

p-27

Microcomputer Intelligence for Technical Training (MITT): The Evolution of an Intelligent Tutoring System

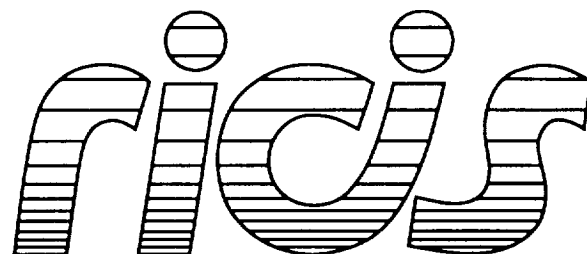
FINAL REPORT

**Jeffrey E. Norton
Bradley J. Wiederholt
William B. Johnson
*Galaxy Scientific Corporation***

October 1990

**Cooperative Agreement NCC 9-16
Research Activity No. ET.11**

**NASA Johnson Space Center
Mission Operations Directorate
Space Station Training Office**



**Research Institute for Computing and Information Systems
University of Houston-Clear Lake**

**(NASA-CR-190180) MICROCOMPUTER INTELLIGENCE
FOR TECHNICAL TRAINING (MITT): THE EVOLUTION
OF AN INTELLIGENT TUTORING SYSTEM Final
Report (Research Inst. for Computing and
Information Systems) 27 p
CSCL 09B G3/61
N92-23091
Unclas
0081273**

TECHNICAL REPORT

The RICIS Concept

The University of Houston-Clear Lake established the Research Institute for Computing and Information Systems (RICIS) in 1986 to encourage the NASA Johnson Space Center (JSC) and local industry to actively support research in the computing and information sciences. As part of this endeavor, UHCL proposed a partnership with JSC to jointly define and manage an integrated program of research in advanced data processing technology needed for JSC's main missions, including administrative, engineering and science responsibilities. JSC agreed and entered into a continuing cooperative agreement with UHCL beginning in May 1986, to jointly plan and execute such research through RICIS. Additionally, under Cooperative Agreement NCC 9-16, computing and educational facilities are shared by the two institutions to conduct the research.

The UHCL/RICIS mission is to conduct, coordinate, and disseminate research and professional level education in computing and information systems to serve the needs of the government, industry, community and academia. RICIS combines resources of UHCL and its gateway affiliates to research and develop materials, prototypes and publications on topics of mutual interest to its sponsors and researchers. Within UHCL, the mission is being implemented through interdisciplinary involvement of faculty and students from each of the four schools: Business and Public Administration, Education, Human Sciences and Humanities, and Natural and Applied Sciences. RICIS also collaborates with industry in a companion program. This program is focused on serving the research and advanced development needs of industry.

Moreover, UHCL established relationships with other universities and research organizations, having common research interests, to provide additional sources of expertise to conduct needed research. For example, UHCL has entered into a special partnership with Texas A&M University to help oversee RICIS research and education programs, while other research organizations are involved via the "gateway" concept.

A major role of RICIS then is to find the best match of sponsors, researchers and research objectives to advance knowledge in the computing and information sciences. RICIS, working jointly with its sponsors, advises on research needs, recommends principals for conducting the research, provides technical and administrative support to coordinate the research and integrates technical results into the goals of UHCL, NASA/JSC and industry.

***Microcomputer Intelligence for
Technical Training (MITT):
The Evolution of an Intelligent
Tutoring System***

FINAL REPORT

Preface

This research was conducted under auspices of the Research Institute for Computing and Information Systems by Galaxy Scientific Corporation, (previously Search Technology). Dr. Glenn B. Freedman and Dr. Christopher J. Dede served as RICIS research coordinators.

Funding has been provided by the Mission Operations Directorate, NASA/JSC through Cooperative Agreement NCC 9-16 between the NASA Johnson Space Center and the University of Houston-Clear Lake. The NASA technical monitor for this activity was Barbara N. Pearson, of the Systems/Elements Office, Space Station Training Office, Mission Operations Directorate, NASA/JSC.

The views and conclusions contained in this report are those of the authors and should not be interpreted as representative of the official policies, either express or implied, of NASA or the United States Government.

**Microcomputer Intelligence for Technical Training
(MITT):
The Evolution of an Intelligent Tutoring System**

**Jeffrey E. Norton
Bradley J. Wiederholt
William B. Johnson**

Prepared by:

**Galaxy Scientific Corporation
2310 Parklake Drive
Suite 300
Atlanta, GA 30345-2904**

Prepared for:

**Air Force Human Resources Laboratory
Training Systems Division
Brooks Air Force Base, Texas 78235-5601**

**RICIS Research Activity Number E.T. 11
Subcontract Number 43**

October 1990

1.0 INTRODUCTION

Microcomputer Intelligence for Technical Training (MITT) uses Intelligent Tutoring System (ITS) technology (Johnson, 1988b; Johnson et al, 1988; Polson and Richardson, 1988; Massey et al, 1988) to deliver diagnostic training in a variety of complex technical domains. Over the past 6 years, MITT technology has been used to develop training systems for nuclear power plant diesel generator diagnosis, Space Shuttle fuel cell diagnosis, and message processing diagnosis for the Minuteman missile. This paper presents an overview of the MITT system, describes the evolution of the MITT software, and describes the benefits of using the MITT system.

2.0 OVERVIEW

MITT is an Intelligent Tutoring System (ITS) for technical training. As an ITS, MITT relies upon internal models of the student, instructor, and expert performance to provide appropriate feedback to the student. MITT monitors student performance and actions (the student model) and compares these actions to the task expert actions (the expert model). Based on the difference between the student and expert actions, MITT provides remediation and instruction accordingly (the instructor model).

In addition to the student, instructor, and expert performance models, ITSs also contain an instructional environment which focuses on the domains and tasks to be learned by the students. Instructional techniques can range from sophisticated simulation to straight forward drill and practice. MITT contains a simulation-oriented instructional environment which focuses on diagnostic-related technical training. To diagnose and troubleshoot a simulated system, MITT allows a student to inspect parts, look at gauges, ask for advice, and replace faulty components.

The MITT approach is useful for developing ITSs for systems with the following characteristics:

- o System is technically complex,
- o System has varied instrumentation,
- o System has multiple test points,
- o System requires practice to develop and maintain proficiency,
- o Troubleshooting procedures exist or can be developed for the system,
- o Access to the real equipment is limited, and
- o The student's job requires diagnostic tasks.

