

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

2

# NAVAL POSTGRADUATE SCHOOL

Monterey, California

AD-A161 254



DTIC  
ELECTE  
NOV 19 1985  
S B D

## THESIS

AN EVALUATION AND IMPLEMENTATION STUDY OF  
THE ZENITH 120 MICROCOMPUTER IN WEST  
COAST FLEET COMMANDS

by

George A. Barnett

September 1985

Thesis Co-advisors:

Daniel R. Dolk

Paul Fischbeck

DTIC FILE COPY

Approved for public release; distribution is unlimited

85 11 15 021

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
	10-1101	254
4. TITLE (and Subtitle) An Evaluation and Implementation Study of the Zenith 120 Microcomputer in West Coast Fleet Commands		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis September 1985
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) George A. Barnett		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943-5100		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943-5100		12. REPORT DATE September 1985
		13. NUMBER OF PAGES 101
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report)  Approved for public release; distribution is unlimited		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)  Microcomputer; Zenith 120; Relational Database; Data Dictionary		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)  Austere funding and the increased administrative burdens being placed on Fleet units will ultimately take their toll on performance, morale, and combat readiness. Applying low cost microcomputer technology presents one method to alleviate many of these burdens. However, the current implementation process being used is <u>not</u> sufficient to meet the needs of the Fleet. Empirical evidence indicates that expenditures on		

DD FORM 1473  
1 JAN 73EDITION OF 1 NOV 65 IS OBSOLETE  
S/N 0102-LF-014-6601

1

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

#20 - ABSTRACT - CONTINUED

computers are not accompanied by the expected rise in productivity.

The use of microcomputers is feasible. They are being used successfully throughout both the civilian and military communities. This study discusses the environment necessary to meet user needs. The central theme is one of standardization and documentation. Necessary tools and general recommendations are discussed.

Accession For	
DTIC	<input checked="" type="checkbox"/>
NTIS	<input type="checkbox"/>
Other	<input type="checkbox"/>
Availability Codes	
Dist	Special
A-1	



Approved for public release; distribution is unlimited.

An Evaluation and Implementation Study of the Zenith 120  
Microcomputer in West Coast Fleet Commands

by

George A. Barnett  
Lieutenant Commander, United States Navy  
B.A., University of South Carolina, 1972

Submitted in partial fulfillment of the  
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

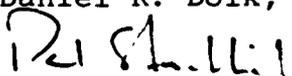
NAVAL POSTGRADUATE SCHOOL  
September 1985

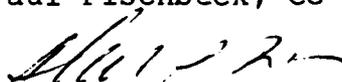
Author:

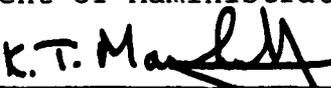
  
\_\_\_\_\_  
George A. Barnett

Approved by:

  
\_\_\_\_\_  
Daniel R. Dolk, Thesis Advisor

  
\_\_\_\_\_  
Paul Fischbeck, Co-Advisor

  
\_\_\_\_\_  
Willis R. Greer, Jr., Chairman,  
Department of Administrative Sciences

  
\_\_\_\_\_  
Kneale T. Marshall,  
Dean of Information and Policy Sciences

## ABSTRACT

Austere funding and the increased administrative burdens being placed on Fleet units will ultimately take their toll on performance, morale, and combat readiness. Applying low cost microcomputer technology presents one method to alleviate many of these burdens. However, the current implementation process being used is not sufficient to meet the needs of the Fleet. Empirical evidence indicates that expenditures on computers are not accompanied by the expected rise in productivity.

The use of microcomputers is feasible. They are being used successfully throughout both the civilian and military communities. This study discusses the environment necessary to meet user needs. The central theme is one of standardization and documentation. Necessary tools and general recommendations are discussed.

