NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

DEVELOPMENT OF A MAINTENANCE ADVISOR EXPERT SYSTEM FOR THE MK 92 MOD 2 FIRE CONTROL SYSTEM: FC-1 DESIGNATION - TIME, FC-1 TRACK - BEARING, ELEVATION AND RANGE, AND FC-2 TRACK - BEARING, ELEVATION AND RANGE

by

Clinton Dean Lewis

September, 1993

Thesis Advisor: Magdi Kamel

REPORT DOCUMENTATION PAGE

Form Approved OMB Np. 0704

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1993	Master's Thesis

4. TITLE AND SUBTITLE: Development of a Maintenance Advisor Expert System for the MK 92 MOD 2 Fire Control System: FC-1 Designation - Time, FC-1 Track - Bearing, Elevation and Range, and FC-2 Track - Bearing, Elevation and Range

5. FUNDING NUMBERS

6. AUTHOR(S) LCDR Clinton Dean Lewis, USN

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Naval Surface Warfare Center, Port Hueneme Division
4363 Missile Way
Port Hueneme CA 93043-4307

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

The MK 92 MOD 2 Fire Control System (FCS) is a fast reaction, lightweight, low manned, high performance multi-function fire control system that gives a ship all the combat functions required for independent tactical operation. The MK 92 MOD 2 FCS Maintenance Advisor Expert System (MAES) was proposed to assist Fire Control technicians in the troubleshooting of this complex fire control system. This thesis addresses the development of a fully functional prototype expert system that provides troubleshooting expertise for performance parameter deficiencies noted in the Daily System Operability Test (DSOT). Specific issues covered include: scope of the project, project background, the expert system development life cycle, hardware and software selection, knowledge acquisition, knowledge representation, knowledge implementation and lessons learned in the process.

14. SUBJECT TERMS Expert System, Prototype, Expert System Development Life Cycle, Knowledge Acquisition, Knowledge Representation, Knowledge Implementation, MK92 Fire Control System, System, Hardware and Software Selection

15. NUMBER OF PAGES 167

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

Approved for public release; distribution is unlimited.

Development of a Maintenance Advisor Expert System
for the MK 92 MOD 2 Fire Control System:
FC-1 Designation - Time, FC-1 Track - Bearing, Elevation and Range, and
FC-2 Track - Bearing, Elevation and Range

by

Clinton Dean Lewis
Lieutenant Commander, United States Navy
B.S., University of California, Santa Barbara, 1979

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
September 1993

ABSTRACT

The MK 92 MOD 2 Fire Control System (FCS) is a fast reaction, lightweight, low manned, high performance multi-function fire control system that gives a ship all the combat functions required for independent tactical operation. The MK 92 MOD 2 FCS Maintenance Advisor Expert System (MAES) was proposed to assist Fire Control technicians in the troubleshooting of this complex fire control system. This thesis addresses the development of a fully functional prototype expert system that provides troubleshooting expertise for performance parameter deficiencies noted in the Daily System Operability Test (DSOT). Specific issues covered include: scope of the project, project background, the expert system development life cycle, hardware and software selection, knowledge acquisition, knowledge representation, knowledge implementation and lessons learned in the process.

TABLE OF CONTENTS

I. INTRODUCTION

A. BACKGROUND

As the Department of Defense draws down, it is impacting all service organizational areas. Nowhere can it be felt more than in the area of technical assistance. Specifically, it is the loss of precious expert knowledge about systems that are gradually being phased out over a long period of time. As new systems are introduced, the limited number of system experts are shifted, thereby producing a gap in technical assistance for the old systems. Expert systems provide a possible answer to this ever widening gap.

In October 1991, Port Hueneme Division (PHD), Naval Surface Warfare Center (NSWC), Port Hueneme, CA, initiated development of a prototype expert system to assist technicians in troubleshooting the MK 92 MOD 2 Fire Control System (FCS). In September 1992, PHD approached the Naval Postgraduate School (NPS) to assist in completing the prototype expert system advisor.

B. OBJECTIVES

This thesis describes the design and implementation of a fully functional MK 92 MOD 2 Fire Control System

Maintenance Advisor Expert System. It addresses all development aspects in the life cycle of an expert system, emphasizing knowledge acquisition, knowledge representation, and knowledge implementation.

C. THE RESEARCH QUESTION

1. Are expert systems a viable option for diagnostic maintenance applications?

2. Can an off-the-shelf expert system shell be used to develop a functional prototype for the MK 92 Fire Control System?

3. What is the most appropriate knowledge representation paradigm to use when developing an expert system for the MK 92 Fire Control System?

D. SCOPE

This thesis develops a fully functional prototype maintenance advisor expert system to evaluate out of tolerance (NOGO) conditions which result from the FCS MK 92 MOD 2 Daily System Operability Test (DSOT) series. The system will evaluate performance NOGO's for the following performance parameters: FC-1 and FC-2 Designation Time and Bearing, FC-1 and FC-2 Track Bearing, Elevation, and Range, FC-4 and FC-5 Designation Time, FC-4 and FC-5 Track Range and Bearing.

The FCS MK 92 MOD 2 prototype was developed in conjunction with another thesis (Smith, 1993) that addresses the same project issues but different implementation

aspects. Both constitute the performance parameters of DSOT. A third thesis (Powell, 1993) provides a cost-benefit analysis of the expert system.

E. METHODOLOGY

Development of the Maintenance Advisor Expert System followed an approach proposed by Prerau (1990, pp.30-51) that consists of three major phases.

The Initial Phase involves gaining management approval, domain selection, and selection of hardware and software. Initial management approval was gained at PHD, NSWC in 1991. The Tartar systems department in turn gained management approval and funding support from the MK 92 project office (NAVSEA Code 62Z) in 1992. PHD solicited Naval Postgraduate School participation in September 1992. A project development team composed of faculty and graduate students was formed. PHD, NSWC selected the domain to include DSOT performance parameters. Hardware was readily available and the software chosen by the NPS team was Symbologic's Adept expert system shell.

The Core Development Phase is where the expertise and experience of a domain expert is entered into the system. The major aspects of this phase are knowledge acquisition, knowledge representation, and knowledge implementation. Knowledge acquisition and documentation were accomplished by

the domain experts at PHD. Knowledge representation was determined by the NPS development team, based in part on the capabilities of the chosen expert system shell. Knowledge implementation involved two stages. First, the rapid development of a feasibility prototype used to demonstrate the capabilities of the system and expert system tool. Second, the development of a fully functional prototype encompassing the performance parameters of DSOT.

The Final Development and Deployment Phase involves the building of a final production system. The scope of this thesis does not extend into the final phase.

F. THESIS ORGANIZATION

This thesis presents a unique format that combines theory and practice. Each Chapter gives the reader a theoretical background on the discussion material. A practical section based on the MK 92 MAES follows in italicized print. It indicates our experience in building the MK 92 MOD 2 prototype.

The thesis is organized as follows: Chapter II provides the reader with a general background and description of the MK 92 Fire Control System. In addition, it introduces and argues for expert system technology as a viable and cost effective approach for assisting shipboard technicians in troubleshooting and maintaining the MK 92 system.

Chapter III describes the expert system development life cycle. Specifically, it details the development life cycle as proposed by Prerau.

Chapter IV discusses knowledge acquisition in detail. It describes the strategy and techniques used in acquiring and documenting the domain knowledge elicited from the expert.

Chapter V addresses the issue of representing the acquired knowledge captured during knowledge acquisition. It discusses several knowledge representation paradigms and the knowledge representation selection process.

Chapter VI examines how the system is implemented. The topics covered include procedure builder issues, display builder issues and run-time issues. Additionally, procedure descriptions are provided with appropriate references to logic diagrams and node descriptions given in appendix A.

Chapter VII discusses lessons learned during system development. Emphasis is placed on insights gained about knowledge acquisition, representation, and implementation as well as the tool used to develop the prototype.

Appendix A contains logic diagrams and node descriptions, while Appendix B is an overview of Adept.

II. BACKGROUND

This chapter provides the reader with a general background and description of the MK 92 Fire Control System and MK 92 Daily System Operability Test (DSOT). In addition, this chapter will motivate the benefits of applying expert system technology by highlighting current difficulties and expense of using experts to troubleshoot diagnostic systems, particularly on board ships.

A. THE MK 92 FIRE CONTROL SYSTEM

The Fire Control System (FCS) MK 92 is a lightweight, low manned, high performance multi-function fire control system selected for use on Patrol Hydrofoil Missile (PH 1) ships, Coast Guard Medium and High Endurance Cutters Class (WHEC 715 - 726) in the MOD 1 configuration and on Guided Missile Frigate (FFG 7) Class ships in the MOD 2 and MOD 6 configurations. MOD 2 configurations are also on Australian Anzac and FFG 7 Class ships.

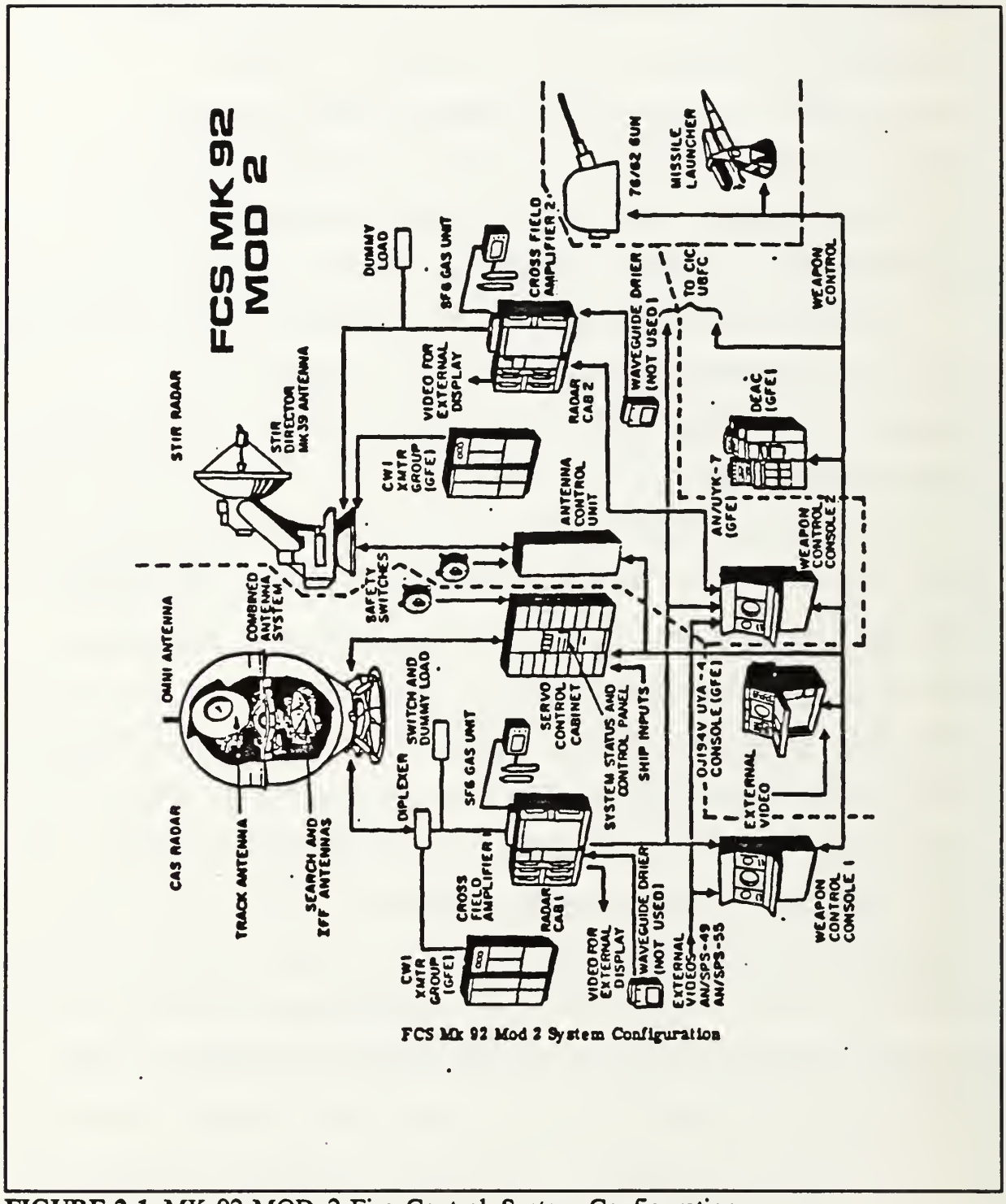
Functionally, the FCS MK 92 is a complex, fast reaction system that gives a ship all the combat functions required for independent tactical operation. It provides for integrated search surveillance, rapid detection and

engagement of fast air targets, as well as simultaneous engagement of surface and shore targets. In this configuration, a Separate Track and Illumination Radar (STIR), and its associated equipment, provide the controls for processing and firing of the Standard Missile (SM 1) from the Guided Missile Launch System (GMLS) MK 13 launcher.

The system has been modularized, as shown Figure 2.1, to promote multi-function capability and to support the system's basic maintenance concept of module replacement and planned maintenance.

The FCS MK 92 is maintained at the organizational and depot levels. It is in the organizational maintenance level that the DSOT resides. Tasks at this level are handled by the ship's Fire Controlman (FCn). They are limited to conducting preventative maintenance in accordance with the Planned Maintenance System (PMS), fault isolation, and corrective maintenance consisting of: replacing modules, assemblies, sub-assemblies, or components.

The FCn will be the primary user of the Mk 92 MAES. The entire thrust of the project was to provide an efficient and effective means for a FC to troubleshoot and correct a DSOT NoGo.



FCS Mk 92 Mod 2 System Configuration

FIGURE 2.1: MK 92 MOD 2 Fire Control System Configuration

B. THE DAILY SYSTEM OPERABILITY TEST (DSOT)

The DSOT is a complete and automatic end-to-end daily checkout of the system from the antennas to the weapons. It provides a rapid and comprehensive capability for quantitative availability assessment and operational training for the FCS Mk 92 MOD 2 system. Availability assessment encompasses the FCS and includes everything from the injection of simulated targets into the radar, to checking validity of system responses to the preprogrammed input target parameters. Additionally, summary response data is provided to the operator on an alphanumeric Total Evaluation Display (TOTE), and the Data Exchange Auxiliary Console (DEAC) provides a hard copy printout of the system's functional performance.

The DSOT has two modes: Local and Normal.

1. Local Mode

The local mode carries out test selection and DSOT is performed from both Weapons Control Console 1 (WCC 1) and Weapons Control Console 2 (WCC 2). See Figure 2.1.

Designation (DES) of Firing Channel (FC) FC-1, FC-2, FC-4, FC-5 and Gun and Launcher (LCHR) engagements are performed locally at the WCC 1 and WCC 2.

2. Normal Mode

This is the more realistic combat mode and should be exercised regularly. DSOT is completely controlled via the Weapons Control Officer (WCO) on a separate console in the Combat Information Center (CIC). The Weapons Support Processor (WSP) program communicates directly with the Weapons Control Processor (WCP) initiating the DSOT program which resides in the WCP. The WCO has the function to calibrate, designate, engage, evaluate and terminate the overall DSOT assessment.

3. The Maintenance Requirement Card

In accordance with the Maintenance Requirement Card (MRC) system, DSOT can be broken into four different tasks. The tasks include: Combined Antenna System (CAS) and STIR transmitter RF power checks, DSOT test initialization and calibration, Firing Channel performance tests, and CAS/STIR receiver sensitivity tests.

a. CAS/STIR Transmitter RF Power Checks

These checks are conducted prior to DSOT initialization to ensure that minimum power exists at the various system cabinets, drawers, and circuits. If minimum power does not exist, then DSOT should not be attempted due

to an inability of the CAS/STIR transmitter/receivers to radiate with sufficient RF power.

b. DSOT Test Initialization and Calibration

In the intialization test, the DSOT software first performs a loop test, from the DSOT controller through the DSOT box and serial link to the AN/UYK 7 computer, to verify digital communication. If the results are successful, the CAS on-line and STIR on-line indications are displayed; otherwise, a CAS off-line and/or STIR off-line indication is displayed.

After successful initialization, the computer program, in conjunction with the DSOT interface, performs an RF Power Calibration test.

The automatic calibration process is the method by which the DSOT equipment establishes the amount of RF power to inject into the front end of the Track/Search receivers in order to simulate real target parameters. The amount of injected RF power is, essentially, an attenuation setting that becomes a power reference for the channel being calibrated. This reference level is important for determining simulated target strength for long and short range targets. Additionally, maintenance tests use the reference level to determine FCS response values. The following four channels are calibrated: CAS/STIR Track

Target Channels; CAS/STIR Track Clutter Channels; CAS/STIR Track ECM Channels; Search Target and Clutter Channels.

The calibration process is performed on each channel in sequence. A summary calibration status (GO/NOGO) is printed out on the Data Exchange Auxiliary Console (DEAC). This printout, as shown in Figure 2.2, will occur any time the DSOT is selected and calibration begins.

If a NOGO should occur, the identified problem has to be dealt with immediately or DSOT will be of no use as an evaluation tool. As the example of Figure 2.2 shows, the target channel (fixed frequency) did not calibrate in the clutter mode. The target channel must first be calibrated to ensure proper results in the remainder of the DSOT test.

c. Performance Test

This test is similar to the calibration process. It is the first step, as shown in Figure 2.3, in MK 92 system evaluation. It consists of the following: injection of simulated targets into firing channels 1,2,4 and 5; providing quantitative assessment of the FCS in the form of a DSOT evaluation (GO/NOGO); producing a numeric error printout on the DEAC; and providing quantitative assessment of the FCS interface with GUN/LAUNCHER ENGAGEMENT.

```
DSOT
CAL* TR TGT FF-GO  7    ST TGT FF-GO 13   SR TGT FF-GO  5

CAL* TR CLT FF-PLO 8    ST CLT FF-GO 12   SR CLT FF-GO  5
CAL* TR ECM FF-GO  8    ST ECM FF-GO 10   SR ECM FF-PHI 8
CAL* TR TGT FA-GO  9    ST TGT FA-GO 13   SR TGT FA-GO  5
CAL* TR CLT FA-PLO 8    ST CLT FA-GO 12   SR CLT FA-GO  5
CAL* TR ECM FA-GO  10   ST ECM FA-GO  9   SR ECM FA-PHI 8

FF -Fixed Frequency      FA -Frequency Agility
TGT-Target               CLT-Clutter
CAL-Calibration          TR -CAS Track Channel
SR -Search Channel        # -Threshold Attenuation Value
ST -STIR Track Channel
Go -TGT,CLT,ECM Channels All Calibrated
PLO-Power Low  (Equates to a NOGO)
PHI-Power High (Equates to a NOGO)
```

FIGURE 2.2: Text Box

If at any time during the FC engagement with a performance test target, an FC or weapon evaluation threshold is exceeded, "NOGO" is displayed on the corresponding FC line of the TOTE. If the thresholds are within limitations, then "GO" is displayed on the TOTE.

After the FC is returned to standby, a summary evaluation is printed by the DEAC. The printout indicates the evaluations that were performed and provides detailed information on any NOGO condition that occurred during the test.

The sample evaluation, shown in Figure 2.4, provides DSOT test printout performance possibilities. The following definitions associated with items 9, 11, 14, and

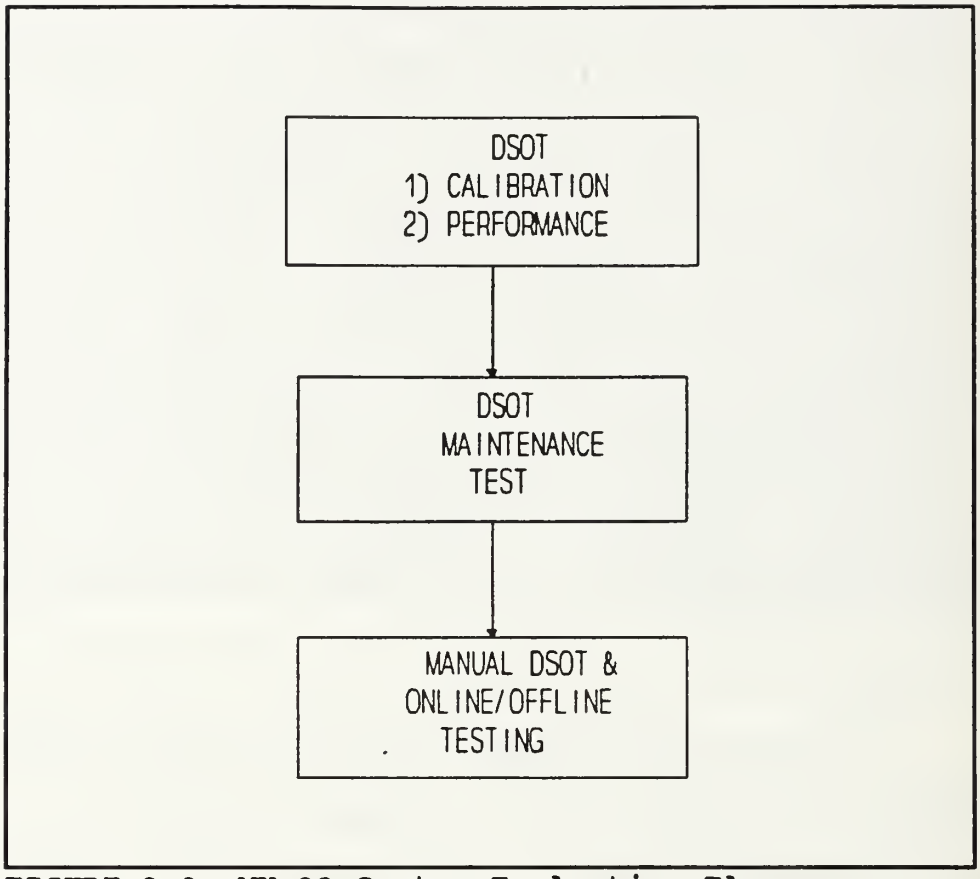


FIGURE 2.3: MK 92 System Evaluation Flow

and 16 are provided as examples:

-- 09. Designation Time FC-1/2. Monitor the time span between Air Target Designation (ATD) or Remote Air Target Designation (RATD) first set until designation match is achieved. Declare NOGO if this time exceeds 6 seconds.

-- 11. Acquisition time error FC-1/2. This is the time between designation match and achieved Air Target Acquisition (ATA). Declare NOGO if this time exceeds 6 seconds.

-- 14. Track bearing error FC-1. This parameter is sampled the same as the FC-1 tracking error. Threshold for the averaged bearing samples is .086 degrees. The threshold value for the instantaneous boresight sample is .800 degrees.

-- 16. Track elevation error FC-2. This parameter is sampled the same as the tracking range error. The threshold for instantaneous boresight sample is .800 degree values.

It should be noted that weapon evaluation printouts are generated only during the DSOT performance test scenario and not during training test scenarios. Additionally, if during DSOT tests, a live target is designated, the tests automatically terminate.

It is possible that while running, or using, the DSOT performance test, problems may be encountered (i.e., inability to acquire targets or degraded tracking with NOGO's). If this happens, the next step, as shown in Figure 2.3, is to conduct WCP controlled maintenance tests.

As shown in Figure 2.3, the last step in the MK 92 System Evaluation Flow is manual DSOT testing. This testing is used when DSOT and DSOT (WCP controlled) maintenance tests have been run and the results indicated NOGO's in FC-1 and FC-2.

d. CAS/STIR Receiver Sensitivity Test

This test is conducted after performance testing to determine a receiver's sensitivity level for distinguishing between signal and noise. It sets the receiver's Signal-to-Noise Ratio (SNR). The test includes the maintenance tests' mentioned in the MK 92 system evaluation flow.