The MITT system consists of two separate environments: MITT Writer and MITT Tutor (see Figure 1). MITT Writer is a development environment used by instructors and subject matter experts to produce a description of training. In contrast, MITT Tutor delivers the instruction that

was developed using MITT Writer.

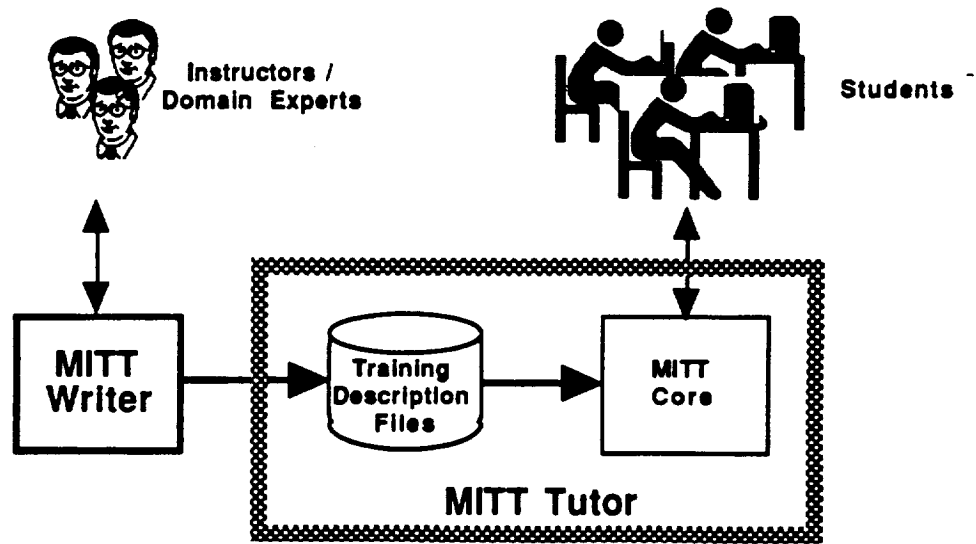


Figure 1. MITT System

The goal of MITT Writer is to support instructors and subject matter experts, rather than programmers, in developing MITT Tutors. MITT Writer provides enhanced ITS authoring capabilities such as support for constructing simulations displays and expert procedures, graphical interactive construction of ITS components, and multiple levels of advice on the methodology for constructing MITT Tutors. MITT Writer allows the user to rapidly develop and maintain MITT Tutors.

The goal of MITT Tutor is to deliver training to students using ITS technology. MITT Tutor uses the data produced by MITT Writer to present problems, monitor student performance, compare student actions to expert actions, and present suitable feedback to the student. MITT Tutor can work with any set of training description files produced by MITT Writer.

Both the MITT Writer and MITT Tutor environments require modest, standard computing platforms. Both environments run on IBM/AT-class personal computers, a platform used widely in the Air Force technical training community. In contrast to other platforms used for ITS development, the IBM/AT platform yields lower hardware cost, a wider base of user experience and familiarity, and increased availability in the training community. On the other hand, these platforms carry the extra development burden of greater attention to memory management,

processor speed, disk access time, graphics display, and other constraints (Neste, 1989). Both MITT environments require an IBM/AT-class platform, enhanced graphics adaptor (EGA) display, and 640 kilobytes (Kb) of base memory. The MITT Writer environment carries the extra requirement of 1 megabyte (Mb) of expanded memory. Both environments are enhanced greatly through the use of a mouse input device, though this accessory is not strictly required.

3.0 EVOLUTION

MITT capitalizes on technology developed by the authors and others over the past ten years (Johnson, 1981; Johnson & Rouse, 1981; Johnson, 1987; Coonan, et al, 1990). This section describes the evolution of the technology that now comprises MITT.

3.1 Supporting Research

The current MITT system evolved from previous experimental evaluations and computer simulations for technical training (Johnson, 1981; Rouse and Hunt, 1984). The first of the evolving simulations was name Troubleshooting by Application of Structural Knowledge (TASK). This simulation provided a variety of context-free simulations that enhanced our knowledge of basic problem-solving. The TASK simulation was used in a variety of experiments that varied characteristics such as the kind of feedback given, the use of computer-aiding, forced pacing, and problem complexity (Rouse, 1979).

The Framework for Aiding and the Understanding of Logical Troubleshooting (FAULT) allowed the user to troubleshoot the network by checking instruments, making observations between parts, and replacing parts. FAULT represents each part in the system as a node in a network. Lines connecting the nodes represent input/output relationships among the parts. The diagram that shows this relationship is called a functional flow diagram (FFD). See Figure 2.

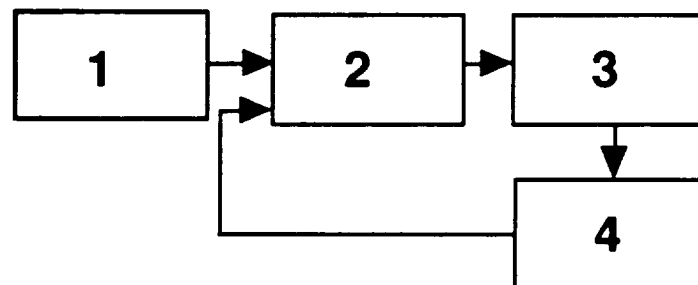


Figure 2. Functional Flow Diagram (FFD)

The FFD helps the student think about the parts and functions of the system. The diagrams are meant to depict the functional (but not necessarily physical) relationships among the parts. On the FFD, function flows from one part to the next in the direction of the arrows. In order for a part to have acceptable, functional flow (output) from it, the part itself and all inputs must also be acceptable. The simulation permits the student to test flow to find the failed component.

TASK and FAULT lead to a number of experimental evaluations, mostly in the university laboratory environment (Johnson, 1981; Rouse & Hunt, 1984). The first serious application of the concept was at the U.S.Army Signal School at Fort Gordon, GA (Johnson & Fath, 1983 & 1984). At the Signal School a graphical interface was added to increase the physical fidelity of the simulation. An experimental evaluation showed that the training method was valuable for teaching electronic troubleshooting.