TABLE OF CONTENTS

I.	INTRODUCTION -----	7
	A. BACKGROUND -----	7
	B. REQUIREMENT ANALYSIS -----	9
	C. SOFTWARE DESIGN -----	10
	D. APPLICATION -----	11
II.	SYSTEMS DESIGN PROCESS -----	13
	A. SYSTEM NEED -----	15
	B. SPECIFICATION AND FEASIBILITY -----	15
	C. COMMUNICATION -----	16
	D. IMPLEMENTATION -----	18
	E. APPLICATION -----	20
	F. SECURITY -----	21
	G. SAFETY -----	21
	H. EVALUATION -----	22
III.	SOFTWARE DESIGN -----	25
	A. ENVIRONMENT -----	25
	B. LANGUAGE -----	26
	C. SELECTING SOFTWARE -----	27
	D. DESIGN GOALS -----	30
	E. DIALOGUE -----	32
	F. DESIGN PRINCIPLES -----	34
IV.	DATA DICTIONARY/DIRECTORY SYSTEM -----	43
	A. FUNCTION -----	43
	B. USERS -----	46

C.	CLASSIFICATION -----	47
D.	SOFTWARE DATA DICTIONARY/DIRECTORY -----	48
E.	SUMMARY -----	49
V.	SQUADRON MAINTENANCE/MANAGEMENT PROGRAM -----	51
A.	REQUIREMENTS SPECIFICATION -----	51
B..	DATABASE DESIGN -----	53
C.	PROGRAMMING -----	55
VI.	CONCLUSIONS AND RECOMMENDATIONS -----	59
A.	CONCLUSIONS -----	59
B.	RECOMMENDATIONS -----	59
	APPENDIX A: REQUIREMENTS ANALYSIS SURVEY -----	63
	APPENDIX B: SOFTWARE DATA DICTIONARY -----	66
	APPENDIX C: SQUADRON MAINTENANCE MANAGEMENT PROGRAM --	73
	LIST OF REFERENCES -----	98
	INITIAL DISTRIBUTION LIST -----	100

## I. INTRODUCTION

### A. BACKGROUND

In the last decade, the Navy has acquired a myriad of microcomputer equipment. Because of this variety, these acquisitions have made it increasingly difficult to maintain any standard regarding usage and software development. On the other hand, the use of microcomputers has the potential to increase productivity. For example, at the Naval Air Development Center, Warminster, researchers recently completed testing on a computer assisted preflight mission planning software package for the P-3C aircraft. This application of desk-top computers may produce a significant reduction in fuel consumption, reduce repetitious preflight mission planning, thereby provide aircrews more time for tactical considerations. Additionally, at the Navy Personnel Research and Development Center, San Diego, Eleanor Robinson and others have tested the effectiveness of using microcomputers for computerized data entry systems to improve the efficiency and reduce errors in Navy personnel records [Ref. 1]. In the Navy today, there are no less than 273 studies, either completed or in progress that employ microcomputer technology.

Modern microcomputer systems are substantially more reliable, have expanded memory, and cost considerably less

than their predecessors. They are no longer simply toys for teenage hackers, but are now being used successfully by millions of individuals and small businesses for a variety of applications. There are estimated to be well over 2 million desk top computers in the U.S. today, or at least one for every twenty desks. By the end of this decade, most predictions estimate this will become one in three [Ref. 2:p. 48].

Many young officer and enlisted personnel have substantial experience with microcomputer hardware and software. They have been exposed to the microcomputer since high school or college and are well versed in its capabilities. It should be a Navy goal to capitalize on this expertise to the fullest extent. There is no doubt that microcomputers will continue to influence every aspect of our foreseeable future.

This thesis is concerned with increasing the productive use of personal computers at the Navy aviation squadron level. A questionnaire is created to ascertain what user needs exist in the PC realm. Techniques for developing and using PC software are discussed. The use of a data dictionary/directory system, (DD/DS) for documenting information resources and as a tool for disseminating information about existing PC software is discussed. A sample program is written in dBASE II to illustrate these techniques.

## B. REQUIREMENTS ANALYSIS

Decision makers in all organizations face an increasing variety of potential information sources. However, choosing the correct system for generating the most useful information is a formidable task. Without question, the microcomputer provides an opportunity for organizations to access and analyze much more data than previously available.

As a result of a massive microcomputer contract with the Zenith Data Systems, Commander Naval Airforce Pacific (CNAF), has supplied hundreds of individual West Coast commands with the Zenith 120 microcomputer. This study investigates some of the problems associated with supplying microcomputer technology to Fleet units.

The Z-120 is an excellent example of the modern microcomputer. Nevertheless, providing an individual command with a valuable tool without adequate training and direction for its use is counterproductive. The use of microcomputers must be controlled and not allowed to proceed in an ad hoc fashion. The following criteria must be considered in microcomputer acquisition:

- user needs
- system feasibility
- system design
- implementation requirements
- evaluation criteria
- security measures

The supply of available software for information processing has continued to grow. More and more it has become advantageous to seek out existing programs before needlessly "reinventing the wheel." However, selecting software must be done with extreme care. The cost in manhours spent needlessly programming and reprogramming due to poor system structure or insufficient capabilities can quickly offset any advantage gained from the computer. Computer programmers long ago developed organizations and standard procedures for sharing software. We are being remiss expending numerous manhours on software that has not been documented or tested.