DSOT PERFORMANCE EVALUATION

1. DSOT EVALUATION

2. START TIME: 00:03:22

3. PARAMETER	TIME OF FIRST NO-GO	TIME OF LAST NO-GO	ACTUAL ERROR	ALLOWED ERROR	
4. NO-GO	FC1 DATG	TGT 2 or FC1 DATLMC			
5. DES R (FC1)			+850	750	YDS
6. DES R (FC2)			-1650	1500	YDS
7. DES BY (FC1)			+1.5	1.2	DEG
8. DES BY (FC2)			+2.0	1.7	DEG
9. DES TIME(FC1/2)			8	6	SEC
10. DES TIME(FC4/5)			41	40	SEC
11. ACQ TIME			9	6	SEC
12. TRACK R	00:04:08	00:08:09	+ 26/45	25/40	YDS
13. TRACK B(FC1)	00:04:08	00:08:09	+.354/.976	.086/.800	DEG
14. TRACK B(FC2)	00:04:08	00:08:09	+.354/.376	.086/.290	DEG
15. TRACK E(FC1)	00:04:08	00:09:09	+.466/-852	.086/.800	DEG
16. TRACK E(FC2)	00:04:08	00:08:09	+.466/.852	.086/.290	DEG
17. GUN B	00:04:08	00:08:09	+ 1.625	.118	DEG
18. GUN E	00:04:08	00:08:09	+ 1.625	.118	DEG
19. LCHR B	00:04:08	00:08:09	+ .625	.50	DEG
20. LCHR E	00:04:08	00:08:09	+ .625	.50	DEG
21. MA6	00:04:08	00:08:09	2.5	2.0	DEG
22. MB6	00:04:08	00:08:09	2.5	2.0	DEG

FIGURE 2.4: DSOT Test Printout

C. THE BENEFITS OF EXPERT SYSTEM TECHNOLOGY

The numerous benefits of expert system technology become evident when there is a need to preserve expertise or to widen the distribution of or access to expertise at a reasonable cost. (Prerau, 1990, pp.3)

Consider the following scenario. The MK 92 FCn has just obtained a DSOT printout from the DEAC. There is a NOGO in FC-1 Acquistion Time and the FC must troubleshoot the system. The first step is to review the MRC deck and then pull out the appropriate manuals. One hour later, the FC has found the material needed to troubleshoot the system. However, after hours of troubleshooting the system and replacing several parts at random, no solution is found. The information provided in the technical manuals was inadequate to resolve the specific difficulty. An expert is then requested to be flown in at considerable expense to provide assistance.

The above scenario occurs quite frequently and is an example of the problems facing the FCn when troubleshooting the MK 92 Fire Control System. Many times the technical manuals only isolate the problem to several circuit cards as the potential fault. This "shotgun" approach, i.e., replacing parts at random for maintenance system troubleshooting, has been a common practice for years and

has had very limited success. It has resulted in an extraordinarily high "no fault evident" rate at the depot repair activity (i.e., the part removed in troubleshooting and turned in for repair, was a perfectly good part). The expense incurred by excessive parts usage and "expert" travel can be significantly reduced with the use of "expert systems".

Expert systems are especially suited to diagnosis and troubleshooting of complex systems, like the MK 92 Fire Control System. These systems share the following characteristics: expertise is scarce; obtaining expertise is expensive; providing expertise in remote locations is difficult; expertise is used in the absence of an expert; and expertise is provided in the form of a software program for efficient and effective maintenance troubleshooting.

Using an expert system to aid MK 92 technicians in diagnosing and troubleshooting their systems has the potential to be of great benefit to the U.S. Navy. Several of the specific benefits to the MK 92 community are (Powell, 1993, pp.38):

-- *Reduced Repair Parts Costs.*

-- *Reduced Mean Time To Repair (MTTR) of Casualties.*

-- *Reduced Reliance on External Technical Assistance.*

-- *Improved Shipboard Training and Knowledge of the MK 92 Fire Control System.*

The benefits are substantial. Two areas where this is particularly evident are unnecessary parts expenditures and technical assistance travel.

The results of Powell's (1993, pp.57) research show that during fiscal year 1991, unnecessary parts expenditures amounted to more than nine hundred thousand dollars. The estimated annual savings, with the use of the MK 92 MOD 2 MAES, amount to over one million dollars over the projected life of the system. The results also show that during fiscal year 1992, estimated travel expenditures amounted to over ninety three thousand dollars (Powell, 1993, pp.73). The potential savings to be realized by MAES deployment is approximately seventeen thousand dollars.

Expert system technology can provide the expert knowledge to improve performance and quality, reduce significant system downtime (i.e., improve system operational readiness) and promote better use of personnel and material resources. (Prerau, 1990, pp.4)

The following chapter will discuss the major phases that are required in the development of an expert system. Although expert systems will vary in their design, these phases are generic enough to give the reader an overview of the steps involved when developing an expert system.

III. EXPERT SYSTEM DEVELOPMENT LIFE CYCLE

This chapter defines and describes what an expert system is and discusses the development life cycle for a generic expert system. In the following chapters, the phases of the life cycle are described in detail.

Several definitions appeared in the literature to describe what an expert system is:

-- "... a program that applies the stored-up knowledge of a human expert to help someone solve a problem or perform a task." (Himes and Sperry, 1991, pp.6)

-- "... an advanced computer program that can, at a high level of competence, solve difficult problems requiring the use of expertise and experience." (Prerau, 1990, pp.3)

-- "... computerized advisory programs that attempt to imitate or substitute the reasoning processes and knowledge of experts in solving specific types of problems." (Turban, 1990, pp.455)

-- "... a type of analysis or problem-solving model, almost always implemented on a computer, that deals with a problem the way an 'expert' does. (Sprague and McNurlin, 1993, pp.455)

From the above definitions, the essence of an expert system lies in its ability to assist users in solving otherwise difficult problems through computer software with a "knowledge" base predicated on that of a human expert.

Several approaches have been proposed for developing an expert system. The approach proposed by Prerau will be

discussed in this chapter and used throughout the thesis for developing the prototype expert system.

Prerau (1990, pp.30-51) breaks the actual development of the expert system into three major phases. These phases, the initial, core and final phases, shown in Figure 3.1, are discussed below.

A. INITIAL PHASE

The initial phase includes: management approval, project team formation, domain selection, and the selection of hardware and software.

1. Management Approval

Management approval involves gaining the support from upper level management. Sometimes this is not a problem, in the case when help is elicited from the top down. Other times it can be extremely frustrating, in the case when the project has to be sold to upper management.

Management approval for the MAES was obtained in steps. First, PHD, NSWC, Tartar Systems department initiated the project and then gained NAVSEA management approval and funding support. NPS was solicited to participate by PHD, NSWC. Second, a feasibility prototype was demonstrated to PHD, NSWC upper management in order to validate that such a system was capable of being built.

2. Project Team Formation

Initial team members are identified and assigned to the project. Team member skill level, qualifications, and training are assessed in order to determine their ability to perform required project functions.

The MAES project team combined members from the Port Hueneme Division, Naval Surface Warfare Center, Tartar System Department, FCS Mk 92 System Division and Naval

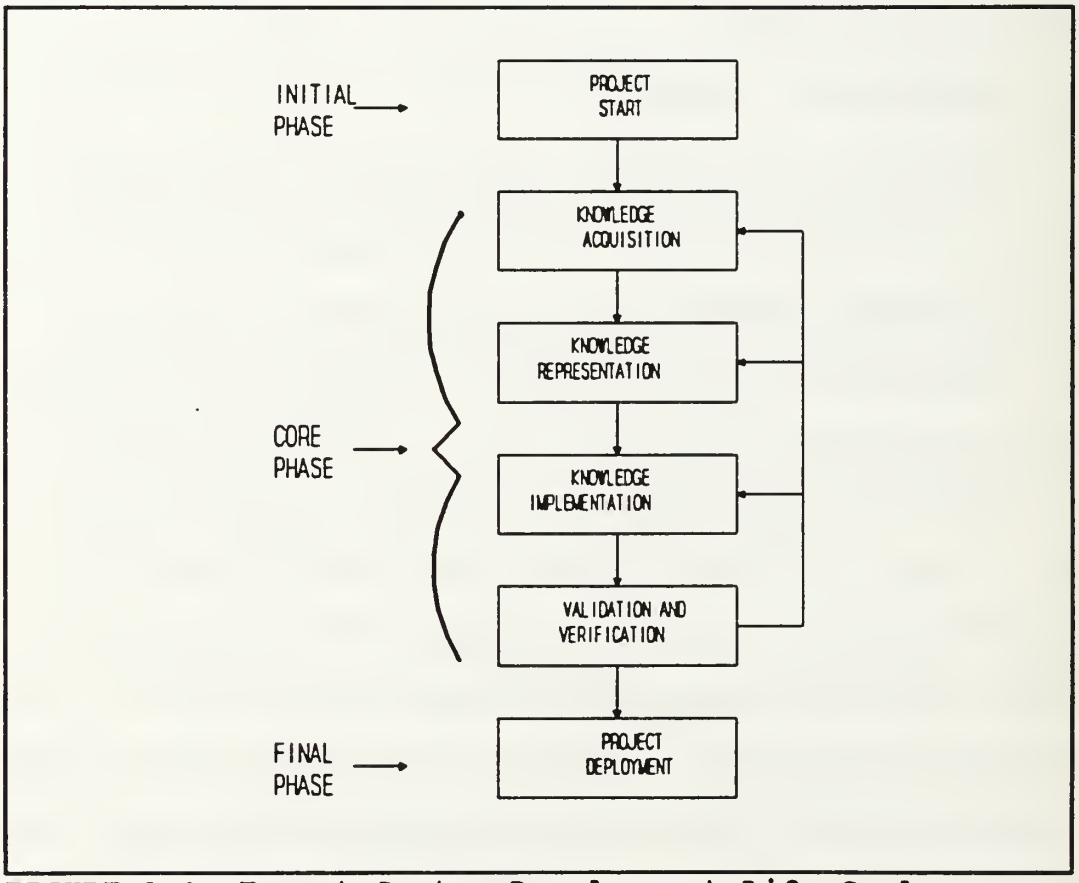


FIGURE 3.1: Expert System Development Life Cycle (Source: Prerau, 1990, pp.224)

Postgraduate School faculty and students. Team member skills, qualifications, and training were sufficient enough to perform all functions required to complete a functional MAES prototype.

3. Domain Selection

Domain selection involves the investigation of management suggested and management assigned applications. The goal is to find the one domain (application) that best suits the project. Appropriate domain selection will probably have more impact upon the eventual success or failure of the system than any other decision. (Prerau, 1990, pp.34)

The MAES domain was assigned by the sponsor; PHD, NSWC. Investigation by the NPS project team concluded that the chosen application area was excellent and could be achieved with available resources and within the projected time schedule.

4. Selection of Hardware and Software

Selection of a project's development environment (hardware and software) should be considered an integral part of the overall system development process. The goal of the selection is to obtain the optimal combination of hardware and software that best suits the project's unique requirements.

A "pitfall" recognized by Prerau (1990, pp.37) and germane to NPS in the MAES project was the availability of an already in place development environment. The initial attempt by PHD, NSWC at developing the MAES utilizing a large development expert system shell ran into difficulities and schedule slippage. The computers and software procured for the project were "suggested" to the NPS development team for developing the new prototype. The development hardware was adequate. However, after extensive evaluation of the available expert system development environment, the original software tool did not meet the NPS project requirements. The original development tool was not specifically designed for diagnostic systems, required additional software to build the user screens and necessitated a far greater learning curve for the NPS development team than time available. After an initial feasibility prototype was built and successfully demonstrated, using the Adept development tool, management was persuaded that changing development software was appropriate.

B. CORE DEVELOPMENT PHASE

The core development phase is concerned with taking the expertise and experience of a domain expert and entering it into the system. The two aspects of this phase include

feasibility and full prototype development. This phase has associated with it three major facets. These are: knowledge acquisition, knowledge representation and knowledge implementation. Each of these is discussed at length in Chapters IV, V, and VI. A brief overview follows.

1. Knowledge Acquisition

Knowledge acquisition is the process of extracting expert knowledge from both "public" (readily available knowledge) and "private" (knowledge privy soley to a domain expert) sources (Walters and Nielsen, 1988, pp.4). Due to the inherent difficulties of transferring knowledge from one person (the domain expert) to another (the knowledge engineer), knowledge acquisition is considered to be the "bottleneck" of expert system development (Buchanan and Shortliffe, 1984. pp.314).

2. Knowledge Representation

Knowledge representation is the process of determining the best representational form to fit the natural structure of the selected task. This can be done by using any or all of the artificial intelligence (AI) paradigms made available by a project's selected knowledge tool. (Prerau, 1990, pp.238)

3. Knowledge Implementation

Knowledge implementation is the process by which acquired knowledge, in its representational form, is implemented into a computer program. An important aspect of implementation is prototyping. The primary purpose of prototyping is to build successive versions of the final developed expert system. Each version adds additional knowledge and capability. In addition, the prototyping process helps determine if sufficient and appropriate knowledge exists and, if so, whether feedback and the results of the initial prototype are positive enough to continue with the project.

C. FINAL DEVELOPMENT AND DEPLOYMENT PHASE

This is the third major phase proposed by Prerau (1990, pp.30) and encompasses building and deploying a final production system. Once the production system is deployed, it will be in a maintenance phase. This is where "bugs" are found and fixed and new additions and enhancements are incorporated into the system's knowledge base.

Another important aspect of any software system development is testing and evaluation. Testing is an ongoing process that examines a system for compliance to the specification. It serves two purposes: to ensure that previously acquired knowledge is reflected appropriately in

the application, and to indicate areas where the application needs improvement. (Walters and Nielsen, 1988, pp.124) Evaluation can be separated into two components: validation and verification.

Validation is determining whether the "right" system is built. In other words, whether the system does what it is supposed to do and at a certain level of accuracy. Validating an expert system is the act of determining the correctness of a system and the level of the correctness.

Verification is determining whether the system is "built right". In other words, whether the system is implemented in the way it is designed. Verifying an expert system is the act of confirming that the program accurately reflects the documented expert knowledge.

The MAES project system development life cycle closely resembles Prerau's (1990, pp.224) system life cycle model.

The "core" of the MAES development process began with a feasibility prototype. The prototype's knowledge base was built on already acquired knowledge initially captured and represented by the domain expert in the form of diagnostic trees. The diagnostic tree's close fit with the selected tool's representational form greatly increased the speed of implementation.

The feasibility prototype successfully demonstrated the project's value and potential as well as confirmed the

ability of the Adept software tool for the project. With upper level management support, the functional prototype was developed on the foundation laid by the feasibility prototype. Its successive iterations are evaluated and verified by the domain experts at regular intervals.

IV. KNOWLEDGE ACQUISITION

Knowledge acquisition is the process by which expert system developers (frequently referred to as knowledge engineers) extract the knowledge (i.e., facts, rules, heuristics, procedures, etc.) that domain experts use to perform the task of interest. Knowledge engineers developing an expert system try to determine from "Public" knowledge sources (i.e., documents, textbooks, and journals) and "Private" knowledge sources (i.e., the experts) the manner in which experts solve the problem (Walters and Nielsen, 1988, pp.4). The result of knowledge acquisition is a specification of the knowledge of the expert system. This is the essential and fundamental part of an expert system. It is what distinguishes expert systems from conventional software programs and gives a system its power. Hence, after the selection of a domain, knowledge acquisition is very likely the most important task in an expert development. (Prerau, 1990, pp.200)

Knowledge acquisition is a relatively new field, and there is continuing research into better methods, including techniques to automate the process. However, for the foreseeable future, most of the knowledge for any large

expert system in a complex domain will be obtained through the interaction of knowledge engineers and domain experts.

Historically, acquiring knowledge from an expert has been found to be, at best, difficult. Reasons for this include: experts truly not knowing the mental process of what goes into the decisions they make; experts not fully understanding the extent of knowledge they use to solve even the simplest of problems; and experts not having a real grasp of how they do their jobs.

This chapter will deal with the conceptual knowledge acquisition issues for building an expert system. Included are the following:

-- Knowledge acquisition concerns when selecting an expert.

-- Knowledge acquisition and the iterative process.

-- Conducting knowledge acquisition sessions.

-- Knowledge recording and documentation practices.

This will be followed by a discussion of the practical knowledge acquisition issues that faced the FCS Mk 92 MAES project team.

A. KNOWLEDGE ACQUISITION AND THE EXPERT

The primary objective of the knowledge acquisition process is to elicit the "Private" knowledge an expert has as related to a chosen task. Experts should, therefore, possess the required level of expertise and certain

attributes conducive to a successful knowledge acquisition process. These attributes include: reputation, communicative skills, temperament, cooperativeness, and availability. Shortfalls in these attributes will make knowledge acquisition exacting and possibly doom the project.

Significant time and effort should be spent in the selection of the domain expert, as the success or failure of the system likely hinges on this very important aspect. (Prerau, 1990, pp.210)

B. ITERATIVE KNOWLEDGE ACQUISITION

Conventionally, initial knowledge acquisition begins by the domain expert methodically describing the task, or if the task is large, the first subtask that the system will focus on. Step by step, and in great detail, each task is covered by the expert and recorded by the knowledge engineer. This process, albeit difficult and time consuming, is repeated until the expert is satisfied that the knowledge recorded by the knowledge engineer is as complete and correct as possible. It is important that the knowledge engineer capture, concisely and succinctly, what is being considered. This includes what decisions are being made and why they are made. (Prerau, 1990, pp.212)

Several ways to acquire and modify knowledge are available to the knowledge engineer. Two of the recommended

ways include employing pre-implementive knowledge acquisition and using knowledge acquisition formalisms directly.

1. Pre-implementive Knowledge Acquisition

Early in a project, when significant amounts of knowledge have not been implemented, Prerau (1990, pp.222) suggests a technique for knowledge acquisition that follows a cycle of: elicit, document, and test, as shown in Figure 4.1.

a. Elicit

This consists of the actual elicitation of knowledge from the domain expert. Early stages entail collecting knowledge from written material or the expert's initial description of the task. In later project stages additional knowledge will be obtained and existing knowledge refined from test results.

b. Document

This step simply consists of documenting the elicited knowledge. Documentation should be standardized and clearly stated. It will serve as a starting point for rectifying inaccuracies and be an essential requirement for future software maintenance.

c. Test

The new knowledge is tested using the following techniques: first, have the expert analyze the new knowledge in the following traditional ways:

(1) Video analysis. This involves video taping an expert performing the domain task. Video tape analysis enables the domain expert to recall each step in the task with exacting detail.

(2) Tape analysis. The domain expert records each step of the task on audio tape allowing post-session analysis by both the expert and knowledge engineer.

Second, analyze the new knowledge using hand simulation. Hand simulation involves physically manipulating, if in rule-base form, which rule comes first, then second, third, etc. until the task is complete.

Third, compare the expert's analysis against those of the hand simulation, if the results are different, find the discrepancy. Then follow the reasoning of the hand simulation until it deviates from that of the expert's. Go back to step 1 and elicit new knowledge from the expert on how to modify the simulation discrepancy, thus bringing the new knowledge into agreement with the expert's analysis.

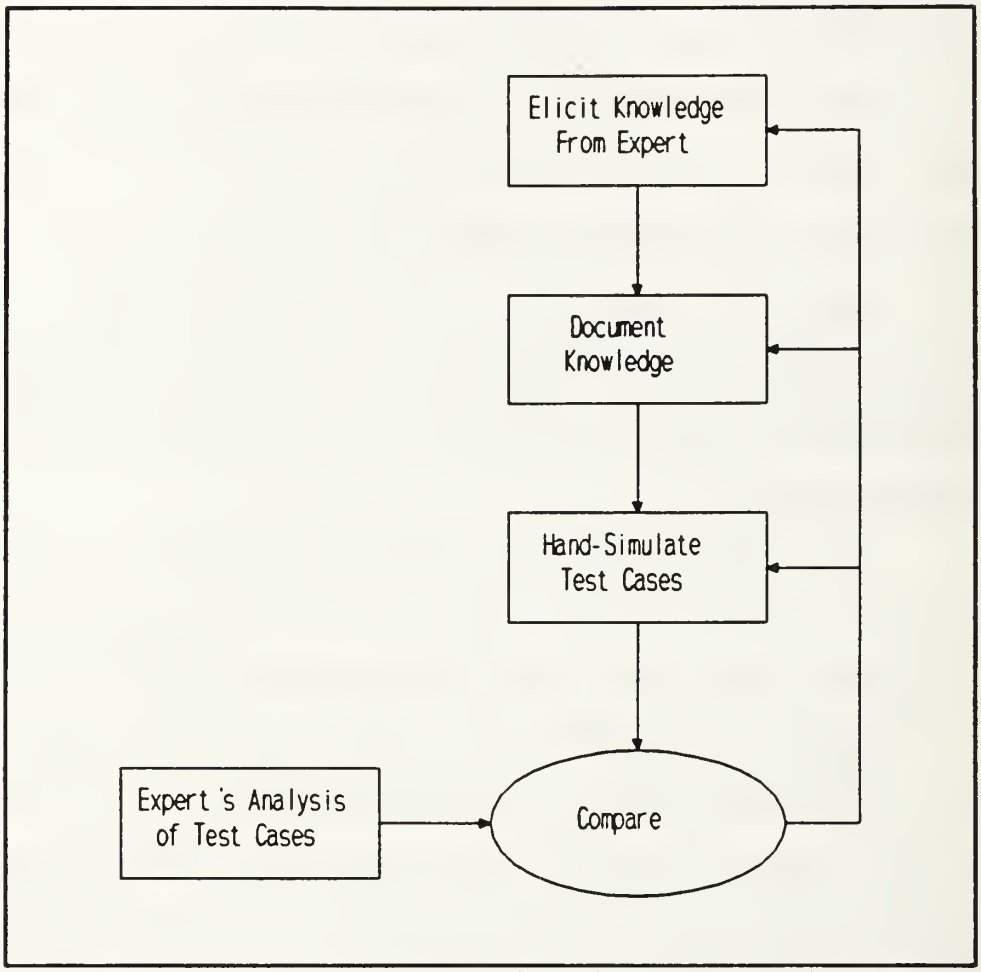


FIGURE 4.1: Knowledge Acquisition Cycle with Hand Simulation (Source: Prerau, 1990)

This iterative cycle of elicit, document and test will continue, revising and expanding the documented knowledge until a body of knowledge is found and implemented.

2. Using Knowledge Acquisition Formalisms Directly

A second method of knowledge acquisition has the experts define their reasoning directly in terms of specific

formalisms (i.e., "if-then" rules and procedures). This will add a measure of convenience to the documentation process since very little or no translation is required by the knowledge engineers. Also, the expert's use of the knowledge acquisition formalism will speed up the traditional "bottleneck" in expert system development, knowledge acquisition. Additionally, an understanding of the knowledge formalism will enable the expert to interpret the knowledge base and make suggestions as to where modifications are required.

This method was used by the PHD, NSWC engineers in developing the MK 92 MOD 2 FCS MAES. The expert's knowledge was represented in diagnostic tree diagrams.

C. KNOWLEDGE ACQUISITION SESSION ISSUES

An important part of setting up knowledge acquisition sessions is actually accomplished prior to the sessions. Knowledge engineers should review various publications, textbooks, manuals or other written materials to become as familiar with the task domain as possible. Once this literature review is finished, however, developers usually find that interviews must be held with the domain experts to capture the remaining knowledge. This is frequently referred to as heuristic (or rule of thumb) knowledge. (Walters and Nielsen, 1988, pp.35)

Setting up and scheduling knowledge acquisition sessions carry with them a number of important concerns which encompass:

-- A need to maximize access to the experts.
-- Minimization of interruptions.
-- Access to the "Prototype" program.
-- Session site and atmosphere considerations.