3.2 Software Versions to be Compared

The MITT system has evolved through the following revisions of the software: DGSIM, MITT Prototype, MITT Fuel Cell Tutor, MITT Missile Tutor. This section describes each version of the system by comparing the different implementations of the features listed in Table 1. Table 2 summarizes these different implementations.

Table 1. Dynamic MITT Features

Functional Advice	The advice that is given to the student is based on the functional flow diagram (FFD). Functional advice knows only that part A is connected to part B, etc. No domain-specific advice can be given here.
Simulation	The term simulation used here refers to how the data values used by the system are updated. These data values can be generated in a variety of ways ranging from the use of static data representations to the use of a full-fidelity simulation.
Animated Flow	The function of the system flows from one part to the next. The flow from one part to another part can be represented with some simple animation on the screen.
Procedural Advice	Many technical training domains enlist specific troubleshooting procedures. The procedural advice captures this procedural knowledge in the form of rules.
Graphics	The interface to MITT uses a large number of graphics. As the system has developed, the resolution of the pictures and the manner in which these graphics are presented has changed.
Time	The amount of <u>simulated</u> time required to perform a test or to replace a part.
Instructor Feedback	The unsolicited feedback is similar to the feedback that would be given by a human instructor looking over the student's shoulder.
Subsystems	The size of the domain depends on two things: (1) the size of the executable system, and (2) the size of the domain-specific files. Sometimes the domain needs to be subdivided into several smaller systems in order to fit into available memory. Each of these systems are referred to as "subsystems".
Memory	The need for memory conservation is essential because of the 640 kilobyte memory restriction. Several different tactics were used to conserve memory at different points in the evolution of MITT.
File-Based	In preparation for MITT Writer, all domain-specific information was removed from the code and put in data files. "Yes" indicates that this was done, "No" indicates that the code still contained domain-specific references.

Table 2. MITT History

	DGSIM	MITT Proto	Fuel Cell	Missile	MITT Core
Functional Advice	Includes time, most powerful, & best test	Includes only best test	Same	Same	Same
Simulation	System is a snapshot. Data values are constant	Same	Adds dynamics to the simulation	Same	Same
		Adds link between mouseable regions & tests	Same	Same	Same
Animated Flow	None	None	Adds animation	Same	Same
Procedural Advice	None	Used CLIPS	Same	Expands student model	Same
Graphics	CGA 4-color	EGA 640x200. Uses custom decompression routines	Refines decompression routines	Uses 3rd party PCX package EGA 640x350	Same
Time	Tracks time required to perform tests	No longer tracks time	Same	Same	Same
Instructor Feedback	Limited to FFD advice	Adds unsolicited feedback (beyond FFD advice)	Same	Same	Same
Subsystems	Multiple subsystems	Same	Single subsystem	Same	Same
Memory	One executable	Four separate executables (Main, Info, Sim, CLIPS)	Same	Two executables Info/Sim and CLIPS	Overlays yield one executable
File-Based	No	No	No	Yes	Yes

3.2.1 DGSIM

DGSIM (Diesel Generator Simulator) is the foundation of the current MITT Tutor. DGSIM was developed for the Electric Power Research Institute working in conjunction with Duke Power Company (Johnson et al, 1986; Maddox et al, 1986; Johnson et al, 1985). The system provided diagnostic training for the emergency standby power system - a diesel powered generator. A FFD for one diesel generator subsystem is shown in Figure 3.

The DGSIM skeleton was modified and enhanced to create MITT. For this reason, this discussion begins with a description of DGSIM, centered around the features listed in Table 1.

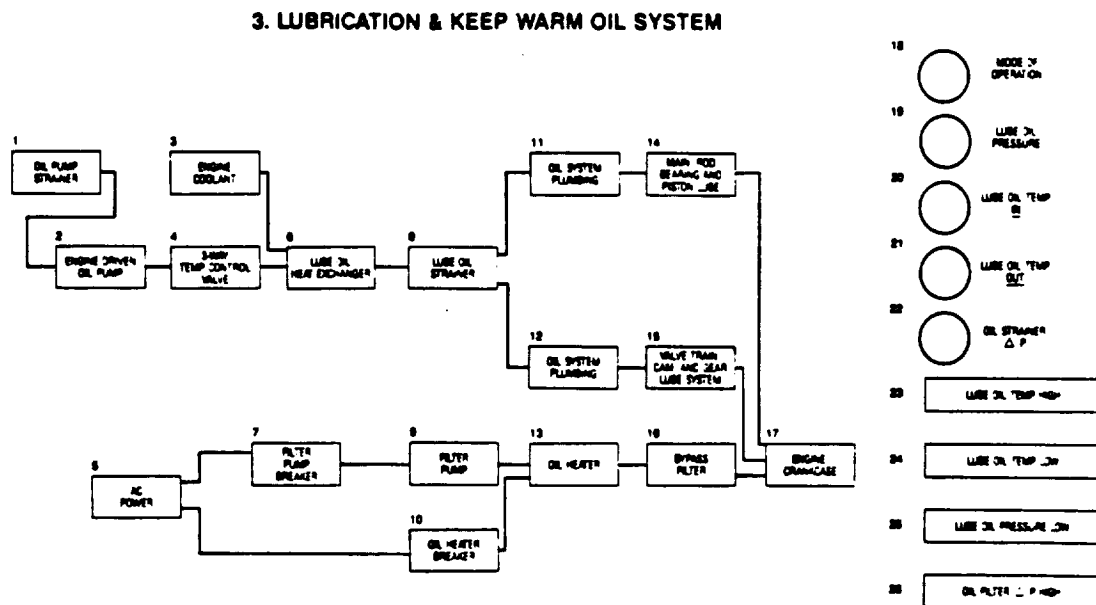


Figure 3. DGSIM Functional Flow Diagram

As mentioned earlier, the FFD concept was an important component in providing generic troubleshooting advice. The functional advice suggests tests that are based on the following criteria: Least Time, Most Powerful Test, and the Best Test. The Least Time advice suggests the test that requires the least amount of time to perform. The Most Powerful advice suggests the test that provides the maximum information gain regardless of the cost. The Best Test advice suggests the test that provides the maximum information gain, while being sensitive to cost.

The data used by DGSIM was static. The gauge readings remained constant throughout the session. When a gauge or panel appeared, all data was immediately displayed.