Chapter II discusses the issues of systems design for microcomputer hardware and software as they relate to the Zenith 120 purchased by the Navy. In particular, areas cited in the requirements analysis survey are discussed in detail.

#### C. SOFTWARE DESIGN

Although many programs such as dBASE II, Lotus, and Wordstar are available to Fleet personnel, there is a tremendous amount of variability in its use.

Software is much more than a mere collection of programs written in some programming language. Actually, the entire gamut of nonelectronic support for the computer is classified as software. When we talk of software, we mean the computer programs, the instructions for their use, and the

necessary design documentation. There are three different kinds of software:

1. Application programs. Software that is especially written to solve some particular problem.
2. System programs. Software that usually comes with the computer and is designed to make creation of application programs easier.
3. Documentation. Software that shows us how to use the programs and how to modify them for special needs. Good programs are characterized by good documentation. Furthermore, good documentation is written before the program itself.

Chapter III provides some techniques and numerous examples to Z-120 users that will help in designing easy to use programs when applying commercial software.

#### D. APPLICATION

Application refers to capitalizing on the computer's speed and accuracy automating some operation currently being done manually. The evolution of computer technology has been the basis of the so-called "age of information" or "second industrial revolution." Of particular note in this expanding technology is the advancement of the microcomputer. Some modern microcomputers can execute 750,000 instructions per second, store four million characters in RAM, and access data in less than 250 thousandths of a second [Ref. 3: p. 65]. They interface with CRTs, printers, and plotters as well as with other microcomputers. As users become increasingly aware of the capabilities of the computer the opportunity to increase their own decision making ability will prove much greater.

The data dictionary/directory system (DD/DS) is presented in Chapter IV as a software system which supports documentation. A DD/DS can be considered as a source of information about the definition, structure, and use of data in contrast to the actual data values themselves. The DD/DS contains the name of each type of data, synonyms, definitions, constraints, programs where it is used, and its relationship to other data. This concept is discussed in detail and an example dictionary using available application programs is provided.

Chapter V is an example of an application program for squadron maintenance managers written in dBASE II. The Squadron Maintenance/Management System is an interactive program capable of assisting managers with decision making by answering such queries as "How many aircraft will be down in the next seven days for special inspections?" The program as written is generic and would need to be expanded to meet the requirements of a particular aircraft squadron.

Chapter VI presents the conclusions and recommends additional steps to assist in the implementation of future microcomputer systems.

## II. SYSTEMS DESIGN PROCESS

The Zenith system purchased by the Navy represents state of the art design which is capable of satisfying many business and personal needs. It can act as a stand alone tool or function as the center of a powerful automation package. Combined with suitable software, it provides practical and affordable solutions to a variety of applications including data processing, telecommunications, networking, and numerical analysis to aid important managerial decisions [Ref. 12].

The checklist at the end of this chapter is designed to help managers ensure that the system they choose will meet the needs of the user. The use of the guidelines provided will facilitate analysis of the existing problems. Requirements are not static, but should reflect research, experience, and technological changes.

Before purchasing a versatile system such as the Z-120, managers must consider user needs and establish requirements for what the system is supposed to do. As a minimum, the following considerations must be addressed:

- Need for a new system
- Feasibility assessment
- System design
- Implementation

- Evaluation
- Security

In cooperation with the Navy Regional Data Automation Center, San Diego, CA, a questionnaire was distributed to over 60 users of the Zenith 120 system purchased by the Navy. They were asked to respond to questions concerning application, implementation, and training requirements. In the area of application, the survey revealed that no mechanism exists for control of application software. With regard to implementation, the survey found that a squadron seldom asked for the computer nor did they have any idea as to what it could do for them. Additionally, once the computer was received, the survey revealed that squadrons lacked the necessary training to make it useful. Users were left to decide where, when, and how the system was to be used. Appendix A provides a copy of that questionnaire and a summary of results.