1. Maximizing Access to the Expert

The knowledge acquisition session should be organized so as to maximize access to the domain experts. How this is accomplished varies with each situation. The most effective way, in the author's opinion, is to gain the needed access through the chain of command's full commitment to the project. Only then is one assured that assigned experts will not have to piecemeal their time and knowledge. When the commitment is made, "blocks of the expert's time" should be scheduled for each session. No other assigned duties should be of a higher priority. (Prerau, 1990, pp.203)

2. Minimizing Interruptions

According to both Prerau (1990) and Walters and Nielsen (1988), interview sessions should not be interruptable. Optimally, sessions should be held away from the work places of both the experts and the knowledge

engineers. The setting should also be quiet, comfortable, and removed from the primary job area.

3. Accessing the "Prototype"

Knowledge acquisition sessions should include a point where the expert(s) can access the partially implemented system program (based on already acquired knowledge) for several reasons:

-- Implementation of acquired knowledge waits until after the knowledge acquisition session. However, it is also beneficial at times to record and implement acquired knowledge simultaneously so that new program run results can be compared with those of a domain expert. This technique allows immediate evaluation of the newly acquired information.

-- Session program runs allow experts to view, in program format, how they accomplish their primary tasks. Memory jogger program runs help the expert in expressing the "exact" nature of how they do their tasks, which in turn allows for updates to the program.

-- Numerous times during knowledge acquisition sessions program runs will elicit desired responses from the expert that were not necessarily anticipated before the sessions.

Therefore, access to the "prototype" program is very important and should be planned for during the knowledge acquisition process. (Prerau, 1990, pp.206)

4. Session Site and Atmosphere

Selection of the site for knowledge acquisition requires some consideration, especially if the project domain experts and the project knowledge engineers are not

collocated (which is usually the case). For example, picking a site away from the expert's primary job area reduces time demands and interruptions. If on the project's development site, it allows access to the expert system program. Also, travel expenses are reduced for either the expert or the knowledge engineers. Special development tools may also be more accessible. (Prerau, 1990, pp.207) Session atmosphere is another important consideration due to its direct relevance for a project's long-term success. An informal atmosphere will probably lead to more productive results. A spirit of mutual respect should be supported and encouraged. "Team building" is the key to unlocking success in any venture where more than one person is relied upon to complete a project.

D. KNOWLEDGE RECORDING/DOCUMENTING PRACTICES

Good techniques for recording the knowledge as it is acquired should support and speed knowledge acquisition sessions while ensuring accuracy for final documentation and representation. (Prerau, 1990, pp.230)

Flexibility should be built into the process. Thus when new domain knowledge is found, it can be easily written down and, if required, changed. The record should be clear and concise to facilitate the transfer of knowledge from temporary (i.e., journals and notepads) to permanent

documentation. Also, practices should provide a mechanism for recording reminders and benefits of the expert system. (Prerau, 1990, pp.231)

E. MK 92 KNOWLEDGE ACQUISITION ISSUES

The discussion will now focus on the practical issues that faced the FCS MK 92 MOD 2 MAES project team. These include:

-- *Selection of domain and domain experts.*

-- *Knowledge acquisition and the iterative process.*

1. Selection of Domain and Domain Experts

For the NPS development team this project was somewhat a departure from standard expert system development described earlier. The NPS project team was given not only the MAES project domain but also the assigned experts as well. Fortunately, the project domain was ideal for an expert system and each assigned expert possessed domain expertise and those attributes necessary to be good knowledge sources during our part of the knowledge acquisition process. The primary expert assigned to the MAES, Mr. Dorin Sauerbier of Paramax, proved to have lengthy experience with the MK 92 MOD 2 Fire Control System and all the related attributes for being an outstanding knowledge source. He also sought out the expertise of other Navy experts and added it to the knowledge base. In addition,

he recorded and documented the knowledge in the form of diagnostic trees. Figure 4.2 shows a rudimentary example of a diagnostic tree developed and documented by the expert and furnished to the NPS development team.

2. Iterative Knowledge Acquisition

It was found that having the partially completed prototype available on a computer during each knowledge acquisition session was invaluable. It allowed one or more of the project's domain experts to view the current implementation, modify the knowledge base, and make recommendations for improvement. This iterative cycle continued until the experts were satisfied with the completeness and accuracy of the knowledge base.

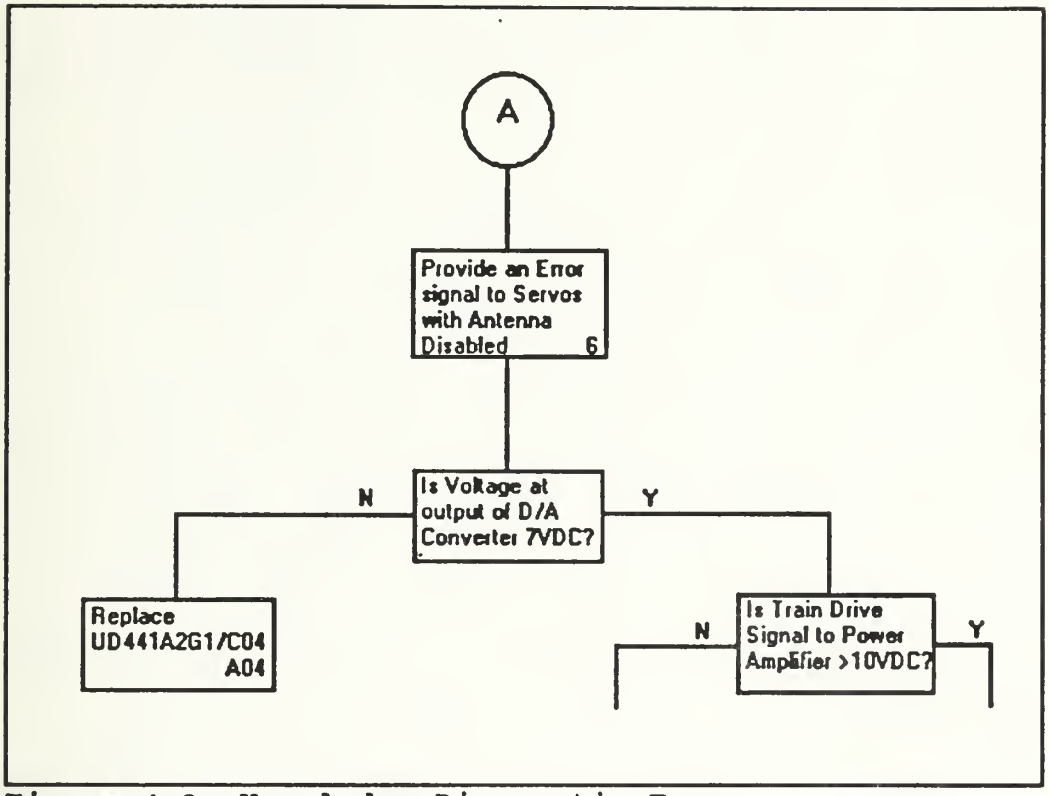


Figure 4.2: Knowledge Diagnostic Tree

V. KNOWLEDGE REPRESENTATION

As the expert system project acquires domain knowledge, expert system developers decide on the best way to represent that knowledge. They must decide how to express every pertinent item, concept, relationship, and structure in the domain and the expert reasoning behind it. Several Artificial Intelligence (AI) paradigms exist for representing the acquired knowledge. They include rules, frames, logic, scripts, semantic nets, and procedures. When selecting a knowledge representational model, it is important to note that the representation form used will also be the basis of the knowledge implementation. (Prerau, 1990, pp.238)

Knowledge representation is an important and sometimes difficult step in any expert system development effort. The representation form chosen should represent the domain's knowledge in a clear, concise, and efficient manner. It should thoroughly detail those domain areas that are important, relevant, and significant.

This chapter discusses issues of how acquired domain knowledge is represented. Specifically, it discusses

knowledge representation paradigms and knowledge representation selection.

A. KNOWLEDGE REPRESENTATION PARADIGMS

An expert system may contain one or more of several types of knowledge (i.e., active, static, declarative, and procedural), and each type ideally is represented by one or more available knowledge representation paradigms. The chosen paradigm should enable the knowledge representers to produce a clear, concise, and more effective knowledge representation. This section will discuss a few of the more common paradigms. These include: rules, semantic networks, frames, and procedures.

1. Rules

The most common way to represent knowledge in an expert system is through the use of rules. Rules, also called production rules, are elicited from the expert and as such draw upon experience, common sense, and the general ways of doing business. Rules are most appropriate when acquired knowledge can be generalized into specific if...then... statements.

Generalized if/then statements are presented logically in the form:

IF <premise> THEN <conclusion>

The text box, as shown in Figure 5.1, is an example of such a rule. Rule 1, derived for a diagnostic expert system, indicates that **IF** there are any transmitter or microwave components replaced, **AND** the summation video in the PAT mode is not in tolerance, **AND** the summation video level is in tolerance, when the AGC IF level is set to -1.1VDC, **THEN** part UD403/PanB - A/16 is recommended for replacement. Knowledge engineers use such statements, which are based on knowledge acquired from the experts, to form sets of rules.

Rule 1:	**IF**	**THERE ARE ANY TRANSMITTER OR MICROWAVE COMPONENTS REPLACED**
	AND	**THE SUMMATION VIDEO LEVEL IS NOT IN TOLERANCE (PAT MODE)**
	AND	**THE SUMMATION VIDEO LEVEL IS IN TOLERANCE WHEN THE AGC IF LEVEL IS SET TO -1.1VDC**
	THEN	**REPLACE UD403/PANB - A/16**

FIGURE 5.1: Text Box

The reasoning process begins with an inference engine. The inference engine provides system control. It is the part of the system that actually does the logical reasoning and planning (Keller, 1987, pp.17). A rule-based inference engine analyzes and processes if/then rules in one of two ways: backward-chaining or forward-chaining. In

backward-chaining, the inference engine works backward from the hypothesized conclusion to locate a known premise that would support the hypothesis. In forward-chaining, the inference engine works forward from a known premise to glean as many applicable conclusions as possible. (Walters and Nielsen, 1988, pp.196)

2. Semantic Networks

Networking is a natural and efficient way of representing knowledge. Networks consist of nodes and links. Nodes contain the represented knowledge (i.e., facts, concepts, and situations) and the links describe the relationship between connected nodes. One of the most common relationships in semantic nets is the "is a" link. This link type allows the facts of one node in the net to be inherited by another in the same hierarchy. For example, in the semantic network of Figure 5.2, one could infer that because mammals are vertebrates, and vertebrates have backbones, then mammals have backbones.

Reasoning within the semantic net is based on the "consequential association" of structures within the network. Essentially, this means that if you want to find out what an animal is, then you can construct a network segment, similar to Figure 5.2. This segment searches the knowledge base for consequential associations, by looking

for "is a" links connected to animal. If a link is located, then an answer is given: "animal is a vertebrate and a live thing."

3. Frames

Frames are knowledge containers with slots that contain information, values, rules and procedural code that can redirect queries until the correct answers or solutions are found. (Sprague and McNurlin, 1993, pp.457)

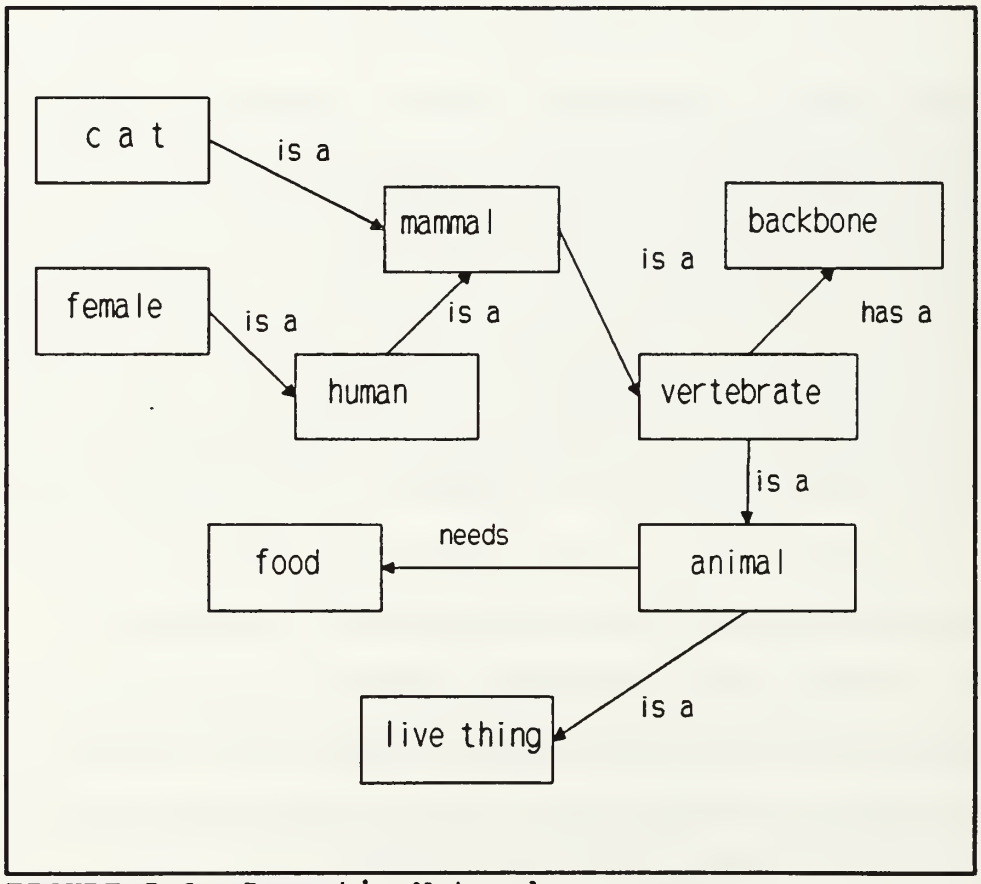


FIGURE 5.2: Semantic Network

Frames are arranged in a hierarchial structure, as shown in Figure 5.3, with the "root", representing the highest level of abstraction, at the top. The bottom frames, containing the actual and specific values, represent the "instance-of" that frame and are called "leaves."

The hierarchy permits inheritance of characteristics from related frames located at higher levels within the hierarchy.

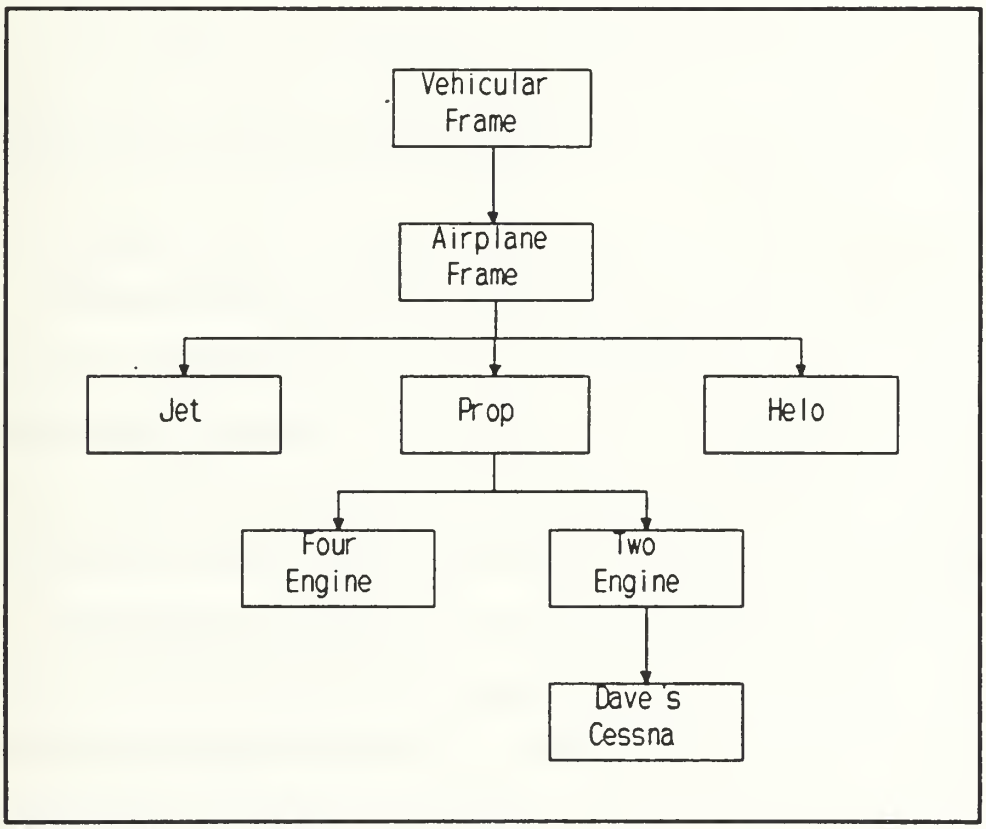


FIGURE 5.3: Frames Network

Reasoning with frames begins at the slot. The slot provides a mechanism for a kind of reasoning called "expectation-driven processing" (Turban, 1990, pp.507). This type of processing essentially starts with empty slots (i.e., questionable expectations) that become filled with data that under certain conditions, confirm those questionable expectations. So, frame-based reasoning is based on confirming expectations and, as such, is often just filling in slot values.

4. Procedures

The procedures paradigm, as shown in Figure 5.4, is a relatively new and simple way of representing knowledge and solving problems.

Prior to the advent of procedures, most systems were built on rules. Rules are fairly simple to understand because it is the way people tend to see problems; a relationship between cause and effect. However, most "real world" expert systems will contain hundreds or even thousands of such rules. Consequentially, most rule-based systems tend to breakdown after a few hundred rules due to the complexity inherent in large scale and rigid hierarchies. Procedure-based representations, by contrast, are simple and flexible networks.

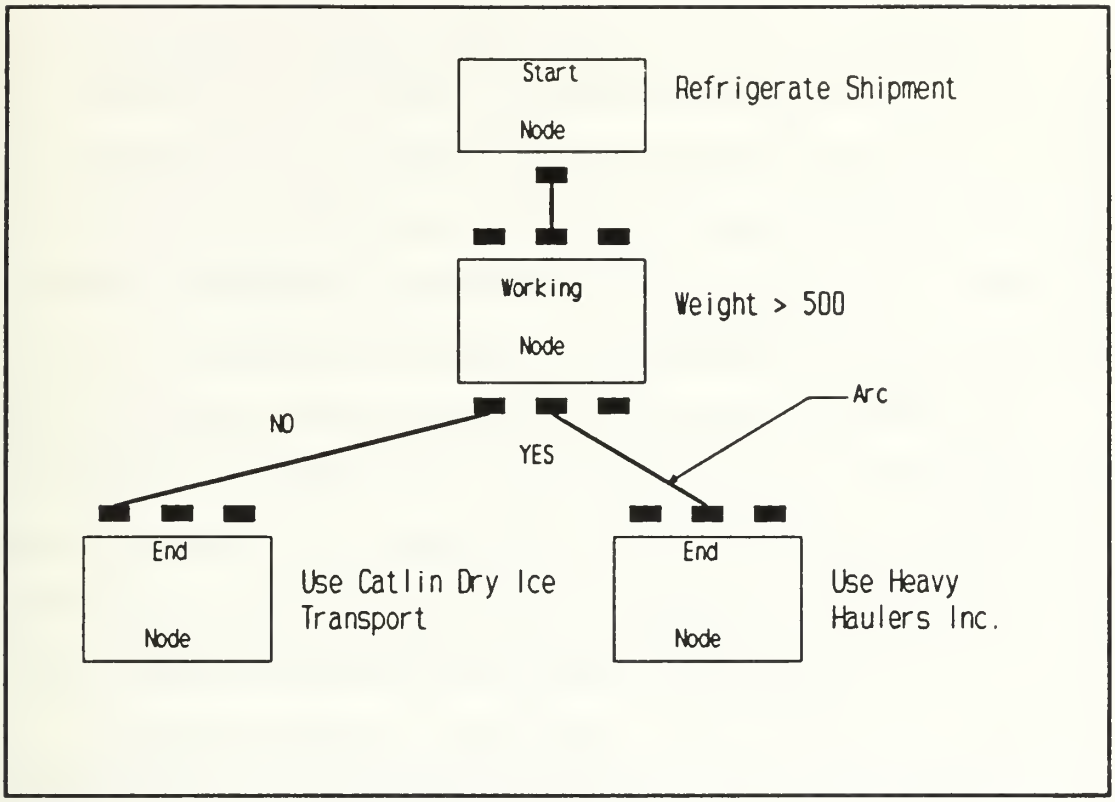


FIGURE 5.4: Procedures Network

The networks are graphical representations of procedures. Like a flow chart, the various steps of the procedure are represented by nodes, encapsulated series of instructions for reaching a goal, that are linked together by arcs which intuitively define the logic flow (Fersko-Weiss, PC Magazine, 1991, pp.58). The procedure network describes all the conditions that must exist before a concluding recommendation can be made (Sperry, PC Week, 1991, pp.51).

Inferencing within a procedures system can be accomplished through simultaneous forward and backward chaining (AI Expert, 1991, pp.60). The use of flexible inferencing allows for more dynamic reasoning, which translates into a system that is able to evaluate current and highly complex conditions and act accordingly.

The MAES is basically a collection of expert "troubleshooting" diagnostic trees being developed around the MK 92 MOD 2 DSOT program. The choice of a procedure based approach for representing knowledge was based on the fact that the knowledge furnished by the NSWC experts could be directly represented in a procedural representation. This knowledge base, when coupled with the inference engine of Adept, becomes the FCS MK 92 MOD 2 expert system.

B. KNOWLEDGE REPRESENTATION SELECTION

According to Walters and Nielsen, (1988, pp.321-330) choosing an appropriate representation for an application's knowledge base is still something of an art.

Currently no algorithm exists that produces the best decomposition and appropriate representations of the expert's knowledge form. However, a set of six guidelines has been established by Walters and Nielsen that may offer some assistance. These include:

-- Select the representation to fit the problem.

-- Decompose the problem.

-- Plan for the needed representations.

-- Work to the strengths of the representations.

-- Keep the problem structure visible.

-- Understand the system being used.

The six guidelines provide an example of the process that knowledge engineers may follow in the selection of a knowledge representation. Certain aspects of the MAES representation selection followed some of these guidelines and, as such, all six guidelines will be discussed.

1. Select the Representation to Fit the Problem

> The form(s) of representation chosen for the knowledge must match the inherent structure of the problem. (Walters and Nielsen, 1988, pp.321)

At first it seems simple, the knowledge engineer has only to compare the "natural" form of the knowledge and the employed inferencing procedures, then, find representations that match these forms (Walters and Nielsen, 1988, pp.321). Unfortunately, "Murphy's Law"[1] dictates that finding a good match of knowledge form to representation rarely happens.

A major constraint in choosing representations may involve project funding. Expert system development tools or "shells" have a wide range of prices, from the low hundreds of dollars to near one hundred thousand dollars. The more

---

[1] Aphorism; anything that can go wrong...will go wrong!

expensive tools likely require more expensive computers as well. Our experience would conclude that the closer the match between the knowledge and the tools ability to represent that knowledge the better. The productivity gains and maintenance savings over the systems life cycle will usually offset the initial expense of the development software.