The graphics for DGSIM were written for a Color Graphics Adapter (CGA) 4-color graphics card. At the time, CGA was the accepted graphics standard. The graphics were displayed using custom decompression software.

As the student performed a test, the simulated time required to perform each test was displayed (e.g. a test may require 45 minutes to perform in the real world). A running total of these times and the number of logical diagnostic errors was kept as a metric of a student's performance.

The combination of the code size and the amount of data presented dictated that the domain be separated into several subsystems. These subsystems were subdivided according to function. For example, the diesel generator contained a Control subsystem and a Cooling/Keep Warm subsystem. The practice of using different subsystems allowed the system to run as one executable file with the supporting subsystem files.

The features Procedural Advice, and Animated Flow were not part of the DGSIM design. These features were added as part of MITT.


3.2.2 MITT Prototype

The initial MITT prototype was a proof-of-concept to show that an effective ITS could be developed at a low cost on an ordinary microcomputer. The domain for the prototype was the Fuel Cell/Electrical Power System for the Space Shuttle. The project spanned 6 months - a one person-year effort.

The functional advice was reduced to include only advice for the Most Powerful test. The advice suggested the test that eliminated the most parts from the feasible set.

The data used by the Prototype was also static. The gauge readings remained constant throughout the session. NASA expressed great interest in providing some sort of simulation to generate dynamic data values. The Prototype did not provide the opportunity to accomplish this, but this was implemented in the follow-on work (MITT Fuel Cell Tutor).

The Prototype added regions on the gauges that were covered by X's ("hidden"). When the student selected the X's, a data value for the particular sensor appeared. This allowed the system to know exactly what the user should have known based on specific observations. This was particularly helpful for displays with a large amount of data. Additionally, some of these "mouseable" areas were linked to equivalent tests. For example, looking at temperature gauge A is equivalent to testing the output of part 5. See Figure 4.

Temperature			Temperature	
Fuel Cell A	XXX	 <div data-bbox="613 615 771 730" style="border: 1px solid black; padding: 2px;"> mouse click yields ... </div>	Fuel Cell A	387
Fuel Cell B	XXX		Fuel Cell B	XXX
Fuel Cell C	XXX		Fuel Cell C	XXX

**Which is
equivalent to
testing the
output of part 5
in FFD.**

Figure 4. Hidden Fields and Test Equivalencies

The animated flow was not part of the MITT Prototype. However, NASA suggested that this animated schematic would be helpful. It was implemented in the next phase.

One of the most important improvements to MITT was the addition of Procedural Advice. Procedural Advice captured established troubleshooting procedures for the Fuel Cell domain. The advice suggests the next action to perform, based on common troubleshooting procedures for the domain. This troubleshooting knowledge is represented in a rulebase as part of the C Language Integrated Production (CLIPS) expert system. CLIPS was developed by NASA and was free to all developers of government software. CLIPS is distributed as part of the MITT Tutor free-of-charge. CLIPS gives the MITT user the power of an expert system without the extra cost that would have been required for a commercial expert system.

The MITT Prototype was developed for an Enhanced Graphics Adapter (EGA) 16-color card. Initially, the graphics used 640x350 resolution. However, after further discussion at a design review, it was determined that a larger student population could be reached by using 640x200 resolution. Once again, custom decompression routines were used to display the pictures.

The simulated time to perform a test was no longer tracked. The additional functionality in the memory-cramped environment dictated the elimination of some previous features; time tracking was one. However, a stopwatch was still run for each student session.

The Prototype also added unsolicited instructor feedback. This feedback is similar to the suggestions that would be given by an instructor looking over a student's shoulder during a session. This feedback gives small hints to the student on how to effectively use the Tutor. This should not be confused with the expert Procedural and Functional advice.

The Prototype also used multiple subsystems. The Fuel Cell required two systems: a Cryogenic subsystem and a Thermal Control subsystem. The student could change subsystems at any time during a problem. Once again, multiple subsystems were used to conserve memory. However, the additional functionality required other memory adjustments.

The system was partitioned into three separate executable files. These files were functionally partitioned as follows: System Information, Simulation, and Procedural Advice. The System Information executable presented information on how to run the system, how to use the mouse, and a description of the domain. The Simulation presented the problems to the student for troubleshooting. The Procedural Advice module presented suggestions on the best course of action based upon procedural knowledge. The use of three different executable files was transparent to the user.

3.2.3 Evaluation of MITT Prototype

The MITT Fuel Cell Prototype was evaluated with a formal user acceptance test at Johnson Space Center. A group of astronauts, flight controllers, and technical training personnel evaluated the tutor. Post-use interviews and questionnaires showed that the Prototype was an acceptable, viable method to supplement diagnostic training for the Fuel Cell.

The user acceptance evaluation revealed a few deficiencies in the interface, as well as in technical data associated with various failure scenarios. The most common request was that the system be enhanced to include additional failure scenarios.

3.2.4 MITT Fuel Cell Tutor

The Fuel Cell Tutor enhanced the Prototype to include additional failure scenarios. It also

incorporated the suggestions made by NASA personnel during development of the Prototype, as mentioned in the previous section.

One of the most important changes to the Prototype involved the simulation. Previously, the system used static data. The Fuel Cell Tutor incorporated a "dynamic" simulation. The dynamics involved updating sensor values at specified intervals. A simple, memory-efficient look-up scheme provided the same effect as a complex, mathematical simulation. Data was refreshed on the displays at 5-second intervals.

Another new addition to the Fuel Cell Tutor involved the schematic that was included in the Fuel Cell Information section. The Prototype included the schematic of the system where the student could select a part and get a description. However, the Fuel Cell Tutor added animation to this display (See Figure 5). The subject matter experts felt that it was important to show how the function flowed from one part to the next part in the system. Lines moved across the screen to show the flow.