The following quote from a typical user indicates the feelings of many;

I don't feel the Z-120 systems are being utilized to their full capacity in most Fleet squadrons. You would find in the majority of commands that the computer is just another word processor, and not used as a computer. The Navy should have developed specific application software for Navy wide use; flight time, maintenance, personnel, etc., so that the system could be functional. As it is now, unless someone has the time to devote to programming or learns to utilize supplied software and make it work for what they want it to do, the Z-120 will be a work processor forever.

## A. SYSTEM NEED

Any system design process begins with the recognition of a need for a system and the establishment of requirements for that system. In most organizations, many different applications may be needed by system users. Because transaction, management, and decision support systems are each resources to the organization, they must be developed according to their value to the organization, and the availability of users to work on them. There must be a well thought out plan for the system, one that tells where it is going, in what order it will be developed, and what resources it will require. Developing and implementing a microcomputer system is no different than any other system and requires that same attention to detail. There are four stages of system development recognized: specification, feasibility, design, and implementation [Ref. 4:pp. 358-397].

## B. SPECIFICATION AND FEASIBILITY

Specification of system need arises either by suggestions from users or by events that are taking place. Some questions that must be answered in the specification phase of the microcomputer acquisition include:

- What should the new application accomplish?
- What reports do we need to produce?
- Who really wants this project?
- What does the data flow require?
- What programs will we need?
- How much time do we have?

These questions are typical of what managers and users must answer prior to evaluating the feasibility of a new system. Unfortunately, many managers do not properly address these issues nor do they listen to what users are telling them. Specification involves an interaction between the system developers and user-managers to clarify exactly what the system requirements will be.

A feasibility evaluation must be accomplished before a system acquisition is approved. That is, assurance is required that the system can be built within reasonable technical, economic, and operational constraints. The technical question concerns the availability of hardware, software, and the expertise required to develop a system that will respond to user needs. Economic feasibility means weighing the benefits gained from the use of the system against their costs. That is, will it cost more to design, install, train personnel, and use the system than it is worth? Operational constraints refer to the effect the system will have on the people who will be associated with it. If a new application will not be used by its intended users, then there is no point in continuing to develop it.

#### C. COMMUNICATION

A serious problem with computer systems development is the inability of the user to specify his requirements for the system due to a lack of knowledge about the system's

capabilities. This problem is aggravated in microcomputers because of the speed of the technical advancements and the increasing variety of software systems becoming available.

On the other hand, the system designer is often not a functional expert and has difficulty interpreting user needs. Compounding this problem is the complexity and interrelatedness of many user systems. Because of the substantial technical hurdles that must be overcome, the design process often becomes centered on building a system that performs the technical functions and ignores the human considerations. In practice, however, after the technical problems are overcome, human factors are the driving force in establishing the usefulness and productivity of the system [Ref. 5].

During the requirements analysis phase of system development, designers must determine and then describe user needs so that the design of hardware and software can occur. A related problem is that the designer may interact only with upper-level managers and not gain a true knowledge of the data requirements.

Since the system is being developed for the user, the requirements must come from them. The user must be interviewed to determine what is to be produced, and they must determine when the specification is complete. The bulk of work in this phase is interviewing users and documenting findings. This can be a sizable task, however, there is no

alternative. Users must be identified in as many distinct ways as they appear and the requirements they represent must be kept uniquely defined and supported throughout the development process.

Question 20 of the survey addresses the subject of user needs. Out of 31 responses, 29 said that they had never been contacted about what they want or need from a micro-computer system. To properly implement and thus receive the maximum benefits of future systems, greater user involvement must be a part of the selection process.

The final aspect of communication is continuity. Communication must be a continuous process that exists well after the system is operational. Channels of communication need to be made available to handle new problems or new requirements as they arise. Systems and users must not become static. As systems evolve, it is essential that users and managers continue to communicate.

NARDAC, Norfolk, Va. publishes a quarterly magazine titled "Chips Ahoy" to fulfill part of this communication need. Additionally, they maintain an electronic bulletin board to share data and program resources [Ref. 6]. It is recommended that NARDAC, San Diego, establish a similar user information publication and telecommunication network.

#### D. IMPLEMENTATION

Although most users are initially willing to give a new system ample opportunity to prove itself worthy, if it

continues to frustrate them the result will be reduced productivity. Some people may even resort to direct action against the system. In a sample of 40 computer installations, Dowling found that 45% had experienced some form of sabotage [Ref. 7:pp. 95-97]. Conversely, positive user attitude enhanced by well planned implementation contributes significantly to system performance and user ability.

