical Research Laboratory at Wright-Patterson Air Force Base in Ohio. This effort involves interfacing Micro SAINT with the Generic Systems Analyst Workstation (GENSAW). The goal of this project is to take an IDEF (ICAM (Integrated Computer-Aided Manufacturing) Definition) description of a system from GENSAW and turn it into an executable Micro SAINT model. The final model will be used to examine the workload required by the operators of the system.

Manpower Requirements

The addition of job queues opened up a host of new application areas for Micro SAINT. We have used the following procedure.

1. Instead of performance time or workload, you will be looking at the allocation of resources within a large operation. We will use the daily maintenance of aircraft as an example. Decide at what level you want to model the system. Since you are examining the allocation of maintenance personnel, the tasks in your model should represent the largest job that a group of them can perform as a unit. For example, "check brakes" is the correct level of decomposition, while "turn wrench" is probably too detailed.

2. With the aid of an SME, construct and validate a Micro SAINT model of the operation. You will be taking a high-level approach, and should use someone who is familiar with the overall operation. Instead of modeling the activities that the mechanics perform, focus your attention on the sequence of steps that the airplanes go through. The maintenance personnel will be resources needed to perform these steps.

3. When the task network is complete, put a job queue in front of any tasks for which the airplane must wait if there are not enough maintenance personnel available to service it.

4. Ask the SME how many classes of maintenance personnel there are, and create a variable for each of them. The values of these variables represent how many persons who represent that class are currently available. For each task in the model, ask the SME to estimate how many representatives of each class it takes to perform it.

5. Add the resource utilization to the tasks. Suppose it takes 2 electricians and 1 mechanic to check the electrical system. For the task "check electrical system," add the following Beginning Effect.

```
electricians = electricians - 2;
mechanics = mechanics - 1;
```

Add the following Ending Effect.

```
electricians = electricians + 2;
mechanics = mechanics + 1;
```

These algebraic expressions have the effect of "using up" workers at the beginning of the task, and "freeing" them again at its conclusion.

6. Add a release condition to the tasks that require resource. This condition forces an airplane to wait in front of the next step if there are not enough workers available. The release condition for the task "check electrical system" would be as follows.

```
electricians >= 2 & mechanics >= 1;
```

7. With the simulation scenario, you can set up the number of maintenance personnel that are available at the beginning of the day. After running the model, you will be able to graph the number of available workers over the course of the day, and the number of airplanes waiting in queues. By experimenting with the initial values for the different kinds of personnel, you will be able to determine the desired numbers.

6. Discussion and Conclusions

Micro SAINT is a microcomputer-based tool for constructing models of human performance. It allows medical researchers to conduct simulation experiments without any knowledge of computei programming. Over the course of this contract we have built a variety of models that show how the system can be used in different application areas. Many of these efforts resulted in design changes to Micro SAINT that made it more powerful without compromising its ease of use. The software itself is supported by a comprehensive user's guide and a training seminar. In this report we have identified and described three potential areas of study: drug effects, workload, and manpower requirements.

Micro SAINT is a tool ready to be used in real applications. We believe that it can be a valuable aid to researchers in the Joint Working Group, the Army, and throughout the United States Government. We have delivered the package to several members of the JWGD3 and trained them in its use. The proliferation of Micro SAINT in government installations has already begun, and we would like to see it continue.

7. References

1. Rick Archer, Carl Drews, Ron Laughery, Susan Dahl, Frederick Hegge (1986). Data on the Usability of Micro SAINT. Proceedings of NAECON '86, p. 855-858, Dayton, Ohio.

2. Kenneth Kramme, K. Ronald Laughery, Carl Drews, Rick Archer (1985). LHX Cockpits: Micro SAINT Simulation Study and Results. Report prepared for Texas Instruments, Dallas, Texas.

3. Rick Archer, Susan Dahl, K. Ronald Laughery (1985). Final Report to Universal Energy Systems. Report prepared for Universal Energy Systems, Contract P.O. #773-001, Fort Rucker Field Office, Alabama.

4. Rick Archer, Carl Drews (1987). Micro SAINT User's Guide 3.0. Published by Micro Analysis and Design, Boulder, Colorado.

8. Distribution List

Commander
U.S. Army Medical Research and Development Command
Attn: SGRD-RMI-S
Building 504
Fort Detrick, Frederick, MD 21701-5012