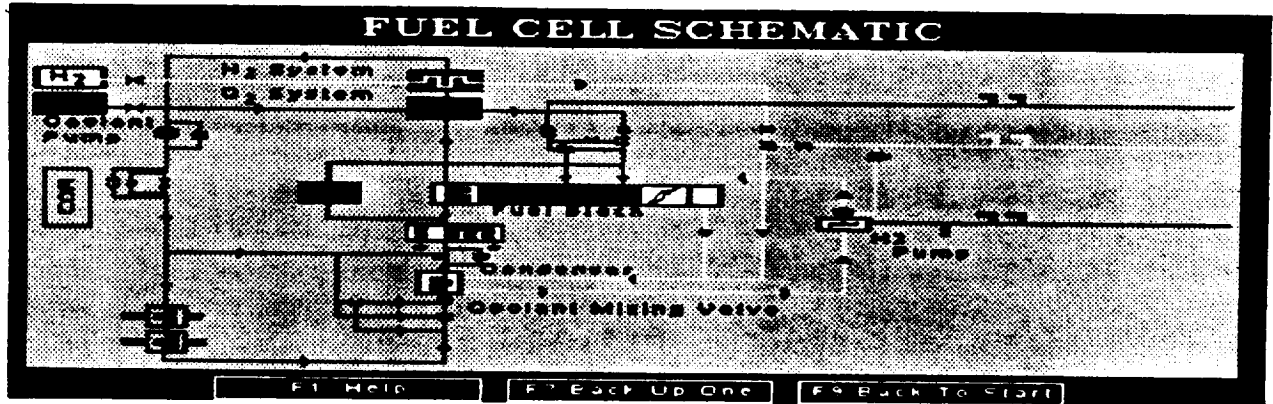


Figure 5. Fuel Cell Schematic

The EGA 640x200 graphics remained in the Fuel Cell Tutor to insure compatibility with available computers at NASA. Also, the graphics decompression routines were optimized. These refinements allowed a smoother and faster transition from one screen to the next screen.

Other portions of the code were also optimized. This allowed the Cryogenic Subsystem and the Thermal Control Subsystem to be combined to form a single system. The student no longer needed to load each subsystem explicitly.

The functional advice, test equivalencies, procedural advice, and instructor feedback remained essentially the same. The need for three separate executables did not change for this phase of MITT either.

3.2.5 MITT Missile Message Processing Tutor

The Missile Tutor project showed that the MITT approach could be applied to other technical domains. The domain chosen was the Message Processing System for the Minuteman Missile (See Figure 6). The majority of the work on this project involved the knowledge engineering for the domain-specific information. Changes were also made to increase the graphics resolution from 640x200 to 640x350. The student model was enhanced to support Procedural advice. All work was performed in parallel to the development of MITT Core (described in section 3.3).

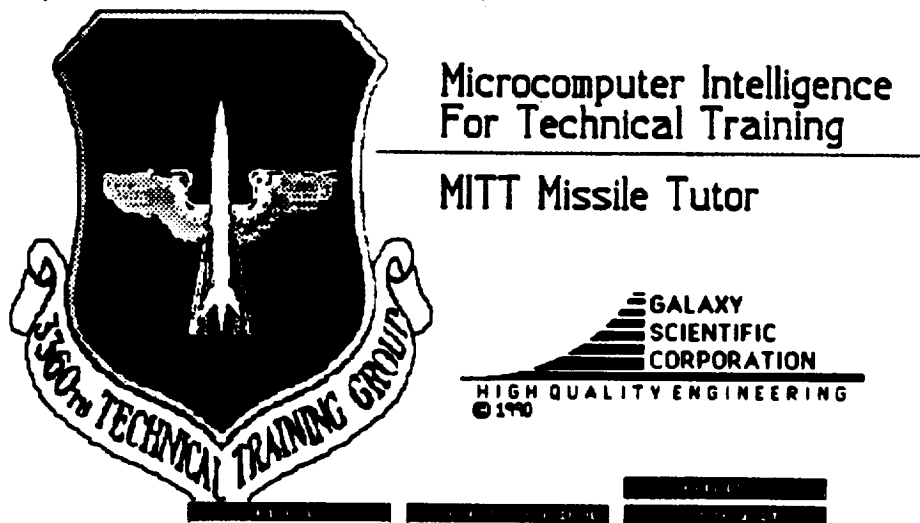


Figure 6. MITT Missile Tutor

The requirements for the Missile domain were very different than the requirements for the Fuel Cell domain. Therefore, this was an excellent test of the generic nature of MITT. The first difference involved the manner in which data was presented to the student in each system. The Fuel Cell presented many video displays or gauges to the students. To receive information about

the system, the student only had to look at the display or gauge. In contrast, the Missile had very few instruments that continuously presented data to the student. Instead, the student is forced to decide not only what equipment to look at, but also how to get the needed data value.

For example, if the astronaut suspected trouble with the Fuel Pump, information would be available on one of the displays or gauges. In contrast, if the Missile student suspected trouble with a drawer, the student would have to indicate which connector to remove and know where to attach a piece of monitoring equipment (See Figure 7). This demanded a much more active role of the student.

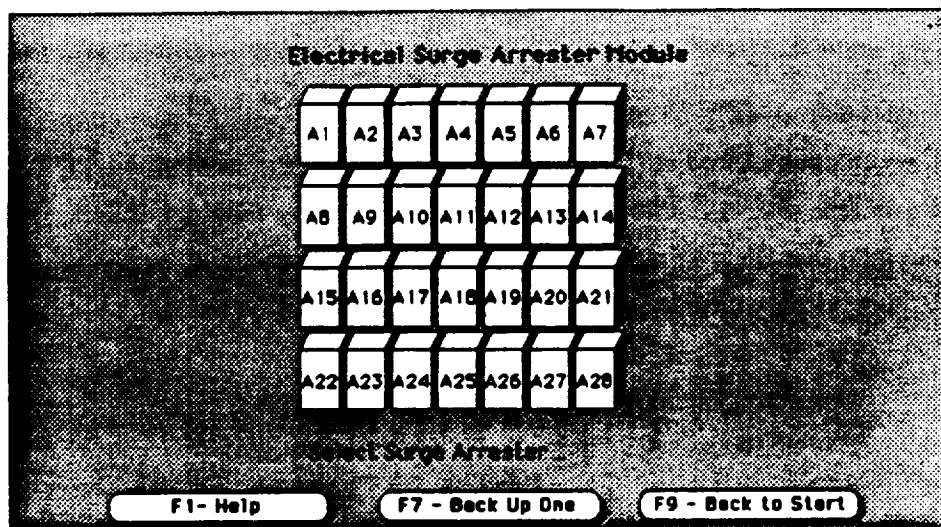


Figure 7. MITT Missile Tutor Screen

The MITT architecture supported the more active role, even though the Missile Tutor seemed to require a more interactive interface. Displays were used to display data (as before), but were also used to collect data (e.g. designating the connector to inspect). The increased fidelity forced trade-offs between memory and functionality.