Knowledge engineers, perhaps unknowingly, may also fall into the "pit" of trying to fit a square peg into a round hole. In other words, they will try to jam a particular body of knowledge into a representation that a previous system development employed or an expert system tool they were sold. There may be several reasons for trying to force a given set of knowledge into a tool's representation capabilities. Vendors often over sell their tool's ability to solve your problem. Many first time developers have somewhat naively been sold a very powerful development tool which is not suitable for their problem. Their mistake is that early on they have not focused on determining the best knowledge representation and then finding a tool that can best implement that representation. Other factors that need to be considered are what additional development software will be necessary to work with the expert system shell and what interfaces to other application software, such as databases, are included.

The fact that a previous system representation was good for the previous system does not mean that it will work for the current developing system. The damage that will occur from the above will outweigh, in terms of development complexity, difficulty, and maintainability, the initial costs of acquiring the right representation for the task at hand.

The MAES project has been one occasion where, in the viewpoint of Walters and Nielsen (1988, pp.321), matching of knowledge form to representation has been the rule instead of the exception.

The inherent structure of the MAES problem was one of diagnostic trees. Therefore, the natural representation for this type of structure would be a form that lent itself to the hierarchial aspects of trees (i.e., procedures). Significant effort was devoted to evaluating the development software. The primary selection criterion was a tool that could best represent the knowledge.

The author also encountered the "pit". NSWC had spent a considerable amount of time and money on a very good expert system shell. Their staff had received training and were familiar with the tool. An initial prototype had also been built, and they hoped NPS could use this work. Their desire was that the NPS team use this tool. However, in our assessment of the problem, we found the representation was

cumbersome, complicated, and did not fit the knowledge form chosen by the domain expert. The NPS faculty found a development tool which had a better match in representing the knowledge, had much less of a learning cure, cost significantly less and had some additional positive features. For instance, Adept had a built-in User interface (screen) builder and also did not charge for runtime application copies that would be distributed to the FFG-7 ships. Because of their significant investment, the initial management pressure was to use the representation of the previous system. NSWC management rightfully questioned the NPS recommended change in development environments. But, given the above facts, agreed to the change. Fortunately, the procedural representation selected has proven to be successful.

2. Decompose the Problem

> Complexity tends to increase exponentially with problem size, with a parallel increase in the development and maintenance resources required, as well as in the error count and debugging effort involved. (Walters and Nielsen, 1988, pp.323)

One of the major drawbacks in expert system development (or with any major software development) revolves around the fact that as the size of the problem increases so does its complexity. In order to decrease the complexity of the problem, problem decomposition is a must.

Knowledge engineers need to "decompose" the larger whole into its smaller parts whenever possible. As a result, the entire system not only becomes more meaningful, but more manageable and maintainable as well. It is only after decomposition that knowledge engineers can apply intelligent reasoning to the selection of a representation (i.e., rules, frames, or procedures) to fit the problem's knowledge structure.

The MAES naturally decomposed by virtue of being a diagnostic system. Diagnostic systems, in most cases, are engineered into distinct modules to facilitate system maintenance.

"Decomposition" of the MAES was not intended to facilitate maintenance, even though it seems to have worked out that way, but to aid the knowledge engineers in selecting a knowledge representation. The breakdown of the structure revealed that a procedure-base representation would be the simplest and most efficient way of building the system.

3. Plan for the Needed Representations

It is important to ensure that a representational form is chosen for a particular problem before a tool, with a default representational form, is selected due to the influence the tool will have on the system's design.

The MAES knowledge form was developed by the NSWC domain expert and already in place prior to the NPS team's involvement. A representation was chosen that "matched" that knowledge form. It is recommended to pick the representational form prior to selecting a system tool.

4. Work to the Strengths of the Representations

A representational form selected, the knowledge engineer should strive to organize a system in such a way as to maximize the selected form's strengths and minimize its weaknesses.

The representational form (procedures) selected for the MAES ideally suited the expert's knowledge form. The strength of procedures lies in its ability to model the real-life diagnostic procedures that experts use in documenting their knowledge and the excellent mapping between the two.

5. Keep the Problem Structure Visible

A major reason for using a particular representation for a certain type problem is to keep the structure of the problem visible to the system engineers. Once the representational form has been chosen, based on its "matching" of the knowledge structure, knowledge engineers

should ensure that the form's advantages are not lost by subsequent actions that would tend to hide the problem's structure.

This was a non-problem for the MAES project team. The inherent nature of procedures-based representation closely matched the expert's knowledge form from the outset. For this reason, the advantages afforded by procedures-based representation were never obscured and the problem's structure remained visible to the knowledge engineers throughout the implementation phase.

6. Understand the System Being Used

The last of the guidelines suggested for choosing a knowledge representation involves some advice, "Understand the (development) system being used" (Walters and Nielsen, 1988, pp.329). Developers do not always understand the systems they are using, in part due to the simple and friendly graphical interfaces that are available in today's commercial off the shelf (COTS) products. COTS products offer developers the need not to know, or understand, the theory that goes behind the product in order to use it. So any system that is engineered using it may lead to unwanted results. Consider the following example. A forward-chaining system (see PP.45) consists of the three rules shown in Figure 5.5.

Rule 1:	IF	I WILL TAKE THE BUS TO WORK
	THEN	I WILL NEED TO PUT ON BOOTS
Rule 2:	IF	THE STREETS WILL BE SLIPPERY
	THEN	I WILL TAKE THE BUS TO WORK
Rule 3:	IF	IT IS SNOWING
	THEN	THE STREETS WILL BE SLIPPERY

FIGURE 5.5: Rule-Set Execution Frequency Example (Source: Walters and Nielsen, 1988, pp.329)

If the premise IT IS SNOWING is asserted to be true, then the conclusion I WILL NEED TO PUT ON BOOTS can be derived. Any number of inferencing mechanisms would produce the same conclusion. The differences arise in how the different systems might arrive at that conclusion:

-- How many passes would have to be made through the rule-set to arrive at the conclusion?

-- Would listing the rules in a different order reduce the number of required premise evaluations?

To increase the system's efficiency, the developer needs to know the number of required premise evaluations to arrive at the desired conclusion and any steps that can be used to reduce that number. (Walters and Nielsen, 1988, pp.330)

The avoidance of unwanted results (i.e., rule-set execution frequency inefficiency) can be directly attributed to the developer knowing what COTS products are available and understanding how each product's capabilities will fit

into the system being used. Representation selection can then be made knowing there are COTS products available that will also match the represented form within the system.

This was a non-problem for the MAES project team. In part, this is due to the system being procedure-based as opposed to rule-based (less complex and more straight forward). Procedures are easily understood and relatively simple in their execution. The MAES is a procedurally represented system. Its developers understood that and were able to choose a COTS product whose capabilities closely matched that represented form. This understanding, in fact, led the system developers to gain significant insight into the domain problem and the knowledge tool being used to implement the knowledge base, discussed at length in Appendix B, that would eventually solve it.

VI. KNOWLEDGE IMPLEMENTATION

Knowledge implementation is the process by which acquired knowledge, in its represented form, is implemented into a computer program.

This chapter will discuss several key theoretical issues that effect knowledge implementation: expert system versus conventional programming implementation, knowledge implementation techniques, and implementation management.

Additionally, a discussion follows focusing on the practical issues of implementing the MK 92 FCS Maintenance Advisor Expert System.

A. EXPERT SYSTEM VS. CONVENTIONAL PROGRAMMING

The implementation of an expert system differs somewhat from the implementation of a conventional program. A conventional program is implemented using complete and full specifications. Specifications for an expert system can not be determined completely prior to implementation. Rather, specification and implementation evolve concurrently. Thus, a full top-down process can not be used. Instead, an expert system uses an iterative process for development. Segments (modules) of knowledge are programmed separately, refined,

and enlarged incrementally as the domain expert validates the implemented knowledge. (Prerau, 1990, pp.266-267)

B. KNOWLEDGE IMPLEMENTATION TECHNIQUES

In one aspect, however, expert system implementation is very similar to that of conventional program implementation. This is in the area of programmer experience. It is advisable for programmers to experiment with the development environment as soon as it is available in order to increase their proficiency. Additionally, general knowledge implementation techniques exist that have proven to be useful: knowledge acquisition rules/procedures and implementation rules/procedures, debugging, and documentation. (Prerau, 1990, pp.276)

1. Knowledge Acquisition Rules/Procedures and Implementation Rules/Procedures

It is clearly evident that there should be a close correspondence between knowledge rules/procedures and implementation rules/procedures. To make coding easier to follow, the method used during acquisition should match the representation used in the implementation. The close correspondence not only aids in development, but assists program maintenance as well. (Prerau, 1990, pp.277)

2. Debugging

As expected, debugging an expert system differs from debugging a conventional program. Each module of a conventional system has its own specifications and can be tested independently before it is incorporated into the main program. The same is not true of an expert system. The expert system must be incrementally debugged as it is being developed. (Prerau, 1990. pp.279)

Because knowledge acquisition continues throughout the development of the expert system, specifications are constantly evolving. Thus, it may be necessary to modify the program before coding is completed.

Programmers can usually debug a conventional program by running test case inputs and arriving at anticipated outputs. Expert system debugging presents a different problem. Not only must the program yield correct results in respect to the knowledge domain, but the domain must also be checked for inaccuracies by a knowledge expert. (Prerau, 1990, pp.279)

3. Documentation

Just as in conventional programs, expert system documentation is an important part of implementation. Because documentation is not a task most programmers enjoy, special attention should be paid to ensure that it is done

correctly. Expert systems require documentation of the knowledge domain as well as the program. Standard features such as inputs, outputs, and module purpose should be recorded. Matching the knowledge representation to the implementation by using rule/procedure correspondence, naming conventions, and specific references may make the documentation more complete and easier to follow. (Prerau, 1990, pp.280)

C. IMPLEMENTATION MANAGEMENT

Implementation management is important to any expert system development. Mechanisms to aid system developers in the performance of implementation are: uniformity of style and configuration management.

1. Uniformity of Style

In order to ensure programming style and display screens are uniform, pre-programming conventions should be agreed upon before any coding begins. These conventions should address logic flow techniques such as case handling, off-page connections, and location of controls and text on the display screens. Conventions enable several programmers to work on the project simultaneously. Once the conventions are agreed upon, they should be rigorously enforced to ensure compliance.

2. Configuration Management

Configuration management and control is an area where many existing development environments are weak. In most cases emphasis is placed on speed, flexibility, and ease of use. However, little effort is devoted to system management capability. It would be beneficial to have a mechanism that provides an ability to facilitate file maintenance. Also it would be useful to have a way of ensuring that project programmers have current and complete copies of the program. If utilities are not available in project software, then system implementers should develop their own methods of performing these functions.

D. VALIDATION AND VERIFICATION

Validation and verification are two important aspects of system evaluation. Validation examines whether the right system was built, or whether the system will operate at a given level of performance. Verification refers to examining whether the system was built right, that is whether the system matches the documented expert knowledge. (Prerau, 1990, pp.300)

Expert systems development, as described in the preceding Chapters, is an iterative process. Therefore validation and verification testing is completed during each phase of the system development.

Validation and verification of the FCS MK 92 MOD 2 Maintenance Advisor Expert System followed the above approach closely. As each procedure was implemented, it was sent to the domain expert for evaluation. This process ensured that the knowledge implementation form matched the expert's knowledge representation form in both logic flow and wording.

The use of an expert shell, such as Adept, greatly enhanced the verification process. Developers are able to concentrate on "matching" the expert's knowledge form, as opposed to concentrating on understanding and debugging the myriad lines of code associated with programming languages such as Lisp and Prolog.

E. MK 92 IMPLEMENTATION ISSUES

The practical issues discussed in this chapter will focus on the implementation aspects of the FCS MK 92 MOD 2 Maintenance Advisor Expert System. These include, procedure builder issues, display builder issues, and run-time issues.

1. Procedure Builder Issues

The project's selected knowledge tool uses a graphical tool set to construct individual procedures that define the skeleton, or framework, of an application. The procedures are also "linked", a process that enables the procedures to work together in solving problems.

Graphical representations and descriptions of the FCS MK 92 MOD 2 MAES procedures, FC-1 Designation Time and FC-1 and FC-2 Track Bearing, Track Elevation, and Track Range are presented in Appendix A. The procedures have been implemented as close to the expert's original knowledge form as possible. The reason for this decision is to promote future enhancements and simplify maintenance of the system's knowledge base.

2. Display Builder Issues

A display is a collection of graphical objects (i.e., buttons, text fields, and list boxes) that receive information from the user to complete a procedure or present results and instructions (Himes and Sperry, 1991, pp.14).

The project's knowledge tool provides a comprehensive toolbox that automatically constructs a default display each time the application's logic requires a user interface. The display builder enables the User to customize the default screen into unique and functional displays. The following display builder issues focus on: screen layout, colors, conventions, fonts, and graphics.

a. Screen Layout

The standard MAES screen is divided into three distinct sections: Main Title Bar, Procedure Box, and Action Box.

(1) Main Title Bar. The title bar, as shown in Figure 6.1, Section A, is located at the top of each display screen. It contains the procedure's title (usually the name of the DSOT firing channel with NOGO condition) and subtitle (usually the troubleshooting location). In the case of the main menu, only the procedure's title is displayed.

This section continuously advises the User which DSOT NOGO is being evaluated and the user's location within that NOGO's diagnostic tree.

(2) Procedure Box. The procedure box, as shown in Figure 6.1, section B, is located in the middle of the display screen. The content of the box varies with each screen, but generally, it contains: bitmap objects, procedure and help text, and occasionally labeled pushbuttons.

The procedure box is where the expert system requires the user to perform a task, or a series of tasks, and respond to queries. The input provided by the user enables the system to continue the diagnosis of the problem.

(3) Action Box. The action box, as shown in Figure 6.1, section C, is located at the bottom of the display screen.

This section contains pushbuttons that enable the user to interact with the expert system. The

number of buttons vary depending on procedure requirements. Generally, each action box has three buttons: yes, no, and help. Button properties also vary, but in most situations, "yes" equates to true, "no" to false, and "help" to user assistance and guidance on performance of tasks.

The action box is where the user interacts with the expert system by acting on information received from the procedure section.

b. Colors

The choice of display screen color is a rather difficult task. First, it is important that the chosen colors be complimentary, yet provide enough contrast to be distinctive to the eye. Second, the colors should be soft, but bright enough for the eye to distinguish individual characteristics. The project's selected tool, Adept, includes a color palette of several available colors. The palette enables the user to differentiate between border and fill colors. Also, shading of any selected color is possible through the tool's color editor. It is important for developers to keep in mind that pleasing all users is next to impossible, so they should choose a design and make it standard throughout the application.

The color scheme in this application is divided into background and foreground. A background layout is

maintained for all displays, while a foreground layout varies from one display to another.

(1) Background. Background colors were chosen to be appealing to the eye, yet not overpowering. Sufficient contrast was added to separate the different sections of the display, while still allowing a smooth transition from one section to another. The chosen colors are navy for the overall background, dark green for procedure and action box backgrounds, blue for procedure and action box title bars, aqua for procedure and action box title names, blue green for procedure and action boxes, and soft yellow for menu title bars.

(2) Foreground. As indicated, the foreground colors are procedure specific. For example, a procedure might have a note associated with one of its diagnostic steps. If so, the "notes" appear on the display screen in blue. The color blue provides sufficient contrast, to the blue green color of the procedure box, so it catches the User's eye. Warnings appear in red, bordered in white, while Cautions appear in yellow, also bordered in white. These are standard safety colors, which provide a stark contrast to the surrounding colors, and the user's eye will recognize them as such.

c. Conventions

Screen conventions are important to application standardization. Essentially, conventions are the rules that knowledge implementers must follow when building the individual modules and procedures that make up the expert system. The conventions discussed are naming, screen, and variable.

(1) Naming Conventions. These conventions standardize the labels that are applied to system procedures, pushbuttons, and title bars. An important aspect of naming conventions is the requirement that applied labels be sufficiently unique within separate procedures to prevent logic overlaps and errors during application. The naming convention for help pushbuttons covered two different situations. The first involved single help screens, with pushbuttons labeled "Return" (returns to DSOT). The second involved multiple help screens, with pushbuttons labeled "Return" (returns to DSOT), "Previous" (returns to the previous screen) or "Continue" (continues help), and possibly "Information" (provides explanatory data). A special situation involves help screens that specifically referred to additional help screens by letter. The special help pushbuttons are labeled "Help X" (X equates to the letter assigned).

(2) Screen Conventions. Screen conventions provide standardization on the location of items within each procedure display section. Essentially, the standard screen, as shown in Figure 6.1, becomes a template for the entire expert system development. Information varies, but its location remains generally the same. For example, the "Help" pushbutton usually resides in the Action Box. However, due to the number of sub-procedure pushbuttons in a menu procedure, in some instances the "Help" pushbutton may be re-located to the Procedure Box.

Procedure conclusion screens require a separate convention based on single or multiple recommendations. Single recommendations conclude with "Recommend Replacing", while multiple recommendations conclude with "Fault Not Isolated to a Single Card Failure. Recommended Replacement Order is:....".

Additionally, Adept can be run in either a VGA or SVGA display mode. Either format is useable, however, it is important that multiple-team development occur in the same display mode.

(3) Variable Conventions. Variables should be as descriptive as possible, while remaining within standard name and screen conventions.

d. Fonts

Wherever possible, the standard application text used was MS Sans Serif (font), Bold (font style), 12 (font size), and black (font color), as shown in Figure 6.1. Exceptions to the standard were the use of a 10 point font to fit large amounts of text into a procedure box, title bar heading, excluding "title only" menus, and the Procedure and Action box title bar, which also substituted aqua for black, as the font color.

Additionally, "Warning and Caution" display screens use a 24 point font in the title, and a 14 point font in the text body.

e. Graphics

The graphic interface of "Windows" was instrumental in the development of this project's display screens. Its point-and-click approach is similar to drawing programs, as such, "Windows" enabled the developers to customize display screens to be more efficient, with the information available and more effective, by ensuring the information was presented in a functional manner.

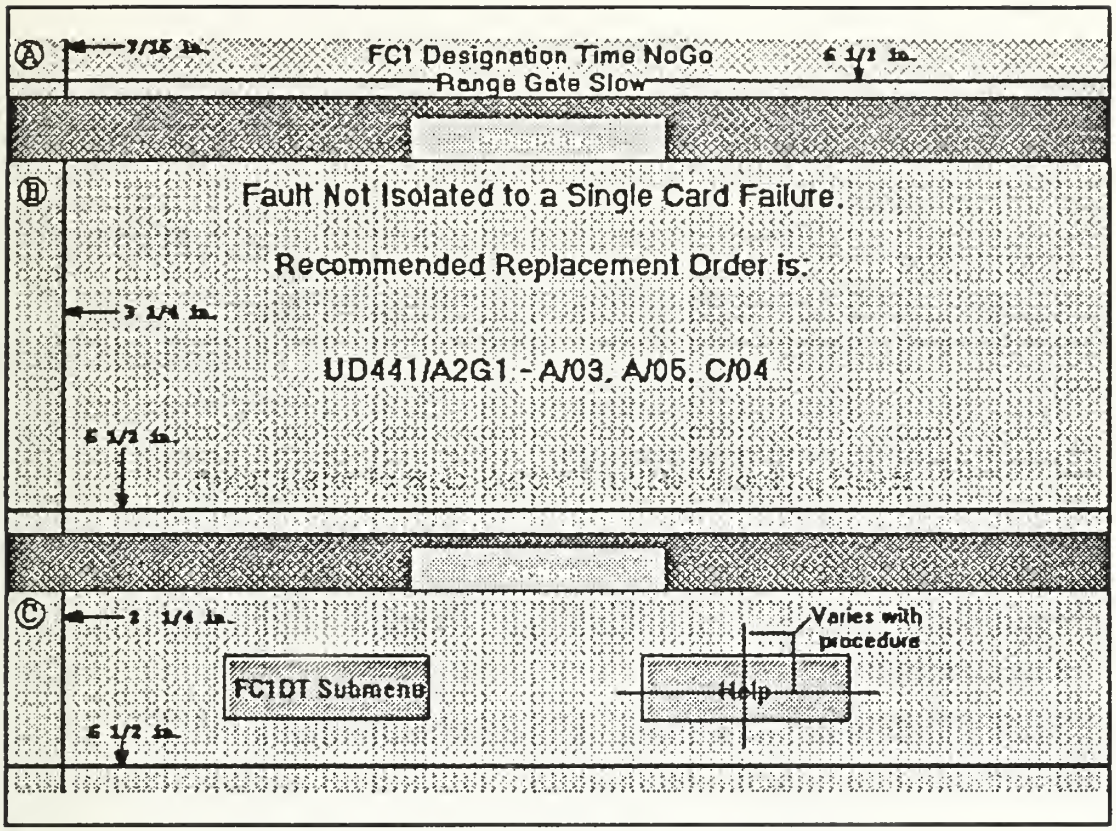


FIGURE 6.1: FCS MK 92 Mod 2 Display Screen

3. Run-Time Issues

The importance of runtime to developers is apparent as a test driver for procedures. The developer can work through a procedure, step-by-step, just as the end-user would, to determine if the procedure functions as it was designed and implemented to function.

Visual debugging, is also an important feature. It allows the developer to navigate through a procedure and easily spot any problems in its logic.

VII. LESSONS LEARNED

This chapter presents some insights gained through the experience of developing the MK 92 MOD 2 Fire Control System Maintenance Advisor Expert System prototype.

A. KNOWLEDGE ACQUISITION

As discussed in Chapter IV, the MAES project knowledge acquisition process was unique in that knowledge acquisition was accomplished entirely by the domain expert. This aspect of the MAES was unequivocally the major reason for the success of the project. The domain expert, without formal training as a knowledge engineer, ably elicited and documented the "expert" knowledge that has been implemented into the FCS MK 92 MOD 2 knowledge base.

The process of knowledge acquisition is by far the most time consuming aspect of developing an expert system. The fact that this part of the project had been accomplished by the expert in a form that closely matched a knowledge representation paradigm substantially reduced the overall time for developing the functional prototype.

The expert system was not hindered by the knowledge acquirer's lack of experience in knowledge acquisition techniques. Although a significant portion had been

completed prior to the NPS team's involvement, it was apparent that the knowledge was elicited accurately and in great detail.

One area where experience would suggest improvement is in the documentation of the acquired knowledge. A more logical way of structuring and representing the knowledge on paper is desirable. For example, a specific procedure was frequently represented on multiple sheets of paper in a somewhat haphazard manner. A better way would have been to break the knowledge down into better structured modules. The key idea is that, with a certain amount of forethought toward the eventual representation and implementation of the knowledge, a more direct method in documenting the knowledge could have been built into the knowledge acquisition process.

B. KNOWLEDGE REPRESENTATION

Knowledge representation, like knowledge acquisition, was also a major issue for this project. The domain's natural tendency to fit into a procedural representation was extremely fortunate for the MAES project team. The "art" of finding a representation to fit a specific knowledge structure can be, at times, quite difficult. It is important to spend enough time searching for the

representation that closely fits the domain knowledge structure. A close relationship between the acquired knowledge and its representation greatly reduces the implementation time.

The importance of a representational fit is immediately apparent when selecting the system's knowledge implementation tool. As discussed in Chapter III, most tools use a default paradigm. The closer a represented knowledge form matches that paradigm the faster and easier the system will be implemented. The MAES representational form closely fit the selected tool's implementation paradigm. This "match" enabled the developers to build a functional prototype in months instead of years.

C. KNOWLEDGE IMPLEMENTATION

This is the area of system development that is the most familiar to the author. Many of the situations encountered during this phase of the life cycle have been alluded to in the literature but were not fully appreciated until experienced. In the following sections, we will discuss these implementation issues: standards, project expert tool, and project support.

1. Standards

In a multiple programmer environment, standardization is of major concern. Before implementation

began, standards were established. This included standards for screen layout, color scheme, object positioning, and text composition. Periodic standardization meetings were held to ensure that established standards were being followed. Standardization can not be overemphasized. Without standardization, multiple programmer environments become next to impossible to coordinate.

2. Project Tool

Adept by Symbologic Corporation was the knowledge tool selected for the MAES project. Adept was chosen for several reasons: visual programming capabilities, quick learning cycle, tool modularity, procedure paradigm, procedure/display building, and graphic interface.

a. Visual Programming Capabilities

Adept combines visual development with a procedures-based paradigm. Visual application development means that programmers can build applications by creating and manipulating graphical objects on the screen. Adept's graphical approach facilitates critical thinking and makes it easy to spot gaps in procedures. (Himes and Sperry, 1991, pp.8-11)

b. Quick Learning Cycle

Adept was easy to learn. A getting started tutorial was simple to follow and provided the necessary steps to gain a quick working knowledge of the program. The reference manual was also well laid out, providing indepth information on Adept's capabilities. Technical phone support was available for problems that could not be solved using documentation or on-line help.

c. Tool Modularity

Adept is especially suited to a multiple procedure environment. System modules, consisting of grouped procedures can be developed and tested as "small" systems within a "larger" system. Project programmers found this to be invaluable as the system expanded in size.

d. Procedure Paradigm

Adept's procedural paradigm matched the domain's knowledge representation very closely. For example, the multi-path divergence of the expert's diagnostic tree diagrams were easily transformed into Adept's node objects. Additionally, the tree "yes" and "no" branches matched Adept's node arcs. The time spent in choosing a suitable tool effectively reduced the time required to implement the system's acquired knowledge.

e. Procedure and Display Building

Adept combines procedure and display building into a single tool. This allowed the programmers to build procedural nodes and their associated displays without having to use a separate software program. This saved valuable time and made it convenient for verifying the knowledge content of each node's display. Also, Adept's node view allows a programmer to view each node and its display sequentially for debugging purposes.

f. Graphical Interface and Other Features

Adept's graphical interface proved to be a flexible and valuable part of the tool. System programmers were able to import bit mapped image files into displays, thereby enhancing the display screen's overall presentation. Also, text insertion and editing is a simple process. Various size text boxes can be created in which font, font style, font size, and font color are created and manipulated to fit a programmer's desires.

An important Adept graphics feature involves the separation of the foreground and background. This enables the programmer to create a consistent background for all displays. On the other hand, the foreground can be changed from one display to another.

The snap-to-grid graphics function of Adept proved invaluable to the MAES programmers when manipulating objects around the display screen. It allowed standard coordinates to be established and ensured that objects were "snapped" into those coordinates.

One final graphics area involves the cut and paste function of Adept. This feature saved programmers from duplicate implementation of procedures which were very similar. Programmers were able to copy, paste, and modify information from one procedure to the next.

D. PROJECT SUPPORT

The FCS MK 92 MOD 2 MAES, like any expert system project, needs support from many areas. Two of these areas are: upper level management and the system project team.

1. Upper Level Management

Upper level management support, as mentioned in Chapter III, is crucial for project initiation. The MAES, as it stands now, has undergone a complete cycle of support-loss of support-support. Various levels of management support are required and must be maintained throughout a project's life. When NPS became part of the project in September 1992, management support at both NSWC and NAVSEA was waning. Building and demonstrating a feasibility prototype to NSWC management greatly restored their

confidence in the project. In March, 1993 the NAVSEA representation, without ever seeing the program or prototype, terminated their support for this project along with several other projects.

The PHD, NSWC and NPS development team strongly believed that this project could be successfully completed and offered significant cost savings and improved system operational readiness to the Navy. In July, 1993 NAVSEA was given a demonstration of the prototype system and briefed on the preliminary findings of a cost-benefit study being done as a NPS student's thesis. After review, they agreed to reinstate support for the project and provide funding for fiscal year 1994.

In todays downsizing military, management demands positive results prior to extending scarce resources. The MAES was demonstrated as a feasibility prototype and proved itself to be a viable system. It is important to remember that support and funding are synonymous when it comes down to a system's continuance or termination.

2. Project Team

Large scale projects, especially where project team members are not colocated, must maintain a positive interaction and support base. This is vital and should be recognized as an important aspect of a successful expert

system development. Fortunately, the MAES project team has enjoyed this kind of interaction and support since project start up.

APPENDIX A PROTOTYPE DESCRIPTIONS

MAIN MENU

Name: Main Menu (FIGURE 1)

Number: 0

Description: Serves as the first menu in program, allows selection of Performance or Calibration portions of the diagnostic program

Called by: Starting the Program (the first screen the operator sees is a FFG 7 class ship with system developer information and a "CONTINUE" button to start the program.)

Calls: Performance and calibration menus

==========

Name: Performance Menu

Number: 1.0

Description: Allows the selection of FC1, FC2, or FC4and5

Called by: Main Menu

Calls: FC1, FC2, or FC4and5

==========

Name: Calibration Menu

Number: 2.0

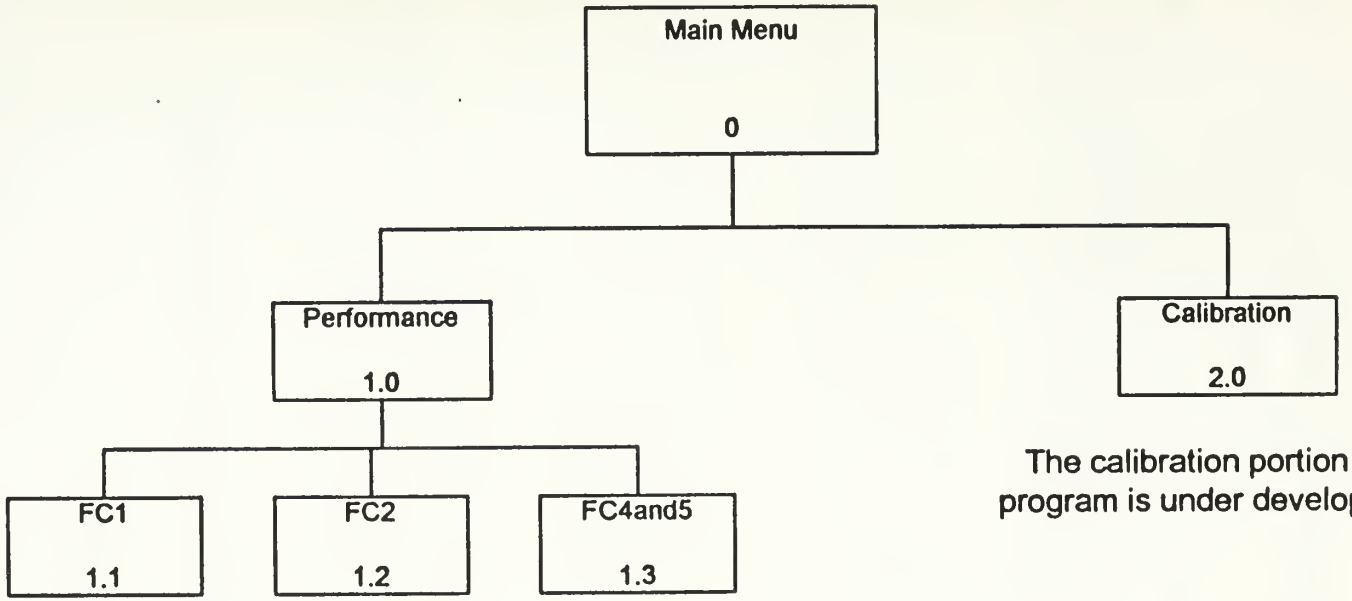
Description: Allows selection of Calibration procedures.

Called by: Main Menu

Calls: (The calibration procedures are under development.)

==========

Main Menu



The calibration portion of the program is under development.

FIGURE 1

FC1 MENU

Name: FC1 Menu (FIGURE 2)

Number: 1.1

Description: Allows selection of FC1 Designation; Time, Range, and Bearing. FC1 ACQ. FC1 Track; Bearing, Elevation, and Range

Called by: FC1 DTRB Menu

Calls: FC1DTRB, FC1ACQ, FC1TBER

==========

Name: FC1 DTRB Menu

Number: 1.1.1

Description: Allows selection of FC1 Designation; Time, Range, and Bearing procedures

Called by: FC1 Menu

Calls: FC1 DT, FC1 TR, and FC1 TB

==========

Name: FC1 ACQ

Number: 1.1.2

Description: Allows selection of FC1 ACQ procedure

Called by: FC1 Menu

Calls: See FC1 ACQ Menu

==========

Name: FC1 TBER Menu

Number: 1.1.3

Description: Allows selection of FC1 Track; Bearing Elevation, and Range procedures

Called by: FC1 Menu

Calls: FC1 TB, FC1 TE, and FC1 TR

==========

FC1 Menu

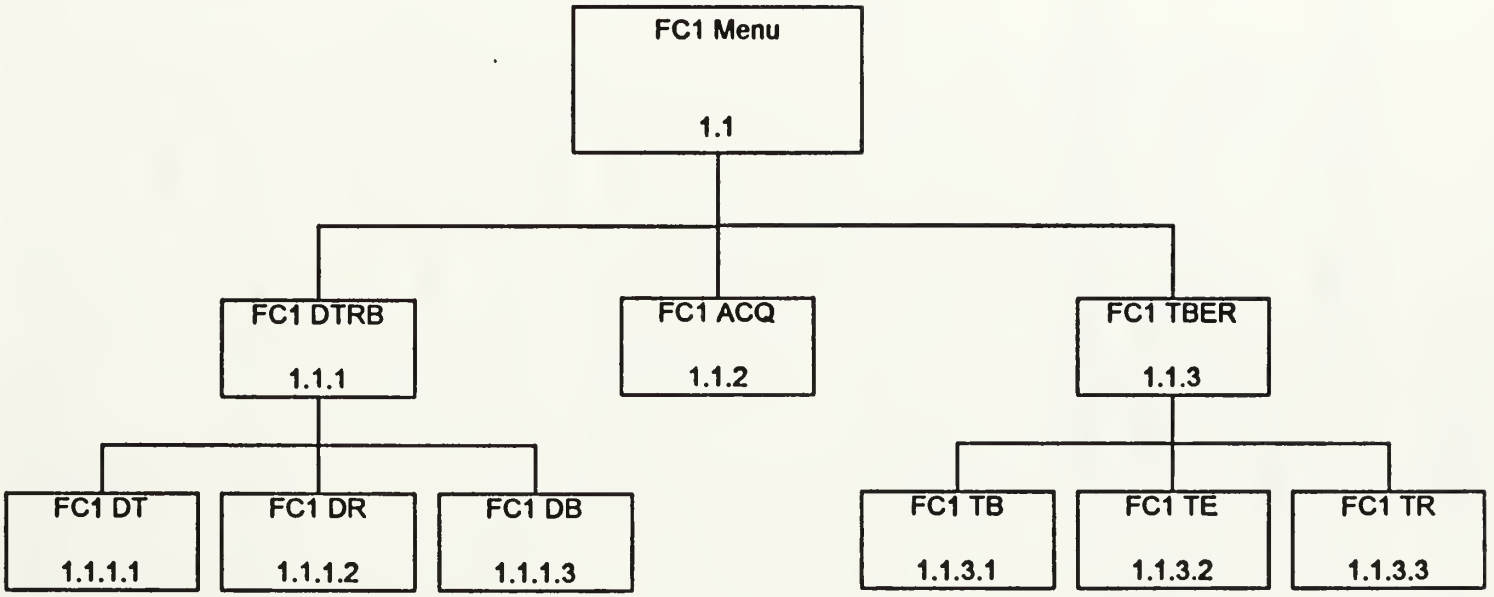


FIGURE 2

FC1 DT

Name: FC1 DT (Figure 3)

Number: 1.1.1.1

Description: Allows selection of one of three FC1 DT cases:
Case 1 -- Range Gate approximately 25K yards. Range Reading on TOTE equals zero or is less than 24K yards or greater than 26K yards.
Case 2 -- Range Gate approximately 25K yards. Range Reading on TOTE approximately 25K yards.
Case 3 -- Range Gate not present or no where near 25K yards.

Called by: FC1 Menu.

Calls: FC1 DT Case 1, FC1 DT Case 2, and FC1 DT Case 3.

==========

Name: FC1 DT Case 1

Number: 1.1.1.1.1

Description: Allows trouble shooting of FC1 DT Case 1 procedure.

Called by: FC1 DT Menu.

Calls: FC1 DT Case 1A.

==========

Name: FC1 DT Case 1A

Number: 1.1.1.1.1.1

Description: Continues trouble shooting of FC1 DT Case 1 procedure.

Called by: FC1 DT Case 1.

Calls: None.

==========

Name: FC1 DT Case 2

Number: 1.1.1.1.2

Description: Allows trouble shooting of FC1 DT Case 2 procedure.

Called by: FC1 DT Menu.

Calls: FC1 DT; No Track Antenna Movement, Track Antenna Slow, No Range Gate Movement, Range Gate Slow, Both No Movement, and Both Slow.

==========

Name: FC1 DT No Track Antenna Movement

Number: 1.1.1.1.2.1

Description: Allows trouble shooting of FC1 DT No Track Antenna procedure.

Called by: FC1 DT Case 2.

Calls: FC1 DT No Track Antenna Movement A.

==========

Name: FC1 DT No Track Antenna Movement A

Number: 1.1.1.2.1.1

Description: Continues trouble shooting of FC1 DT No Track Antenna procedure.

Called by: FC1 DT No Track Antenna Movement.

Calls: None.

==========

Name: FC1 DT Track Antenna Slow

Number: 1.1.1.1.2.2

Description: Allows trouble shooting of FC1 DT Track Antenna Slow procedure.

Called by: FC1 DT Case 2.

Calls: None.

==========

Name: FC1 DT No Range Gate Movement

Number: 1.1.1.1.2.3

Description: Allows trouble shooting of FC1 DT No Range Gate Movement procedure.

Called by: FC1 DT Case 2.

Calls: None.

==========

Name: FC1 DT Range Gate Slow

Number: 1.1.1.2.4

Description: Allows trouble shooting of FC1 DT Range Gate Slow procedure.

Called by: FC1 DT Case 2.

Calls: None.

==========

Name: FC1 DT Both No Movement

Number: 1.1.1.1.2.5

Description: Allows trouble shooting of FC1 DT Both No Movement procedure.

Called by: FC1 DT Case 2.

Calls: None.

==========

Name: FC1 DT Both Slow

Number: 1.1.1.1.2.6

Description: Allows trouble shooting of FC1 DT Both Slow.

Called by: FC1 DT Case 2.

Calls: None.

==========

Name: FC1 DT Case 3

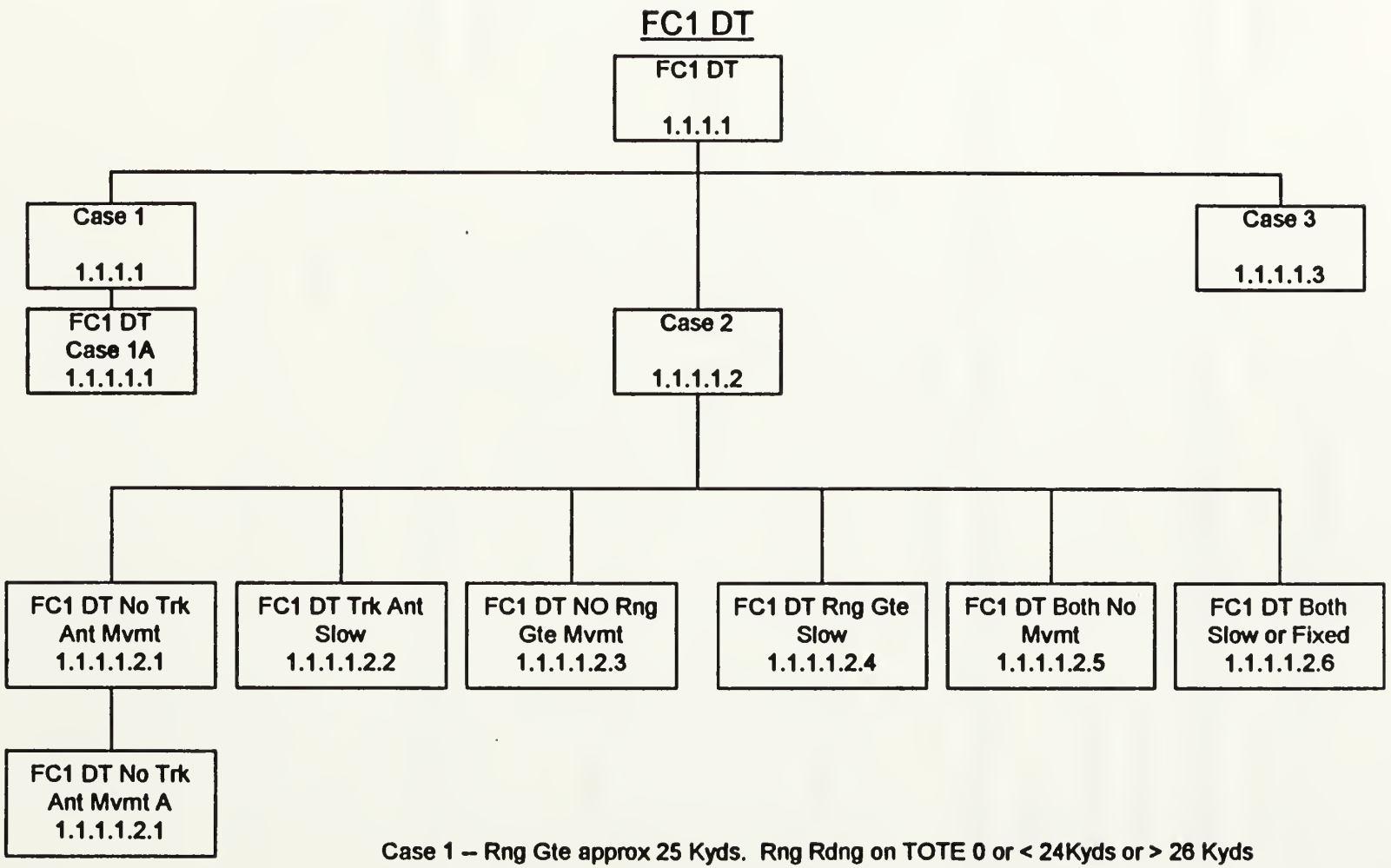
Number: 1.1.1.1.3

Description: Allows trouble shooting of FC1 DT Case 3 procedure.

Called by: FC1 DT Menu.

Calls: None.

==========



Case 1 -- Rng Gte approx 25 Kyds. Rng Rdng on TOTE 0 or < 24Kyds or > 26 Kyds

Case 2 -- Rng Gte approx 25 Kyds. Rng Rdng on TOTE approx 25Kyds.

Case 3 -- Rng Gte not present or no where near 25Kyds.

FIGURE 3

FC1 TB

Name: FC1 TB (FIGURE 4)

Number: 1.1.3.1

Description: Allows selection of one of three FC1 TB modes:

PDT Mode -- Pulse Doppler Transmission

PAT Mode -- Pulse Amplitude Transmission

Both Modes -- PDT and PAT Modes

Called by: FC1 Menu

Calls: FC1 TB PDT Mode, FC1 TB PAT Mode, and FC1 TB Both Modes

==========

Name: FC1 TB PDT Mode

Number: 1.1.3.1.1

Description: Allows trouble shooting of the FC1 TB PDT Mode procedure

Called by: FC1 TB Menu

Calls: FC1 TB C

==========

Name: FC1 TB C

Number: 1.1.3.1.1.1

Description: Continues trouble shooting of FC1 TB PDT Mode procedure

Called by: FC1 TB PDT Mode

Calls: None

==========

Name: FC1 TB PAT Mode

Number: 1.1.3.1.2

Description: Allows trouble shooting of FC1 TB PAT Mode procedure

Called by: FC1 TB Menu

Calls: FC1 TB C

==========

Name: FC1 TB C

Number: 1.1.3.1.1.1

Description: Continues troubleshooting of FC1 TB PAT Mode procedure

Called by: FC1 TB PAT Mode

Calls: None

==========

Name: FC1 TB Both Modes

Number: 1.1.3.1.3

Description: Allows troubleshooting of combined FC1 TB PDT Mode and FC1 TB PAT Mode

Called by: FC1 TB Menu

Calls: FC1 TB Low XTAL Current, FC1 TB Track Antenna Oscillations, and FC1 TB PAT/PDT Common Receiver Circuits

==========

Name: FC1 TB Low XTAL Current

Number: 1.1.3.1.3.1

Description: Allows troubleshooting of FC1 TB Low XTAL Current.

Called by: FC1 TB Both Modes

Calls: FC1 TB TACQ A, FC1 TB TACQ Aa, FC1 TB F

==========

Name: FC1 TB TACQ A

Number: 1.1.3.3.3.1.1

Description: Continues troubleshooting of FC1 TB Low XTAL Current

Called by: FC1 TB Low XTAL Current

Calls: FC1 TB TACQ Aa

==========

Name: FC1 TB TACQ Aa

Number: 1.1.3.3.3.1.1.1

Description: Continues troubleshooting of FC1 TB Low XTAL Current

Called by: FC1 TB TACQ A

Calls: None.

==========

Name: FC1 TB F

Number: 1.1.3.1.3.2.1

Description: Common troubleshooting procedure to FC1 TB Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Called by: FC1 TB Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Calls: FC1 TB C, FC1 TB D

==========

Name: FC1 TB C

Number: 1.1.3.1.1.1

Description: Continues troubleshooting of FC1 TB Both Modes

Called by: FC1 TB PAT Mode

Calls: None

==========

Name: FC1 TB D

Number: 1.1.3.1.3.2.1.1

Description: Continues Troubleshooting of FC1 TB Both Modes.

Called by: FC1 TB F

Calls: FC1 TB Case 1, FC1 TB Case 2, FC1 TB Case 3

==========

Name: FC1 TB Case 1

Number: 1.1.3.1.3.2.1.1.1

Description: Allows troubleshooting of FC1 TB Case 1

Called by: FC1 TB D

Calls: None.

==========

Name: FC1 TB Case 2

Number: 1.1.3.1.3.2.1.1.2

Description: Allows troubleshooting of FC1 TB Case 2

Called by: FC1 TB D

Calls: FC1 TB E

==========

Name: FC1 TB E

Number: 1.1.3.1.3.2.1.1.2.1

Description: Allows troubleshooting of FC1 TB Case 2

Called by: FC1 TB Case 2

Calls: None.

==========

Name: FC1 TB Case 3

Number: 1.1.3.1.3.2.1.1.3

Description: Allows troubleshooting of FC1 TB Case 3

Called by: FC1 TB D

Calls: None.

==========

Name: FC1 TB Track Antenna Oscillations

Number: 1.1.3.1.3.2

Description: Allows troubleshooting of FC1 TB Low XTAL Current.