3.3 MITT Core / MITT Tutor

The first step toward developing the authoring system (MITT Writer) involved taking the Fuel Cell code and separating the domain-specific knowledge from the instructional environment. Referring back to Figure 1, this refers to the Training Description Files and MITT Core respectively. Our strategy was to create a generic, domain-independent "engine" (MITT Core) that

could be used with domain-specific data files. These files, rather than the program, would be modified in order to create a new ITS.

As domain-specific knowledge was removed, the code was optimized so that only two executable files were needed - a combined Information / Simulation module and a separate executable file for Procedural Advice. This two executable file architecture was used for the Missile Tutor. However, after the delivery of the Missile Tutor, compiler overlay techniques were used to produce a single executable file that operates within the 640 kilobyte limit. The overlay of the two modules resulted in a savings of about 200 kilobytes of memory. This extra memory is now available for additional domain knowledge.

3.4 MITT Writer

3.4.1 Design Goals for MITT Writer

The initial acceptance of the Fuel Cell and Missile tutors suggests that the MITT Tutor architecture is a reasonable alternative for simulation-based diagnostic training. In order for the MITT tutoring system to be widely accepted into DOD and other government and/or industry training, there are three primary criteria that must be met. They are listed below.

- o Delivers training on available computers
- o Delivers challenging training / simulation scenarios
- o Has reasonable development and maintenance costs

If MITT Tutors are going to have wide-spread acceptance, they must operate on computer systems that are already installed in training installations. Such computers are 80286-based processors with 640 kilobytes of RAM. EGA color displays, combined with keyboard and mouse inputs, are quite common. Therefore, the MITT Tutors are designed to operate within these constraints.

MITT Tutors must be able to provide challenging, simulation-based problems to trainees. The architecture used for the Fuel Cell and Missile Tutors is simulation-based, permitting the student to engage in most of the diagnostic actions that are available with real equipment. This design must be available for any new MITT Tutors.

The development and maintenance of MITT Tutors must be affordable. This goal can be achieved with the creation of a development environment that can be used by government training

developers and subject matter experts. Technical instructors are likely users of such development tools (Johnson, 1988a). The MITT Writer authoring environment must guide technical personnel to organize their knowledge in a format that is aligned with the MITT Tutor architecture. The system must be relatively easy to use. It must guide the developer, provide help/advice with the development, and check the newly-created system for completeness. The MITT Writer system, proposed earlier (Johnson, Neste, and Duncan, 1989) and described below, is designed to fulfill these criteria.

3.4.2 MITT Writer Description

The authoring environment for MITT is called MITT Writer (See Figure 8). MITT Writer allows the author to design and edit domain-specific files for use by MITT Tutor. The system supports direct manipulation through a graphical interface. Multiple levels of validation are offered including correctness, completeness, and consistency.

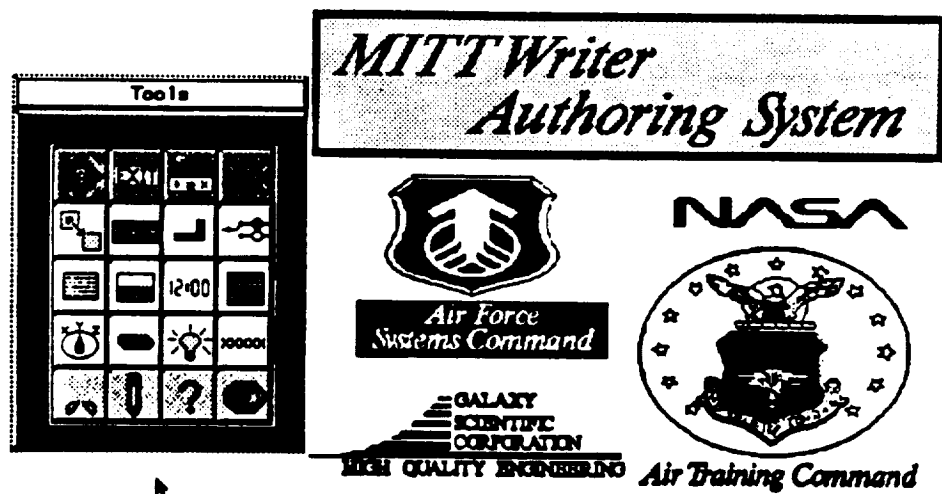


Figure 8. MITT Writer Authoring System

The architecture of MITT Writer is shown in Figure 9. The system contains the following modules: Display Manager, Main Routine, Advice System, Object Editors, Consistency Checker, Help System, Database Support, and MITT Tutor File Support.

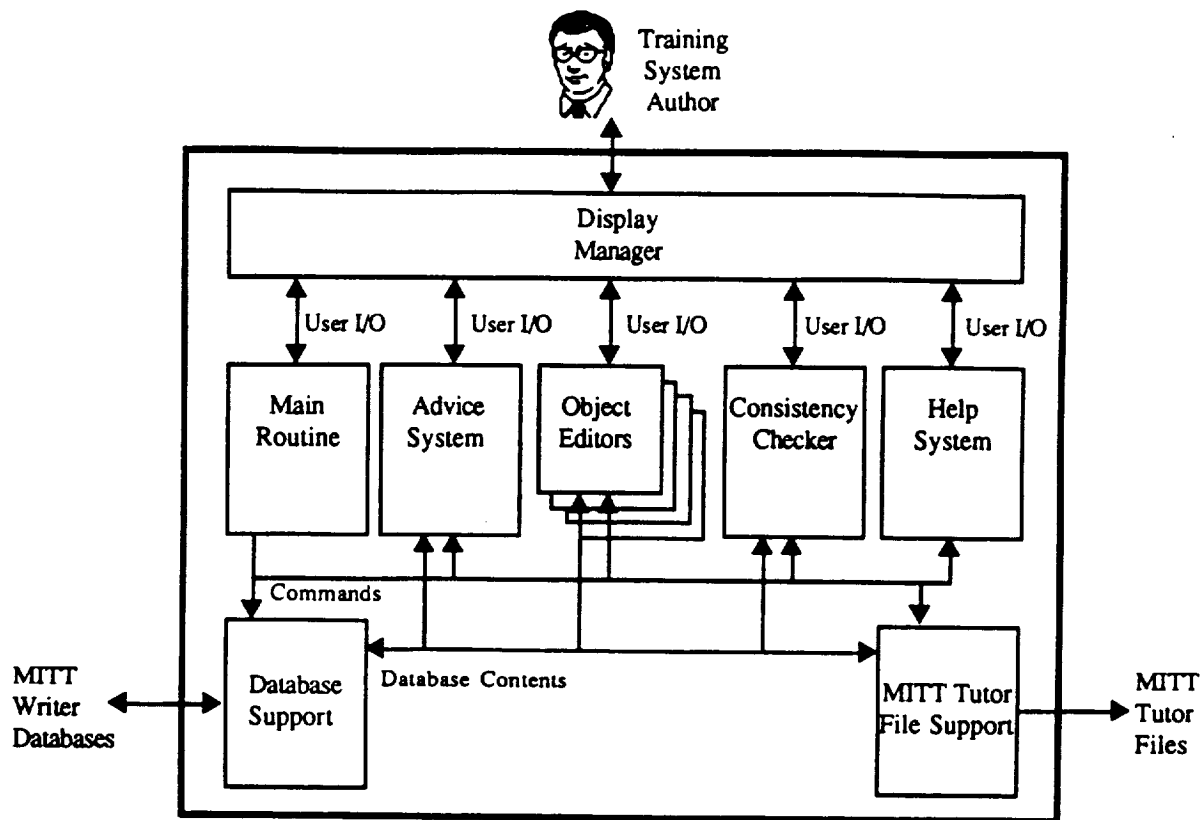


Figure 9. MITT Writer Internal Architecture

The Display Manager is an intermediary between the author and the rest of the MITT Writer modules. The interface to the system is window-oriented and requires a manager to transfer control from one window to the next window. The object-oriented design centralizes interface management routines.

The Main Routine is responsible for initializing and terminating each session. It also maintains information about each session as well as information about the current state of the database.

The Advice System tutors the user on the MITT Writer authoring procedure. The embedded advisory expert system compares the state of the database to the "expert" authoring procedure. The "expert" authoring procedure evolved from over five years of experience in developing MITT tutors.

The Object Editors allow the author to add objects to the system. Objects exist for the training system description, system components, system connectivity, malfunctions, faults, alarms, instructor model, procedures, interface screens, and screen display elements. The editors are both text-based and graphics-based. The graphics that are used by MITT Writer must be generated in outside of MITT Writer.

Figure 10 shows one of the text screens used in the screen editor. This screen specifies general information about a simulation screen. This includes the screen name, the graphics file to be shown, and controls to modify the help text and the refresh rate for this screen. The interactive display editor allows the user to add links, data fields, buttons, etc. to the graphics screen.

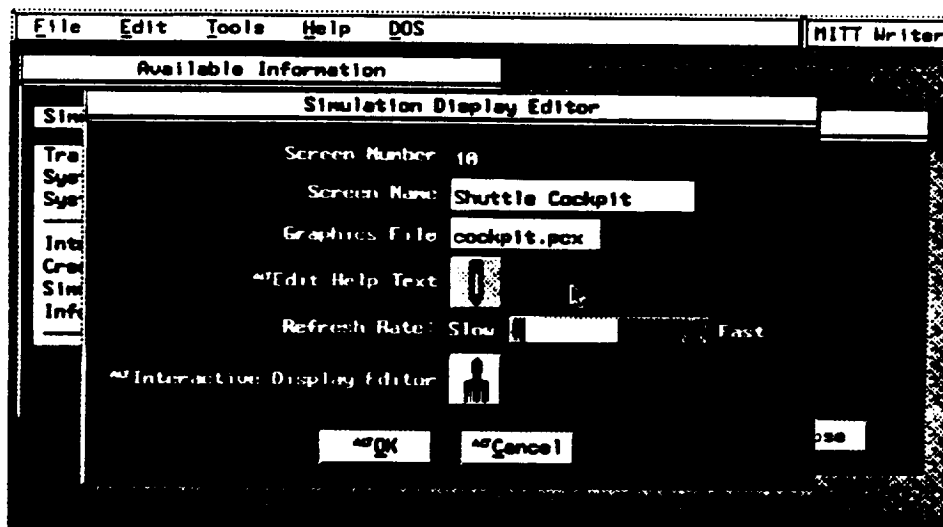


Figure 10. Sample MITT Writer Object Editor Screen

The Consistency Checker is used to ensure that the files that are written by MITT Writer will be readable by MITT Tutor. It compiles a list of such inconsistencies with two levels - warnings and errors. Warning are minor inconsistencies that would result in an incomplete MITT Writer database; however, this database could still be run in MITT Tutor. Errors are not compatible with MITT Tutor and would have to be corrected before attempting to run MITT Tutor. An example error message reads "Simulation Screen 'Main Menu' linked to undefined screen

'Check Voltage"', while an example warning message is "*Sensor #1 is defined without an equivalent test defined*".

The Help System gives the user both orientation help and context-specific help. Orientation help gives general information about MITT Writer, how to use the menus, etc. Context-specific help lets the user know about the current activity (e.g. how a particular editor is used).

The Database Support module operates on the MITT Writer database. This support allows both random access and sequential access to the database. The implementation of this support provides quick access to the database with minimal memory overhead. Database support is provided for authored objects that the author creates, as well as for system help and advice that is given to the author.

MITT Tutor File Support scans the MITT Writer database to produce the training files used by the MITT Tutor. This support ensures that the database is consistent, but not necessarily complete before translation. Completeness is assured through consultation with the MITT Writer Advice System

4.0 BENEFITS OF THE MITT APPROACH

The MITT system is attractive for the following reasons: (1) it provides enhanced ITS authoring capabilities, (2) it can reduce development costs, and (3) it can reduce delivery costs.

The MITT system is attractive because it uses proven ITS technology. MITT Writer provides support to construct simulations, student interfaces, instructional strategy, and expert advice to students in an interactive, graphical environment. The author is not only given these tools, but is advised on a method for developing new training description files for use by the MITT Tutor.