Called by: FC1 TB Both Modes

Calls: FC1 TB TACQ A, FC1 TB TACQ Aa, FC1 TB F

==========

Name: FC1 TB F

Number: 1.1.3.1.3.2.1

Description: Common troubleshooting procedure to FC1 TB Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Called by: FC1 TB Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Calls: FC1 TB C, FC1 TB D

==========

Name: FC1 TB PAT/PDT Common Receiver Circuits

Number: 1.1.3.1.3.3

Description: Allows troubleshooting of FC1 TB Common Receiver Circuits

Called by: FC1 TB Both Modes

Calls: FC1 TB F

==========

Name: FC1 TB F

Number: 1.1.3.1.3.2.1

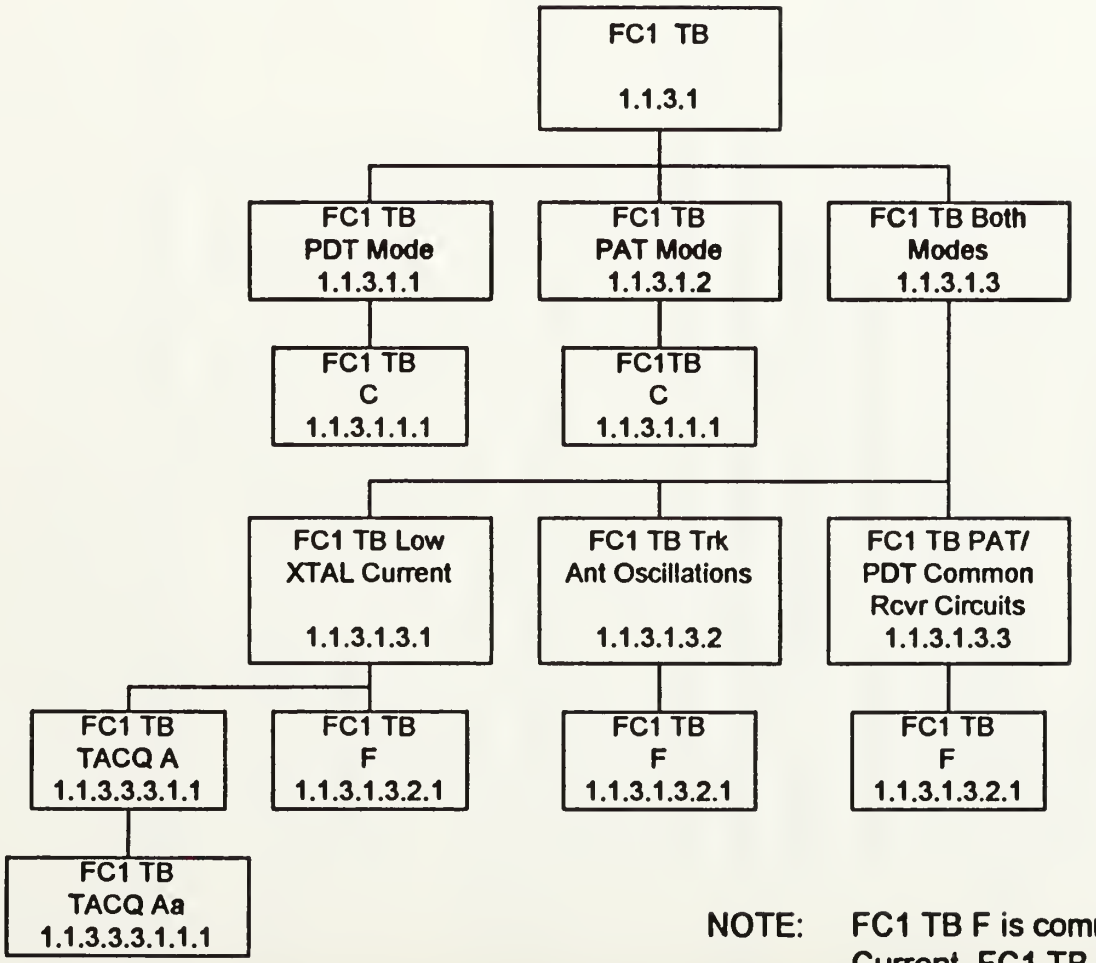
Description: Common troubleshooting procedure to FC1 TB Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Called by: FC1 TB Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Calls: FC1 TB C, FC1 TB D

==========

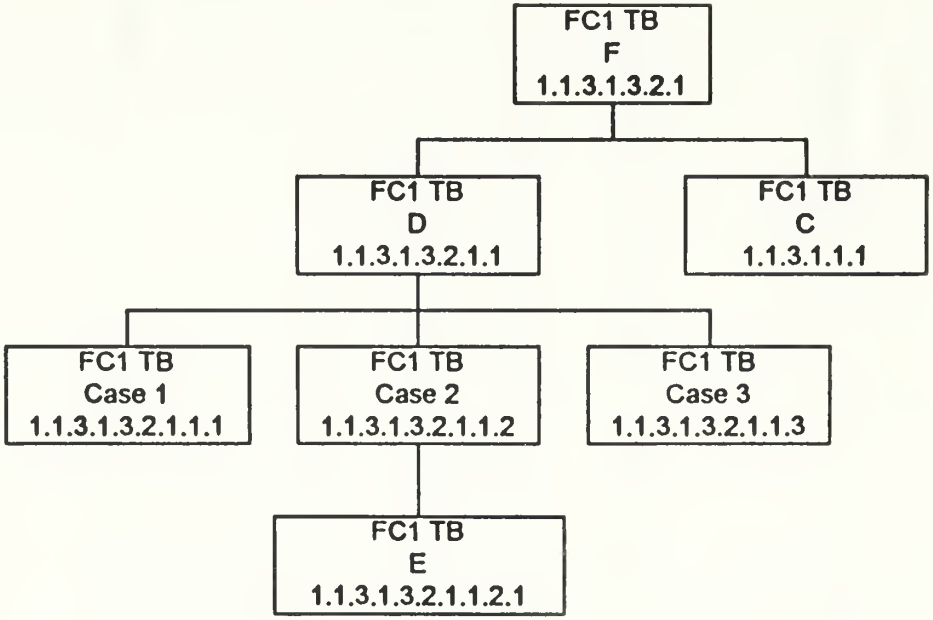
FC1 TB



NOTE: FC1 TB F is common to FC1 TB Low XTAL Current, FC1 TB Trk Ant Oscillations, and FC1 TB PAT/PDT Common Receiver Circuits.

FIGURE 4

FC1 TB F



103

FIGURE 4A

FC1 TE

Name: FC1 TE (FIGURE 5)

Number: 1.1.3.2

Description: Allows selection of FC1 TE Modes

PDT Mode -- Pulse Doppler Mode

PAT Mode -- Pulse Amplitude Mode

Both Modes -- PAT/PDT Modes

Called by: FC1 Menu

Calls: FC1 TE PDT Mode, FC1 TE PAT Mode, FC1 TE Both Modes

==========

Name: FC1 TE PDT Mode

Number: 1.1.3.2.1

Description: Allows troubleshooting of FC1 TE

Called by: FC1 TE

Calls: FC1 TE C

==========

Name: FC1 TE C

Number: 1.1.3.2.1.1

Description: Continues troubleshooting of FC1 TE PDT Mode

Called by: FC1 TE PDT Mode

Calls: None

==========

Name: FC1 TE PAT Mode

Number: 1.1.3.2.2

Description: Allows troubleshooting of FC1 TE

Called by: FC1 TE

Calls: FC1 TE C

==========

Name: FC1 TE C

Number: 1.1.3.2.1.1

Description: Continues troubleshooting of FC1 TE PAT Mode

Called by: FC1 TE PDT Mode

Calls: None

==========

Name: FC1 TE Both Modes

Number: 1.1.3.2.3

Description: Allows troubleshooting of combined FC1 TE PDT Mode and FC1 TE PAT Mode

Called by: FC1 TE Menu

Calls: FC1 TE Low XTAL Current, FC1 TE Track Antenna Oscillations, and FC1 TE PAT/PDT Common Receiver Circuits

==========

Name: FC1 TE Low XTAL Current

Number: 1.1.3.2.3.1

Description: Allows troubleshooting of FC1 TE Low XTAL Current.

Called by: FC1 TE Both Modes

Calls: FC1 TE TACQ A, FC1 TE TACQ Aa, FC1 TE F

==========

Name: FC1 TE TACQ A

Number: 1.1.3.3.3.1.1

Description: Continues troubleshooting of FC1 TE Low XTAL Current

Called by: FC1 TE Low XTAL Current

Calls: FC1 TE TACQ Aa

==========

Name: FC1 TE TACQ Aa

Number: 1.1.3.3.3.1.1.1

Description: Continues troubleshooting of FC1 TE Low XTAL Current

Called by: FC1 TE TACQ A

Calls: None.

==========

Name: FC1 TE F

Number: 1.1.3.2.3.2.1

Description: Common troubleshooting procedure to FC1 TE Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Called by: FC1 TE Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Calls: FC1 TE C, FC1 TE D

==========

Name: FC1 TE C

Number: 1.1.3.2.1.1

Description: Continues troubleshooting of FC1 TE Both Modes

Called by: FC1 TE F

Calls: None

==========

Name: FC1 TE D

Number: 1.1.3.2.3.2.1.1

Description: Continues Troubleshooting of FC1 TE Both Modes.

Called by: FC1 TE F

Calls: FC1 TE Case 1, FC1 TE Case 2, FC1 TE Case 3

==========

Name: FC1 TE Case 1

Number: 1.1.3.2.3.2.1.1.1

Description: Allows troubleshooting of FC1 TE Case 1

Called by: FC1 TE D

Calls: None.

==========

Name: FC1 TE Case 2

Number: 1.1.3.2.3.2.1.1.2

Description: Allows troubleshooting of FC1 TE Case 2

Called by: FC1 TE D

Calls: FC1 TE E

==========

Name: FC1 TE E

Number: 1.1.3.2.3.2.1.1.2.1

Description: Allows troubleshooting of FC1 TE Case 2

Called by: FC1 TE Case 2

Calls: None.

==========

Name: FC1 TE Case 3

Number: 1.1.3.2.3.2.1.1.3

Description: Allows troubleshooting of FC1 TE Case 3

Called by: FC1 TE D

Calls: None.

==========

Name: FC1 TE Track Antenna Oscillations

Number: 1.1.3.2.3.2

Description: Allows troubleshooting of FC1 TE Track Antenna Oscillations

Called by: FC1 TE Both Modes

Calls: FC1 TE F

==========

Name: FC1 TE F

Number: 1.1.3.2.3.2.1

Description: Common troubleshooting procedure to FC1 TE Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Called by: FC1 TE Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Calls: FC1 TE C, FC1 TE D

==========

Name: FC1 TE PAT/PDT Common Receiver Circuits

Number: 1.1.3.2.3.3

Description: Allows troubleshooting of FC1 TE Common Receiver Circuits

Called by: FC1 TE Both Modes

Calls: FC1 TE F

==========

Name: FC1 TE F

Number: 1.1.3.2.3.2.1

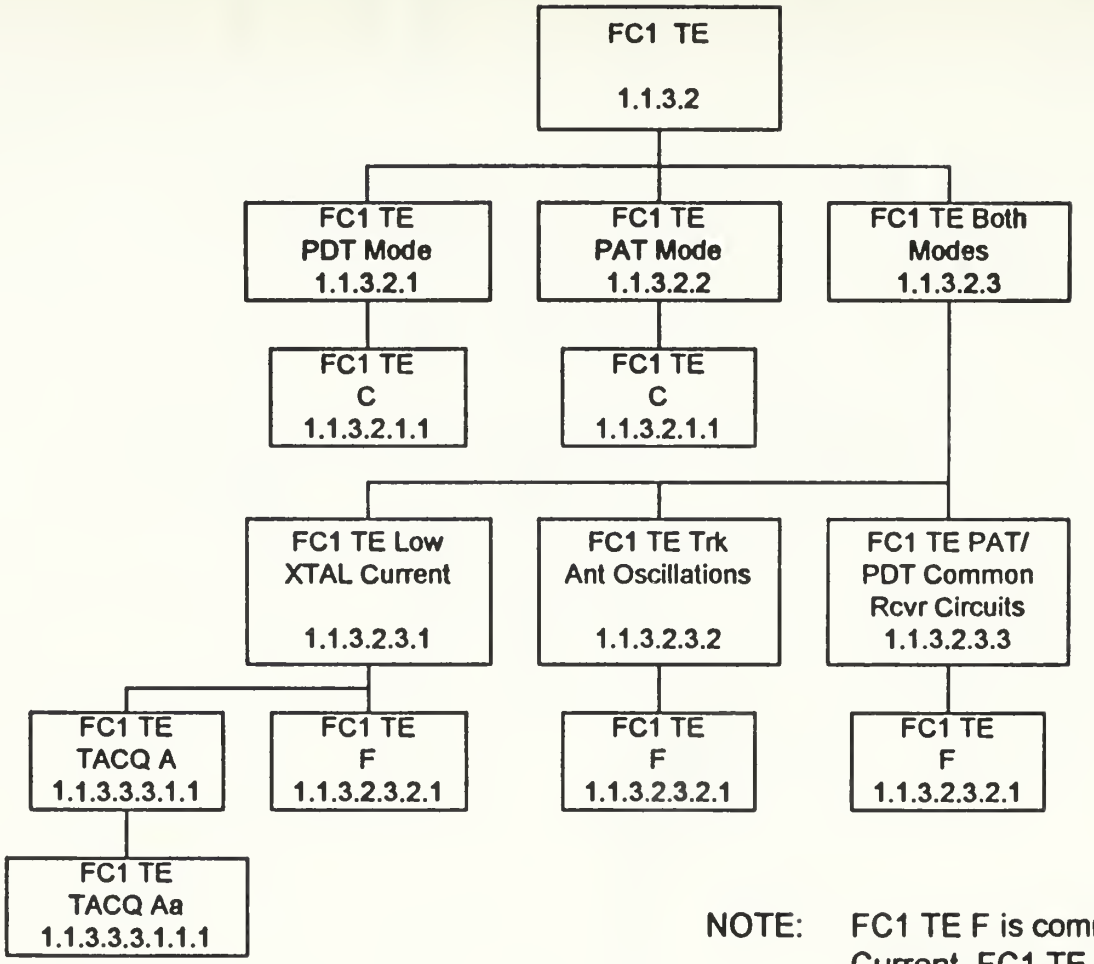
Description: Common troubleshooting procedure to FC1 TE Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Called by: FC1 TE Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Calls: FC1 TE C, FC1 TE D

==========

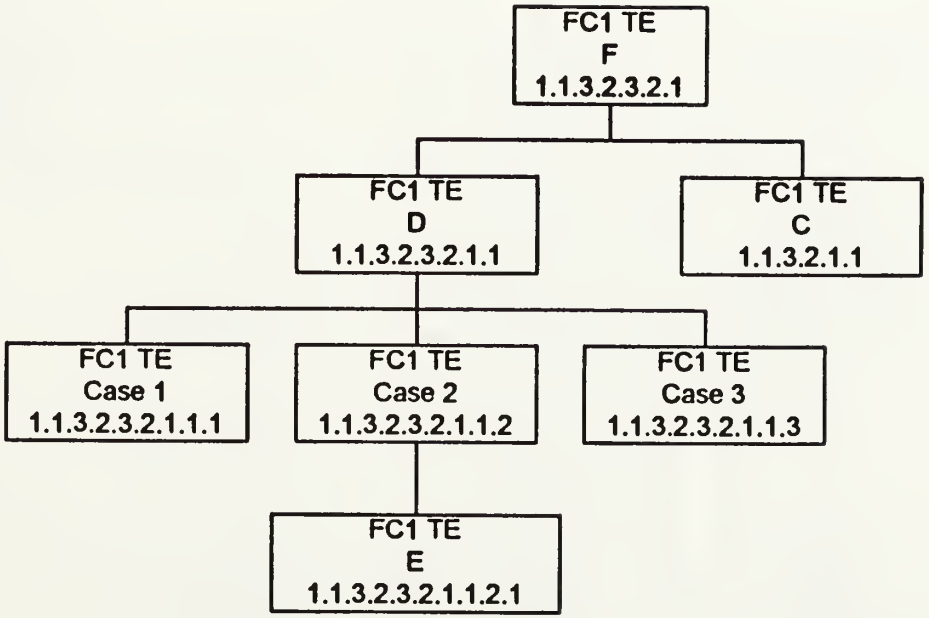
FC1 TE



NOTE: FC1 TE F is common to FC1 TE Low XTAL Current, FC1 TE Trk Ant Oscillations, and FC1 TE PAT/PDT Common Receiver Circuits.

FIGURE 5

FC1 TE F



112

FIGURE 5A

FC1 TR

Name: FC1 TR (FIGURE 6)

Number: 1.1.3.3

Description: Allows selection of FC1 TR Modes

PDT Mode -- Pulse Doppler Mode

PAT Mode -- Pulse Amplitude Mode

Both Modes -- PAT/PDT Modes

Called by: FC1 Menu

Calls: FC1 TR PDT Mode, FC1 TR PAT Mode, FC1 TR Both Modes

==========

Name: FC1 TR PDT Mode

Number: 1.1.3.3.1

Description: Allows troubleshooting of FC1 TR

Called by: FC1 TR

Calls: FC1 TR C

==========

Name: FC1 TR C

Number: 1.1.3.3.1.1

Description: Continues troubleshooting of FC1 TR PDT Mode

Called by: FC1 TR PDT Mode

Calls: None

==========

Name: FC1 TR PAT Mode

Number: 1.1.3.3.2

Description: Allows troubleshooting of FC1 TR

Called by: FC1 TR

Calls: FC1 TR C

==========

Name: FC1 TR C

Number: 1.1.3.3.1.1

Description: Continues troubleshooting of FC1 TR PAT Mode

Called by: FC1 TE PDT Mode

Calls: None

==========

Name: FC1 TR Both Modes

Number: 1.1.3.3.3

Description: Allows troubleshooting of combined FC1 TR PDT Mode and FC1 TR PAT Mode

Called by: FC1 TR Menu

Calls: FC1 TR Low XTAL Current, TR Gate Circuits, TR D, TR Transmitter Microwave

==========

Name: FC1 TR Low XTAL Current

Number: 1.1.3.3.3.1

Description: Allows troubleshooting of FC1 TR Low XTAL Current

Called by: FC1 TR Both Modes

Calls: FC1 TR TACQ A, FC1 TR TACQ Aa, FC1 TR D

==========

Name: FC1 TR TACQ A

Number: 1.1.3.3.3.1.1

Description: Continues troubleshooting of FC1 TR Low XTAL Current

Called by: FC1 TR Low XTAL Current

Calls: FC1 TR TACQ Aa

==========

Name: FC1 TR TACQ Aa

Number: 1.1.3.3.3.1.1.1

Description: Continues troubleshooting of FC1 TR Low XTAL Current

Called by: FC1 TR TACQ A

Calls: None.

==========

Name: FC1 TR D

Number: 1.1.3.3.3.3

Description: Continues troubleshooting of FC1 TR Both Modes

Called by: FC1 TR Both Modes

Calls: FC1 TR C, FC1 TR Sub D, FC1 TR E

==========

Name: FC1 TR C

Number: 1.1.3.3.1.1

Description: Continues troubleshooting of FC1 TR D

Called by: FC1 TR D

Calls: None

==========

Name: FC1 TR Sub D

Number: 1.1.3.3.3.3.1

Description: Continues troubleshooting of FC1 TR D

Called by: FC1 TR D

Calls: FC1 TR E

==========

Name: FC1 TR E

Number: 1.1.3.3.3.3.1.1

Description: Continues troubleshooting of FC1 TR D

Called by: FC1 TR Sub D

Calls: None

==========

Name: FC1 TR Gate Circuits

Number: 1.1.3.3.3.2

Description: Continues troubleshooting of FC1 TR Both Modes

Called by: FC1 TR Both Modes

Calls: FC1 TR D

==========

Name: FC1 TR D

Number: 1.1.3.3.3.3

Description: Continues troubleshooting of FC1 TR Gate Circuits

Called by: FC1 TR Gate Circuits

Calls: FC1 TR C, FC1 TR Sub D, FC1 TR E

==========

Name: FC1 TR Trans Micro

Number: 1.1.3.3.3.4

Description: Continues troubleshooting of FC1 TR Both Modes

Called by: FC1 TR Both Modes

Calls: FC1 TR D

==========

Name: FC1 TR D

Number: 1.1.3.3.3.3

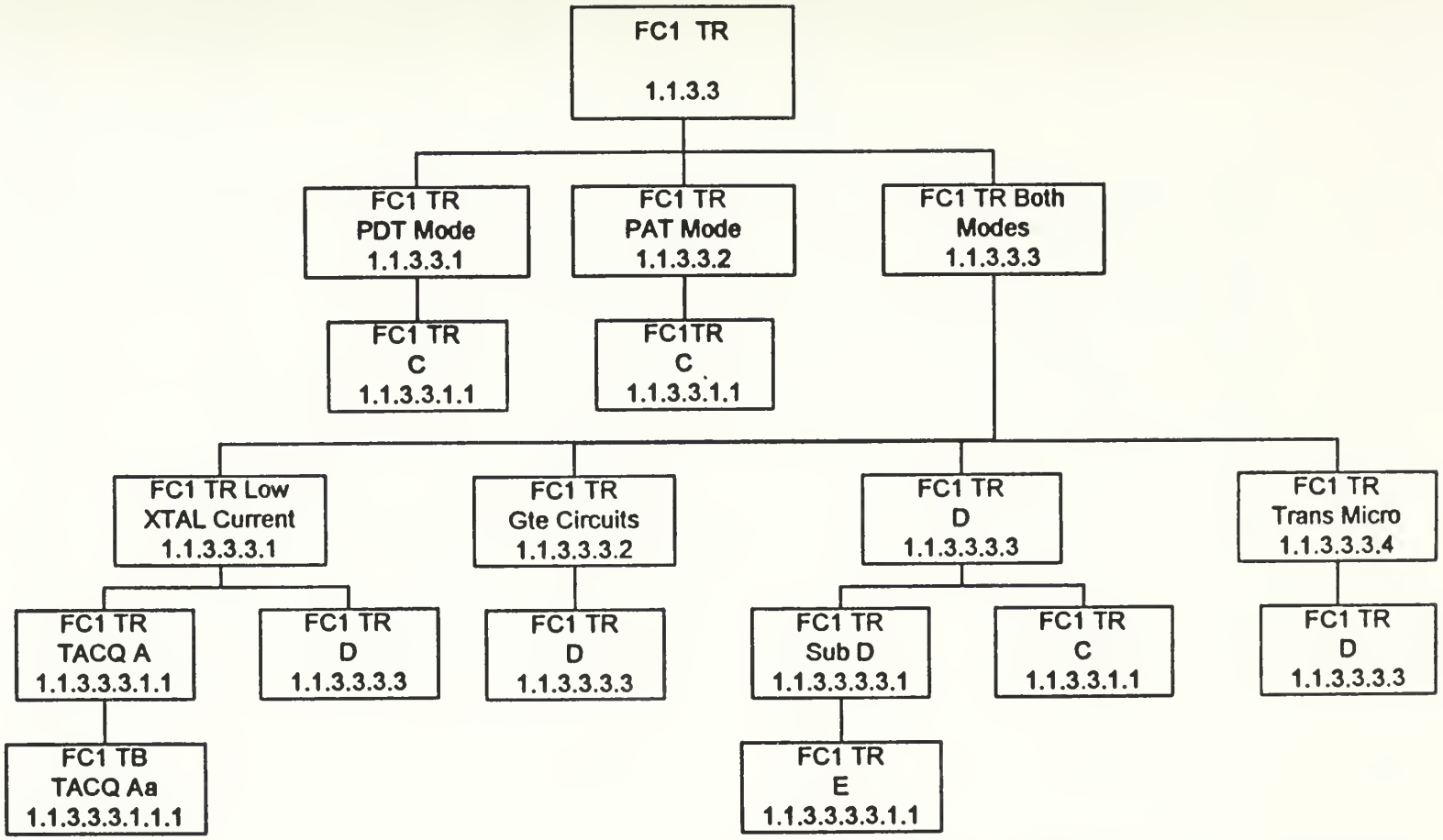
Description: Continues troubleshooting of FC1 TR Trans Micro

Called by: FC1 TR Trans Micro

Calls: FC1 TR C, FC1 TR Sub D, FC1 TR E

==========

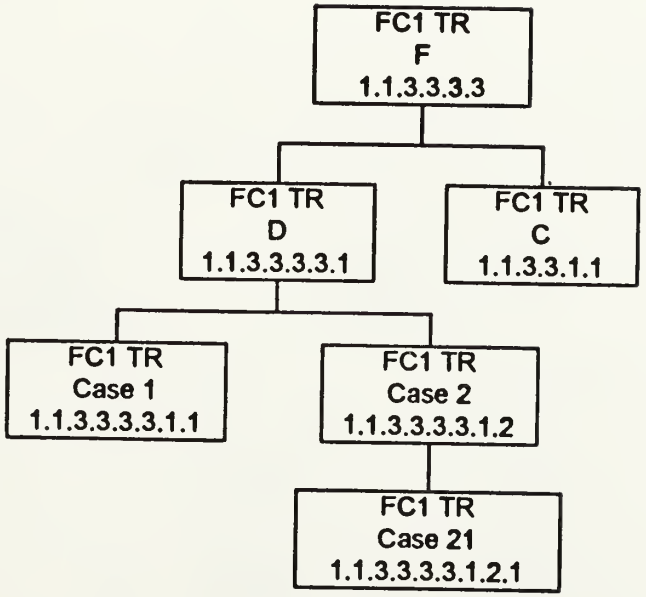
FC1 TR



NOTE: FC1 TR D is common to FC1 TR Low XTAL Current, FC1 TR Gte Circuits, and FC1 TR Trans Micro.