The development costs are reduced because MITT Writer allows for rapid development and easy maintenance of an ITS. The authoring system is designed so that the subject matter expert can also be the developer. Development of a new MITT domain does not require programmers. Once a new domain is developed, MITT Writer supports easy modification of the domain. Therefore, as specifications for a system change, the training system can be modified also. There is no need to

start over. Also, MITT Writer will provide a standard instructional environment for the technical students.

The delivery costs are reduced because the system is designed to run on low-cost, readily-available microcomputers. The system will run on standard hardware available today in the government and DOD training centers. The Air Force receives a powerful training tool without the expense of dedicated training delivery workstations.

5.0 NEXT STAGE OF MITT EVOLUTION

The first user workshop for MITT Writer is scheduled for early 1991. The expectation is that a series of new MITT Tutors will emerge; each more creative than the previous one. The Air Force Human Resources Laboratory, in conjunction with Air Training Command, will support MITT Writer with orientation workshops and user-suggested enhancements. The result will be a variety of cost-effective MITT Tutors that provide simulation-based technical training throughout the Department of Defense and other United States Government agencies.

6.0 ACKNOWLEDGEMENTS

The authors would like to thank the following personnel for their continued support throughout the development of MITT. Their guidance, ideas, and patience were crucial to the success of MITT.

USAF-HRL: Dr. Wes Regian, Lt. Col Hugh Burns, Lt. Charles Capps, Capt. Kevin Kline, Capt. Michael Slaven, Capt. Kevin Glass

USAF-ATC: Capt. Ed Arnold, Capt. Archie Smith, Mr. Ernesto Podagrosi, MSgt. Wayne Griffin, MSgt. Rick Olsen

NASA: Barbara Pearson, Lt. Dan Soderlund, Ken Swiatkowski, Ronald Lee, Sean Kelly

Search Technology: Dr. Philip C. Duncan, Dr. Michael E. Maddox, Dr. Ruston M. Hunt, Dr. William B. Rouse

RICIS: Dr. Nancy Bell, Dr. Glenn Freedman, Dr. Glen Houston

Galaxy Scientific: Les Neste, Lica Browning, Bill Pitts, Leo Utsman

7.0 REFERENCES

Coonan, T.A., Johnson, W.B., Norton, J.E., & Sanders, M.G. (1990). A hypermedia approach to technical training for the electronic information delivery system. *Proceedings of the Eighth Conference on Instruction Delivery*. Warrenton, VA: Society for Applied Learning Technology.

Johnson, W.B. (1981). Computer simulations for fault diagnosis training: An empirical study of learning from simulation to live system performance. (Doctoral Dissertation, University of Illinois, 1980), (*Dissertation Abstracts International*, 4111). 4625-A. (University Microfilms No. 8108555).

Johnson, W.B. (1987). Development and evaluation of simulation-oriented computer-based instruction for diagnostic training. In W.B. Rouse (Ed.), *Advances in man-machine systems research: Vol. 3*. Greenwich, CT: JAI Press, 99-125.

Johnson, W.B. (1988a). Developing expert system knowledge bases for technical training. In L.D. Massey, J. Psotka, and S.A. Mutter (Eds.), *Intelligent Tutoring Systems: Lessons Learned* (pp 83-92). Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Johnson, W.B. (1988b). Intelligent tutoring systems: If they are such good ideas, why aren't there more of them. *Proceedings of the 10th Annual Interservice/Industry Training Systems Conference*, Orlando, FL: The Industrial Security Association, 399-406.

Johnson, W.B. and Fath, J.L. (1983). Design and initial evaluation of a mixed-fidelity courseware for maintenance training. *Proceedings of the 27th Annual Meeting of the Human Factors Society* (pp 1017-1021). Norfolk, VA: Human Factors Society

Johnson, W.B. and Fath, J.L. (1984). *Implementation of a mixed-fidelity approach to maintenance training* (TR-661). Alexandria, VA: U.S. Army Research Institute for the Behavioral and Social Sciences.

Johnson, W.B., Maddox, M.E., Rouse, W.B., & Kiel, G.C. (1985). *Diagnostic training for nuclear power plant personnel, volume 1: Courseware development* (EPRI NP-3829). Palo Alto, CA: Electric Power Research Institute.

Johnson, W.B., Neste, L.O., and Duncan, P.C. (1989). An authoring environment for intelligent tutoring systems. *Proceedings of the 1989 IEEE International Conference on Systems, Man, and Cybernetics*. Boston, MA, 761-765.

Johnson, W.B., Norton, J.E., and Duncan, P.E., and Hunt, R.M. (1988). *Development and demonstration of an intelligent tutoring system for technical training (MITT)* (AFHRL-

TP-88-8). Brooks AFB, TX: The Air Force Human Resources Laboratory.

Johnson, W.B., and Rouse, W.B. (1981). Analysis and classification of human errors in troubleshooting live aircraft power plants. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-12,(3),389-393.

Johnson, W.B., Wiederholt, B.J., and Maddox, M.E. (1986). *Diagnostic training demonstration: Instructor and student manuals and sample diskettes* (EPRI NP-4493P). Palo Alto, CA: Electric Power Research Institute.

Maddox, M.E., Johnson, W.B., & Frey, P.R. (1986). *Diagnostic training for nuclear power plant personnel, volume 2: Implementation and evaluation* (EPRI NP-3829-II). Palo Alto, CA: Electric Power Research Institute.

Massey, L.D., Psotka, J., Mutter, S.A. (Eds.) (1988), *Intelligent Tutoring Systems:Lessons Learned*. Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Neste, L.O. (1989). Overcoming microcomputer constraints in the development of an intelligent tutoring system. *Fall Symposium on Computing Research and Development*. Houston, TX: University of Houston Clear Lake, Research Institute for the Computing and Information Sciences.

Polson, M.C., and Richardson, J.J. (Eds.) (1988). *Foundations of Intelligent Tutoring Systems*. Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Rouse, W.B. (1979). Problem solving performance of first semester maintenance trainees in two fault diagnostic tasks. *Human Factors*, 21(5), 611-618.

Rouse, W.B. and Hunt, R.M. (1984). Human problem solving in fault diagnosis tasks. In W.B. Rouse (Ed.), *Advances in Man-Machine Systems Research: Vol 1*. Greenwich, CT: JAI Press, 195-222.