FIGURE 6

FC1 TR F



120

FIGURE 6A

FC2 MENU

Name: FC2 (FIGURE 7)

Number: 1.2

Description: Allows selection of FC2 Designation; Time, Range, and Bearing. FC2 ACQ. FC2 Track; Bearing, Elevation, and Range

Called by: FC2 Performance Menu

Calls: FC2DTRB, FC2ACQ, FC2TBER

==========

Name: FC2 DTBR

Number: 1.2.1

Description: Allows selection of FC2 Designation; Time, Range, and Bearing procedures

Called by: FC2 Menu

Calls: FC2DT, FC2TR, and FC2TB

==========

Name: FC2 ACQ

Number: 1.2.2

Description: Allows selection of FC2 ACQ procedures

Called by: FC2 Menu

Calls: See FC2 ACQ Menu

==========

Name: FC2 TBER Menu

Number: 1.2.3

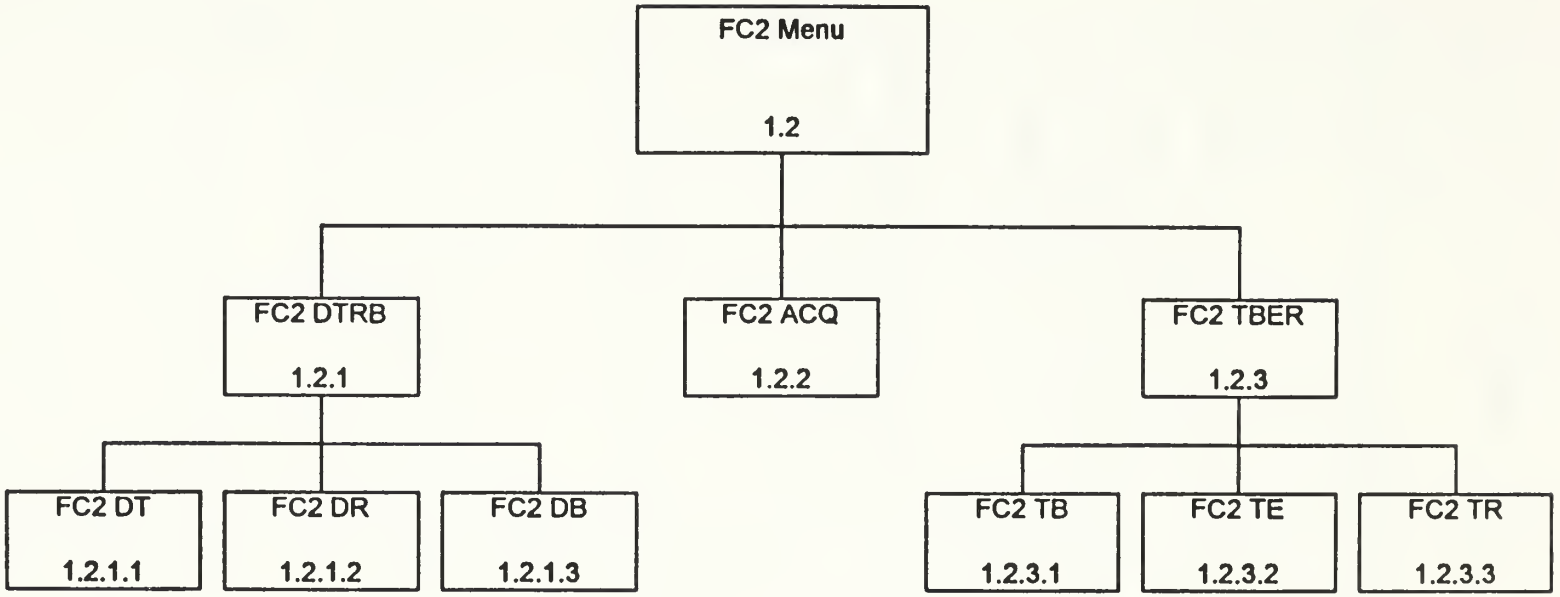
Description: Allows selection of FC2 Track; Bearing,Elevation, and Range procedures

Called by: FC2 Menu

Calls: FC2 TB, FC2 TE, FC2 TR

==========

FC2 Menu



123

FIGURE 7

FC2 TB

Name: FC2 TB (FIGURE 8)

Number: 1.2.3.1

Description: Allows selection of one of three FC2 TB modes:

PDT Mode -- Pulse Doppler Mode

PAT Mode -- Pulse Amplitude Mode

Both Modes -- PAT and PDT Modes

CALLED BY: FC2 TBER Menu

CALLS: FC2 TB PDT Mode, FC2 TB Mode, and FC2 TB Mode

==========

Name: FC2 TB PDT Mode

Number: 1.2.3.1.1

Description: Allows trouble shooting of the FC2 TB PDT Mode procedure

Called by: FC2 TB Menu

Calls: FC2 TB C

==========

Name: FC2 TB C

Number: 1.2.3.1.1.1

Description: Continues trouble shooting of FC2 TB PDT Mode procedure

Called by: FC2 TB PDT Mode

Calls: None

==========

Name: FC2 TB PAT Mode

Number: 1.2.3.1.2

Description: Allows trouble shooting of FC2 TB PAT Mode procedure

Called by: FC2 TB Menu

Calls: FC2 TB C

==========

Name: FC2 TB C

Number: 1.2.3.1.1.1

Description: Continues troubleshooting of FC2 TB PAT Mode procedure

Called by: FC2 TB PAT Mode

Calls: None

==========

Name: FC2 TB Both Modes

Number: 1.2.3.1.3

Description: Allows troubleshooting of combined FC2 TB PDT Mode and FC2 TB PAT Mode

Called by: FC2 TB Menu

Calls: FC2 TB Low XTAL Current, Track Antenna Oscillations, and PAT/PDT Common Receiver Circuits

==========

Name: FC2 TB Low XTAL Current

Number: 1.2.3.1.3.1

Description: Allows troubleshooting of FC2 TB Low XTAL Current

Called by: FC2 TB Both Modes

Calls: FC2 TB TACQ B, FC2 TB TACQ Ba, FC2 TB F

==========

Name: FC1 TB TACQ B

Number: 1.2.3.3.3.1.1

Description: Continues troubleshooting of FC2 TB Low XTAL Current

Called by: FC2 TB Low XTAL Current

Calls: FC2 TB TACQ Ba

==========

Name: FC2 TB TACQ Ba

Number: 1.2.3.3.3.1.1.1

Description: Continues troubleshooting of FC2 TB Low XTAL Current

Called by: FC1 TB TACQ B

Calls: None

==========

Name: FC2 TB F

Number: 1.2.3.1.3.2.1

Description: Common troubleshooting procedure to FC2 TB Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Called by: FC2 TB Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Calls: FC2 TB C, FC2 TB D

==========

Name: FC2 TB C

Number: 1.2.3.1.1.1

Description: Continues troubleshooting of FC2 TB Both Modes

Called by: FC2 TB PAT Mode

Calls: None

==========

Name: FC2 TB D

Number: 1.2.3.1.3.2.1.1

Description: Continues Troubleshooting of FC2 TB Both Modes.

Called by: FC2 TB F

Calls: FC2 TB Case 1, FC2 TB Case 2, FC2 TB Case 3

==========

Name: FC2 TB Case 1

Number: 1.2.3.1.3.2.1.1.1

Description: Allows troubleshooting of FC2 TB Case 1

Called by: FC2 TB D

Calls: None

==========

Name: FC2 TB Case 2

Number: 1.2.3.1.3.2.1.1.2

Description: Allows troubleshooting of FC2 TB Case 2

Called by: FC2 TB D

Calls: FC2 TB Case 21

==========

Name: FC2 TB Case 21

Number: 1.2.3.1.3.2.1.1.2.1

Description: Allows troubleshooting of FC2 TB Case 2

Called by: FC2 TB Case 2

Calls: None

==========

Name: FC2 TB Case 3

Number: 1.2.3.1.3.2.1.1.3

Description: Allows troubleshooting of FC2 TB Case 3

Called by: FC2 TB D

Calls: FC2 TB Case 3A

==========

Name: FC2 TB Case 3A

Number: 1.2.3.1.3.2.1.1.3.1

Description: Allows troubleshooting of FC2 TB Case 3

Called by: FC2 TB Case 3

Calls: None

==========

Name: FC2 TB Track Antenna Oscillations

Number: 1.2.3.1.3.2

Description: Allows troubleshooting of FC2 TB Both Modes

Called by: FC2 TB Both Modes

Calls: FC2 TB F

==========

Name: FC2 TB F

Number: 1.2.3.1.3.2.1

Description: Common troubleshooting procedure to FC2 TB Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Called by: FC2 TB Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Calls: FC2 TB C, FC2 TB D

==========

Name: FC2 TB PAT/PDT Common Receiver Circuits

Number: 1.2.3.1.3.3

Description: Allows troubleshooting of FC2 TB Common Receiver Circuits

Called by: FC2 TB Both Modes

Calls: FC2 TB F

==========

Name: FC2 TB F

Number: 1.2.3.1.3.2.1

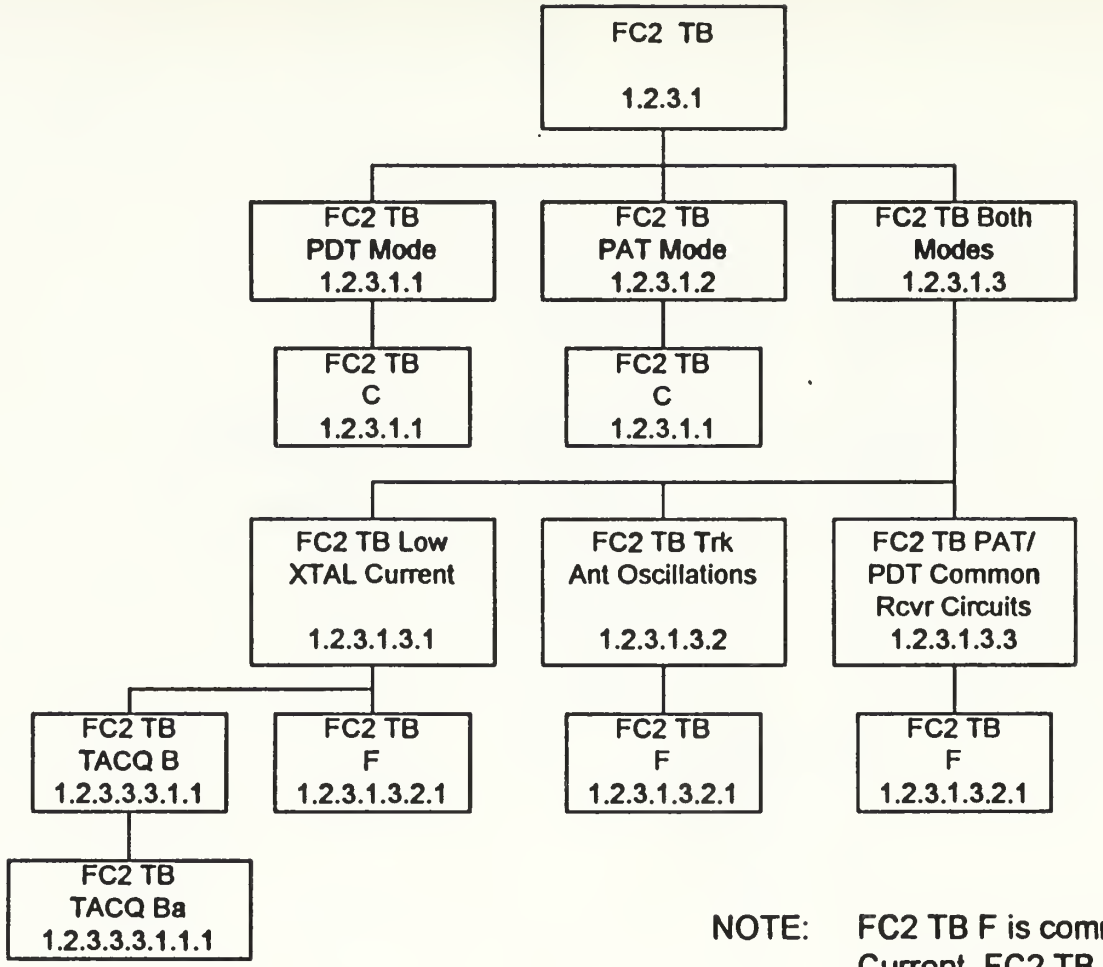
Description: Common troubleshooting procedure to FC2 TB Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Called by: FC2 TB Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Calls: FC2 TB C, FC2 TB D

==========

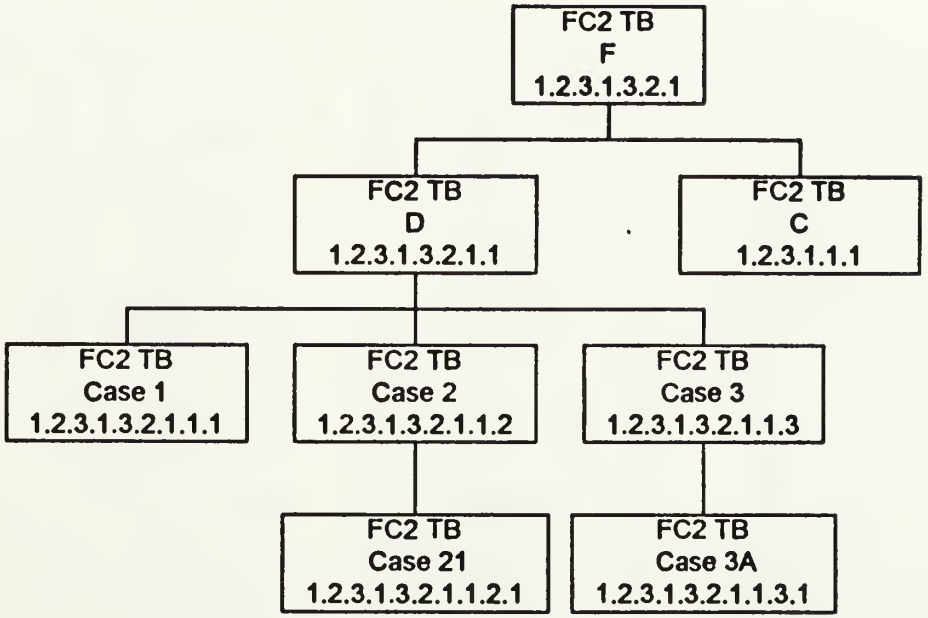
FC2 TB



NOTE: FC2 TB F is common to FC2 TB Low XTAL Current, FC2 TB Trk Ant Oscillations, and FC2 TB PAT/PDT Common Receiver Circuits.

FIGURE 8

FC2 TB F



132

FIGURE 8A

FC2 TE

Name: FC2 TE (FIGURE 9)

Number: 1.2.3.2

Description: Allows selection of FC2 TE Modes

PDT Mode -- Pulse Doppler Mode

PAT Mode -- Pulse Amplitude Mode

Both Modes -- PAT/PDT Modes

Called by: FC2 Menu

Calls: FC2 TE PDT Mode, FC2 TE PAT Mode, FC2 TE Both Modes

==========

Name: FC2 TE PDT Mode

Number: 1.2.3.2.1

Description: Allows troubleshooting of FC2 TE

Called by: FC2 TE

Calls: FC2 TE C

==========

Name: FC2 TE C

Number: 1.2.3.2.1.1

Description: Continues troubleshooting of FC2 TE PDT Mode

Called by: FC2 TE PDT Mode

Calls: None

==========

Name: FC2 TE PAT Mode

Number: 1.2.3.2.2

Description: Allows troubleshooting of FC2 TE

Called by: FC2 TE

Calls: FC2 TE C

==========

Name: FC2 TE C

Number: 1.2.3.2.1.1

Description: Continues troubleshooting of FC2 TE PAT Mode

Called by: FC2 TE PDT Mode

Calls: None

==========

Name: FC2 TE Both Modes

Number: 1.2.3.2.3

Description: Allows troubleshooting of combined FC2 TE PDT Mode and FC2 TE PAT Mode

Called by: FC2 TE Menu

Calls: FC2 TE Low XTAL Current, Track Antenna Oscillations, and PAT/PDT Common Receiver Circuits

==========

Name: FC2 TE Low XTAL Current

Number: 1.2.3.2.3.1

Description: Allows troubleshooting of FC2 TE Low XTAL Current

Called by: FC2 TE Both Modes

Calls: FC2 TE TACQ B, FC2 TE TACQ Ba, FC2 TE F

==========

Name: FC2 TE TACQ B

Number: 1.2.3.3.3.1.1

Description: Continues troubleshooting of FC2 TE Low XTAL Current

Called by: FC2 TE Low XTAL Current

Calls: FC2 TE TACQ Ba

==========

Name: FC2 TE TACQ Ba

Number: 1.2.3.3.3.1.1.1

Description: Continues troubleshooting of FC2 TE Low XTAL Current

Called by: FC2 TE TACQ B

Calls: None

==========

Name: FC2 TE F

Number: 1.2.3.2.3.2.1

Description: Common troubleshooting procedure to FC2 TE Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Called by: FC2 TE Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Calls: FC2 TE C, FC2 TE D

==========

Name: FC2 TE C

Number: 1.2.3.2.1.1

Description: Continues troubleshooting of FC2 TE Both Modes

Called by: FC2 TE F

Calls: None

==========

Name: FC2 TE D

Number: 1.2.3.2.3.2.1.1

Description: Continues Troubleshooting of FC2 TE Both Modes.

Called by: FC2 TE F

Calls: FC2 TE Case 1, FC2 TE Case 2, FC2 TE Case 3

==========

Name: FC2 TE Case 1

Number: 1.2.3.2.3.2.1.1.1

Description: Allows troubleshooting of FC2 TE Case 1

Called by: FC2 TE D

Calls: None

==========

Name: FC2 TE Case 2

Number: 1.2.3.2.3.2.1.1.2

Description: Allows troubleshooting of FC1 TE Case 2

Called by: FC2 TE D

Calls: FC2 TE Case 21

==========

Name: FC2 TE Case 21

Number: 1.2.3.2.3.2.1.1.2.1

Description: Allows troubleshooting of FC2 TE Case 2

Called by: FC2 TE Case 2

Calls: None

==========

Name: FC2 TE Case 3

Number: 1.2.3.2.3.2.1.1.3

Description: Allows troubleshooting of FC2 TE Case 3

Called by: FC2 TE D

Calls: FC2 TE Case 3A

==========

Name: FC2 TE Case 3A

Number: 1.2.3.2.3.2.1.1.3.1

Description: Allows troubleshooting of FC2 TE Case 3

Called by: FC2 TE Case 3

Calls: None

==========

Name: FC2 TE Track Antenna Oscillations

Number: 1.2.3.2.3.2

Description: Allows troubleshooting of FC2 TE Track Antenna Oscillations

Called by: FC2 TE Both Modes

Calls: FC2 TE F

==========

Name: FC2 TE F

Number: 1.2.3.2.3.2.1

Description: Common troubleshooting procedure to FC2 TE Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Called by: FC2 TE Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Calls: FC2 TE C, FC2 TE D

==========

Name: FC2 TE PAT/PDT Common Receiver Circuits

Number: 1.2.3.2.3.3

Description: Allows troubleshooting of FC2 TE Common Receiver Circuits

Called by: FC2 TE Both Modes

Calls: FC2 TE F

==========

Name: FC2 TE F

Number: 1.2.3.2.3.2.1

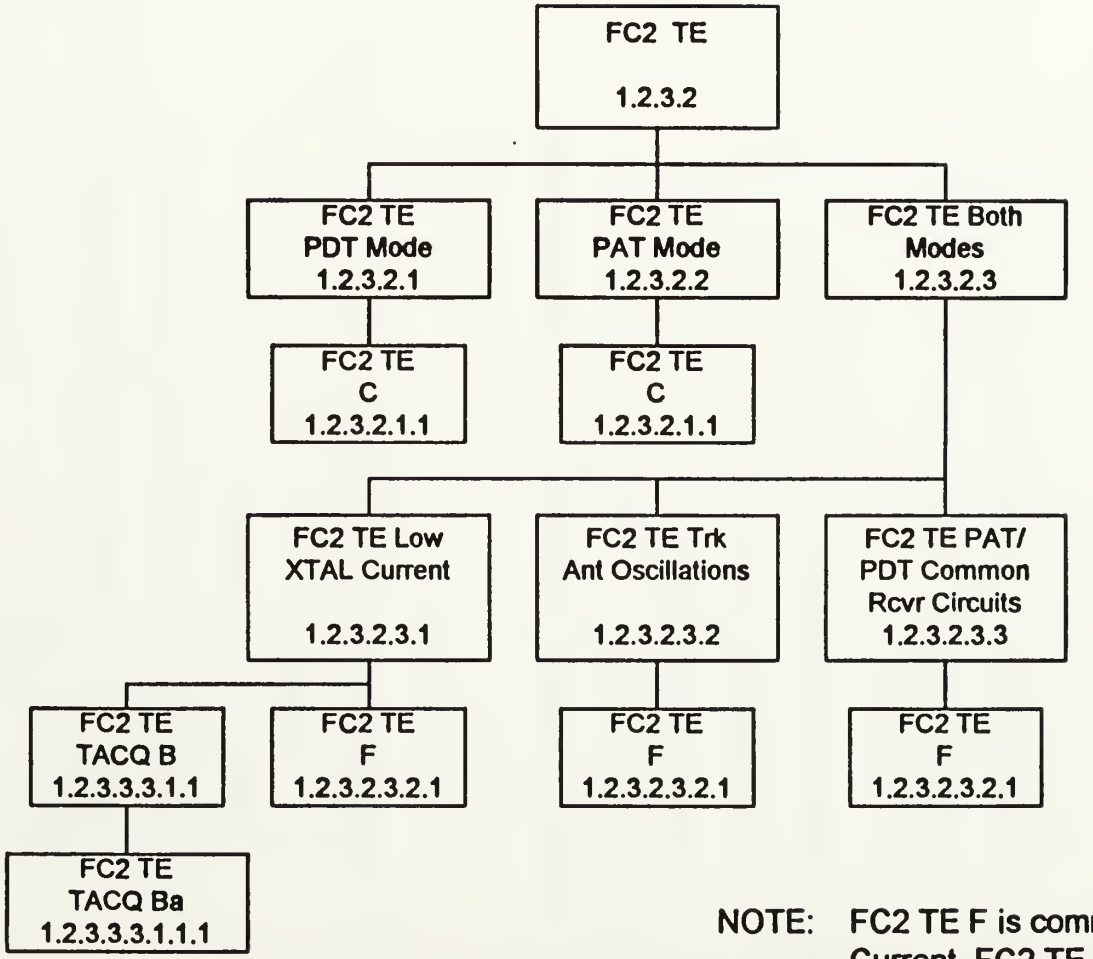
Description: Common troubleshooting procedure to FC2 TE Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Called by: FC2 TE Low XTAL Current, Track Antenna Oscillations and PAT/PDT Common Receiver Circuits

Calls: FC2 TE C, FC2 TE D

==========

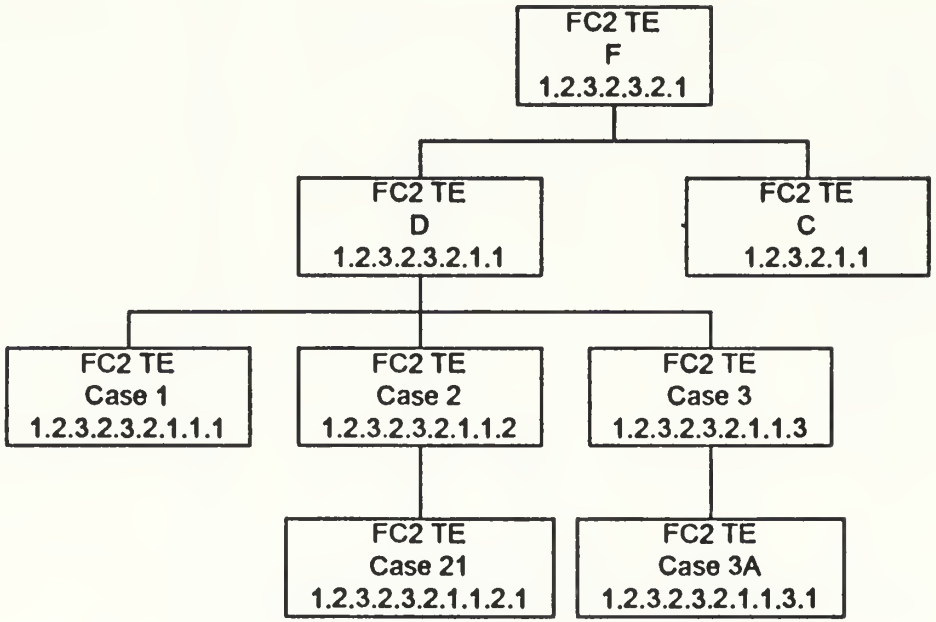
FC2 TE



NOTE: FC2 TE F is common to FC2 TE Low XTAL Current, FC2 TE Trk Ant Oscillations, and FC2 TE PAT/PDT Common Receiver Circuits.

FIGURE 9

FC2 TE F



141

FIGURE 9A

FC2 TR

Name: FC2 TR (FIGURE 10)

Number: 1.2.3.3

Description: Allows selection of FC2 TR Modes

PDT Mode -- Pulse Doppler Mode

PAT Mode -- Pulse Amplitude Mode

Both Modes -- PAT/PDT Modes

Called by: FC2 Menu

Calls: FC2 TR PDT Mode, FC2 TR PAT Mode, FC2 TR Both Modes

==========

Name: FC2 TR PDT Mode

Number: 1.2.3.3.1

Description: Allows troubleshooting of FC2 TR

Called by: FC2 TR

Calls: FC2 TR C

==========

Name: FC2 TR C

Number: 1.2.3.3.1.1

Description: Continues troubleshooting of FC2 TR PDT Mode

Called by: FC2 TR PDT Mode

Calls: None

==========

Name: FC2 TR PAT Mode

Number: 1.2.3.3.2

Description: Allows troubleshooting of FC2 TR

Called by: FC2 TR

Calls: FC2 TR C

==========

Name: FC2 TR C

Number: 1.2.3.3.1.1

Description: Continues troubleshooting of FC2 TR PAT Mode

Called by: FC2 TE PDT Mode

Calls: None

==========

Name: FC2 TR Both Modes

Number: 1.2.3.3.3

Description: Allows troubleshooting of combined FC2 TR PDT Mode and FC2 TR PAT Mode

Called by: FC2 TR Menu

Calls: FC2 TR Low XTAL Current, Gate Circuits, F, Transmitter Microwave

==========

Name: FC2 TR Low XTAL Current

Number: 1.2.3.3.3.1

Description: Allows troubleshooting of FC2 TR Low XTAL Current

Called by: FC2 TR Both Modes

Calls: FC2 TR TACQ B, FC2 TR TACQ Ba, FC2 TR F

==========

Name: FC2 TR TACQ B

Number: 1.2.3.3.3.1.1

Description: Continues troubleshooting of FC2 TR Low XTAL Current

Called by: FC2 TR Low XTAL Current

Calls: FC2 TR TACQ Ba

==========

Name: FC2 TR TACQ Ba

Number: 1.2.3.3.3.1.1.1

Description: Continues troubleshooting of FC2 TR Low XTAL Current

Called by: FC2 TR TACQ B

Calls: None

==========

Name: FC2 TR F

Number: 1.2.3.3.3.3

Description: Continues troubleshooting of FC2 TR Both Modes

Called by: FC2 TR Both Modes

Calls: FC2 TR C, FC2 TR D

==========

Name: FC2 TR C

Number: 1.2.3.3.1.1

Description: Continues troubleshooting of FC2 TR F

Called by: FC2 TR F

Calls: None

==========

Name: FC2 TR D

Number: 1.2.3.3.3.3.1

Description: Continues troubleshooting of FC2 TR F

Called by: FC2 TR F

Calls: FC2 TR Case 1, FC1 TR Case 2

==========

Name: FC2 TR Case 1

Number: 1.2.3.3.3.3.1.1

Description: Continues troubleshooting of FC1 TR D

Called by: FC1 TR D

Calls: None

==========

Name: FC2 TR Case 2

Number: 1.2.3.3.3.3.1.2

Description: Continues troubleshooting of FC1 TR D

Called by: FC1 TR D

Calls: FC2 TR Case 21

==========

Name: FC2 TR Case 21

Number: 1.2.3.3.3.3.1.2.1

Description: Continues troubleshooting of FC1 TR D

Called by: FC1 TR Case 2

Calls: None

==========

Name: FC2 TR Gate Circuits

Number: 1.2.3.3.3.2

Description: Continues troubleshooting of FC2 TR Both Modes

Called by: FC2 TR Both Modes

Calls: FC2 TR F

==========

Name: FC2 TR F

Number: 1.2.3.3.3.3

Description: Continues troubleshooting of FC2 TR Gate Circuits

Called by: FC2 TR Gate Circuits

Calls: FC2 TR C, FC2 TR D

==========

Name: FC2 TR F

Number: 1.2.3.3.3.3

Description: Continues troubleshooting of FC2 TR Both Modes

Called by: FC2 TR Both Modes

Calls: FC2 TR C, FC2 TR D

==========

Name: FC2 TR Trans Micro

Number: 1.2.3.3.3.4

Description: Continues troubleshooting of FC2 TR Both Modes

Called by: FC2 TR Both Modes

Calls: FC2 TR F

==========

Name: FC2 TR F

Number: 1.2.3.3.3.3

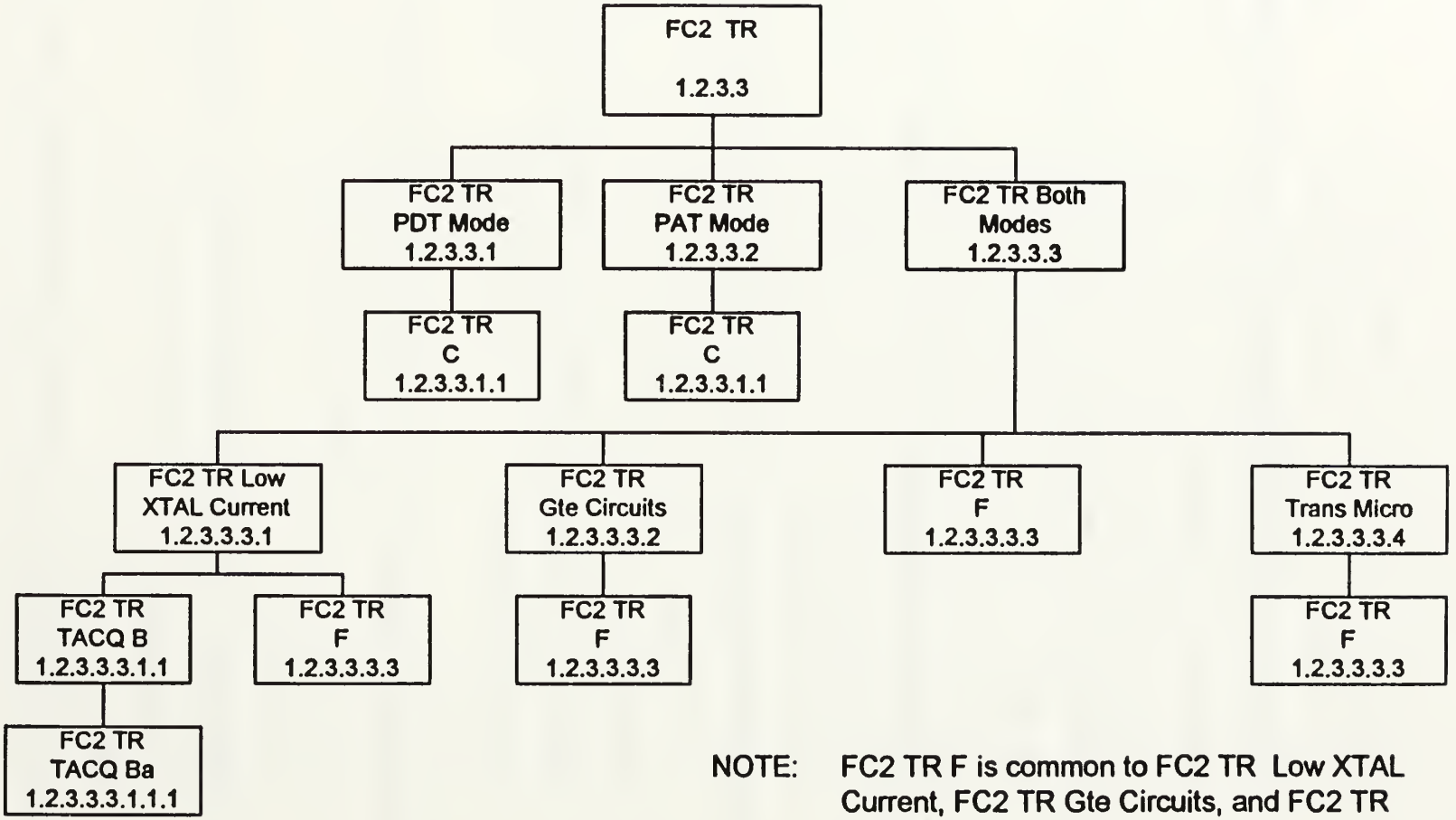
Description: Continues troubleshooting of FC2 TR Trans Micro

Called by: FC2 TR Trans Micro

Calls: FC2 TR C, FC2 TR D

==========

FC2 TR



NOTE: FC2 TR F is common to FC2 TR Low XTAL Current, FC2 TR Gte Circuits, and FC2 TR Trans Micro.

FIGURE 10

FC2 TR F

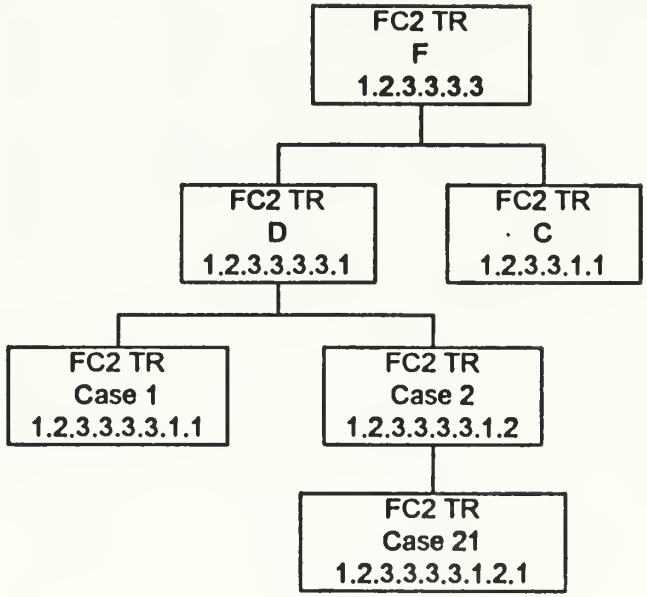


FIGURE 10A

APPENDIX B ADEPT OVERVIEW

Symbologic Adept is an integrated, easy-to-use development environment that combines visual programming and a powerful procedures-based approach to expert systems.

Symbologic Adept was designed expressly to make it easy to model business and technical procedures, and then turn those procedures into interactive software applications. The kinds of procedures that are particularly suited for automation using Adept, include: equipment diagnostic and troubleshooting procedures, scientific procedures, medical and health care procedures, and training procedures.

Adept differs from rules-based systems in that it combines visual development with a procedures-based paradigm. Visual application development means that programmers can build applications by creating and manipulating graphical objects on the screen. Adept's graphical approach facilitates critical thinking and makes it easy to spot gaps in procedures. (Himes and Sperry, 1991, pp.8-11)

Adept has three components that allow a programmer to build and test expert system applications: procedure builder, a set of graphical tools used to build procedures

and link them together, display builder, a set of graphical tools used to customize the display screens that serve as the interface for the application, and runtime, a program used to run and debug developed applications.

A. PROCEDURE BUILDER

Procedure builder is a graphical tool set used to construct the procedural skeleton, or framework, of an application. Each procedural skeleton is made up of smaller units called "nodes". Nodes are graphical objects that represent the various steps in a procedure. Developed nodes are defined by the information that is entered to its immediate right. This information could represent tasks or decisions. Green, "yes", Red, "no", and Blue, "unknown" arcs link the nodes together to indicate a logical sequence of steps. The graphical network of nodes and arcs define a procedure. (Himes and Sperry, 1991, pp.12-13)

B. DISPLAY BUILDER

Adept will automatically create a "default display" each time the application needs to communicate with the user, i.e., when a display node is created. The display is a collection of graphical objects, i.e., buttons, text fields, and list boxes, that elicit information from the user that is needed to complete a procedure, or that presents results and instructions.

Display builder's comprehensive tool set enables the user to manipulate the default display by using a point-and-click approach similar to many drawing programs. This approach makes it possible, through Adept's color line, tool, and shape palettes, to construct unique and functional screen displays. (Himes and Sperry, 1991, pp.14-15)

Specifically, display builder enables the user to:

-- Create standard Microsoft Windows push buttons, radio buttons, and check boxes.

-- Create standard Windows list boxes and text fields.

-- Create graphic shapes and apply colors to them.

-- Import bitmap graphics from other Windows programs.

-- Design a background common to every display in the application.

C. RUNTIME

Adept Runtime contains an "inference engine" that provides the tool's reasoning capability. It decides which procedures to apply to solve a particular problem and then guides a User through those procedures. Adept is able to draw inferences and conclusions as it works through procedures and interacts with a user. By evaluating the statements attached to nodes in procedures, then taking one action or another based on its evaluation, Adept is able to navigate through complex procedures. (Himes and Sperry, 1991, pp.16)

Essentially, creating an application in Adept is a three-stage process that is repeated several times until the application is complete:

-- Create a procedure using Procedure Builder.
-- Customize procedure screens with Display Builder.
-- Test the logic of the procedure using Runtime.

D. PROCEDURAL GUIDELINES

Discussion to this point has focused on building, displaying, and running procedures. Combining developed procedures will produce an application program. It is important to have a firm grasp on the application's design before development begins. The following guidelines should be kept in mind:

-- Create a hierarchy of procedures.
-- Design small and compact procedures.
-- Use the procedures as resources.

1. Create a Hierarchy of Procedures

Adept starts the application at the highest possible level. A main procedure is created and then child procedures follow, at lower levels of detail, that solve individual components of the larger problem.

True top-down designs are not possible in expert system development due to the lack of complete program specifications to guide programming efforts. However, most

applications will have a level from which all other design levels seem to fall. (Prerau, 1990, pp.267)

2. Design Small and Compact Procedures

Clarity and simplicity are important when building expert systems. The number of procedure nodes should be kept to a minimum, the recommended maximum is between 20 and 30 nodes. Adept is capable of handling the maximum number of nodes, but maintaining and verifying a number much larger than 30 will be difficult.

3. Use Procedures as Resources

Adept is very capable in the area of modularity and reusability. For example, if a series of steps are repeated in more than one procedure, create a child procedure that embodies the steps. The child procedure can then be linked to each procedure that uses those steps. Maintaining one common procedure is better than maintaining several separate procedures.

E. SYSTEM REQUIREMENTS

- -- Personal computer using an 80286 or higher processor (386 recommended).
- -- 2MB of memory.
- -- VGA, Super VGA, or monochrome VGA monitor and adapter card.
- -- 5.25 inch high-density (1.2MB) or 3.5 inch high-density (1.44MB) disk drive.

-- Hard disk drive with at least 3 MB free space.

-- Microsoft mouse or compatible pointing device.

-- Microsoft Windows 3.0 or later version.

F. COMMENTS

The information presented in this overview is available in greater detail through Symbologic Corporation or the SoftSell company.

Symbologic Corporation
15379 Northeast 90th street
Redmond, Washington 98052
(206) 881-3938

SoftSell
16150 N.E. 85th, Suite 224
Redmond, Washington 98052
(800) 886-3125

LIST OF REFERENCES

Buchanan, B.G., and Shortliffe, E.H., *Rule-Based Expert Systems, The MYCIN Experiments of the Stanford Heuristic Programming Project*, Addison-Wesley Publishing Company, Menlo Park, California, 1984.

Fersko-Weiss, H., "Symbologic's Adept Builds Expert Systems Graphically," *PC Magazine*, v.10, 31 December 1991.

Himes, A., and Sperry, S., *Using Symbologic Adept*, Version 2.1, Symbologic Corporation, 1991.

Keller, R., *Expert System Technology, Development and Application*, Yourdon Press, Prentice-Hall Inc., New Jersey, 1987.

Naval Ship Weapon Systems, Engineering Station, *Fire Control Technician Training DSOT*, U.S. Navy, various.

Navy Training Plan S-30-7307D, *Fire Control System (FCS) MK 92 MODs 1/2/6*, U.S. Navy, various, 4 March 1991.

OPNAV 4790/83 (Rev. 2-82), *Maintenance Requirement Card (MRC)*, U.S. Navy, July 1992.

Pallatto, J., "Symbologic Ships Windows Version of Expert System," *PC Week*, v.8, 12 August 1991.

Port Hueneme Division, Naval Surface Warfare Center, "System Requirements for the Engineering Development Model (EDM) FCS MK 92 Maintenance Advisor Expert System," 21 August 1992.

Powell, S.H., *Economic Analysis of the Mk 92 MOD 2 Maintenance Advisor Expert System*, Master's Thesis, Naval Postgraduate School, Monterey, California, September 1993.

Prerau, D.S., *Developing and Managing Expert Systems: Proven Techniques for Business and Industry*, Addison-Wesley Publishing Company Inc, Menlo Park, California, 1990.

Smith, C.D., *Development of a Maintenance Advisor Expert System for the MK 92 MOD 2 Fire Control System*, Master's Thesis, Naval Postgraduate School, Monterey, California, September 1993.

Sprague, R.H., and McNurlin, B.C., *Information Systems Management in Practice*, 3d ed., Prentice-Hall Inc., 1993.

Symbologic Corporation, "A Computer's Intuition," *AI Expert*, v.6, December 1991.

Turban, E., *Decision Support and Expert Systems: Management Support Systems, Second Edition,* Macmillan Publishing Company, New York, 1990.

Walters, J.R., and Neilsen, N.R., *Crafting Knowledge-Based Systems: Expert Systems Made Easy Realistic*, John Wiley & Sons, New York, 1988.

BIBLIOGRAPHY

Coffee, P., "Windows-based Tool Adept at Building Expert Systems," *PC Week*, v.8, 26 August 1991.

Dessert, P.E., "Heuristic Development," *AI Expert*, December 1991.

Eisehnstadt, Marc, and Brayshaw, Mike, "A Knowledge Engineering Toolkit," *Byte*, October 1990.

Eisehnstadt, Marc, and Brayshaw, Mike, "A Knowledge Engineering Toolkit part2," *Byte*, November 1990.

Halstead, Rodd, "Develop Advanced Expert Systems," *Byte*, January, 1990.

Kidd, A.L., *Knowledge Acquisition for Expert Systems*, Plenum Press, New York, 1987.

Knaus, R., "Go with the flow. (combining expert systems and flowcharts to knowledge bases)," *AI Expert*, v7,June 1992.

Knaus, R., "Using Existing Knowledge," *AI Expert*, December 1991.

Pallatto, J., "Symbologic Upgrade Helps Novices with Expert-system Development," *PC Week*, v.7, 19 November 1990.

Pedersen, Ken, *Expert Systems Programming,* John Wiley and Sons, New York, 1989.

Sacerdoti, E.D., "Managing Expert-System Development," *AI Expert*, v.6, May 1991.

Summers, Eric, "ES: A Public Domain Expert System," *Byte*, October 1990.

Tzafestas, Spyros G., *Knowledge-Based System Diagnosis, Supervision and Control,* Plenum Press, New York, 1989.

Waterman, Donald A., *A Guide to Expert Systems,* Addison-Wesley Publishing Company, Menlo Park, California, 1986.

Wexelblat, R.L., "On Interface Requirements for Expert Systems," *AI Magazine*, v.10, Fall 1989.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center — 2
   Cameron Station
   Alexandria, VA. 22304-6145

2. Library, Code 52 — 2
   Naval Postgraduate School
   Monterey, CA. 93943-5002

3. Naval Sea Systems Command, Code 62Z — 1
   2531 Jefferson Davis Hwy.
   Washington, D.C. 22242-51603

4. Naval Sea Systems Command, Code 62ZP — 1
   2531 Jefferson Davis Hwy.
   Washington, D.C. 22242-51603
   Attn: ED McGill

5. Naval Sea Systems Command, Code 62ZPG — 1
   2531 Jefferson Davis Hwy.
   Washington, D.C. 22242-51603
   Attn: FCC Stein

6. Commander, Code 4W32 — 3
   Naval Surface Warfare Center, Port Hueneme Division
   4363 Missile Way
   Port Hueneme, CA. 93043-4307
   Attn: Henry Seto

7. Professor Magdi Kamel, Code AS/Ka — 2
   Naval Postgraduate School
   Monterey, CA. 93943-5000

8. Professor Martin J. McCaffrey, Code AS/Mf — 2
   Naval Postgraduate School
   Monterey, CA. 93943-5000

9. LCDR Clinton D. Lewis, USN 1
HELANTISUBRON (L) Four Five
Box 357128
San Diego, CA. 92135-7128