During the implementation phase, the training of users in the organization occurs. Users include those who interact directly with the system, those who will be supplying data to the system, and any others who will be required to use the reports and documents it may generate. All these individuals need to be aware of the importance of their role and the way they are affected by the system. As a rule, the user should be able to learn the system by actually working with it. The following training items should be considered as a minimum:

- computer assisted instruction where possible
- programs especially designed for beginners
- review programs for infrequent users
- programs for experienced users
- minimum competency standards before working on data
- a non-technical description of the system
- a logical description of how the system can aid managers
- ready access to reference material

## E. APPLICATION

During this study, one question heard time and again from many users of the Z-120 is, "What do we use it for?" The average hours per week of usage for all commands on the Z-120 was 9.1. The constraint with usage is not the capability of the system, but the lack of available programs applicable to Fleet squadrons. Supplying squadrons with dBase II, Lotus, etc., is useful but not sufficient. What users need now are programs that are written for particular applications.

Additionally, many users of the Zenith system indicated in the survey that they are faced with varying degrees of resistance by senior officers and petty officers. This reaction to a new computer system is not uncommon in many organizations. To reduce the resistance generated by the system, command attention must be given to implementation. All personnel should be encouraged to participate and use the system. Open channels of communication between users and managers must be established and maintained. Finally, a commitment of squadron resources to support the system should be accomplished by incorporating the system into the squadron's formal training plans.

Users of the Z-120 surveyed noted several deficiencies that they would like to see corrected. First, they desire a letter quality printer capable of producing OCR messages. Second, they would like to have a telecommunications

capability via a modem access for intra-squadron sharing of software. Finally, increased training is essential if all users are to take full advantage of the system.

#### F. SECURITY

The key to security is personnel who are trustworthy and always aware of security issues. Ignorance, apathy, and frustration are, however, just as dangerous as deliberate dishonesty. Security methods should not be intricate or burdensome. Nevertheless, they must be complete and thorough. Security measures must be a part of any initial training and periodic reminders concerning security provided to all personnel.

If processing classified data, users must ensure that TEMPEST systems are utilized, that the computer is turned off after each use, and that disks containing classified data are treated appropriately. Users of the Z-120 have access to enormous amounts of classified material and must ensure that they are complying with appropriate instructions. OPNAVINST 5239.1A addresses ADP security issues, additionally, Ch-1 of 1 April 85, is applicable to all DON activities.

#### G. SAFETY

The National Institute for Occupational Safety and Health has stated that CRTs do not present a radiation hazard to the user either working on or near the terminal.

"There is no known way that CRTs can cause visual defects of the eye, increase the optical components or change the strength of and pliability of ciliary muscles" [Ref. 5: pp. 2-6]. It is recommended that after two hours of continuous CRT work that the user switch to some other activity for at least 10-15 minutes to rest the eyes and counter fatigue. Users should ensure that the screen is clear, stable, and free of glare, and that the screen angle, keyboard height and user's chair are easily adjustable. The user's work station needs to be designed based on human factors and not be the result of haphazard acquisition and placement of equipment.

#### H. EVALUATION

As the implementation process is being accomplished, an evaluation and monitoring scheme should be initiated. No matter how good a job one does of initially selecting a new system and implementing it, the unexpected will occur. Only by carefully monitoring the process can timely response result.

Once begun, the implementation process is evolutionary and continuous. Of course, time, cost, and manpower may not allow for all of the implementation processes discussed here. However, the more thorough and well defined the process, the more productive the system will be. Fleet commands must continue to incorporate new ideas and methods.

Supporting the use of microcomputer technology can significantly aid in this task.

## SYSTEMS DESIGN CHECKLIST

	Yes	No*	N/A
1. Have the potential users been identified?	___	___	___
2. Have potential users been contacted for requirements?			
a. Reports, forms, graphics?	___	___	___
b. Application programs required?	___	___	___
c. System programs required?	___	___	___
3. Have problems with the current system been identified? Can they be remedied?	___	___	___
4. Has the new system been examined in its entirety from data sources, input, output, and manipulation?	___	___	___
5. Has formal training been designed?	___	___	___
6. Have behavioral goals been established?	___	___	___
7. Is there a training program for each level of user?	___	___	___
8. Have users been provided with a brief nontechnical description?	___	___	___
9. Have mechanisms been provided for users to make suggestions or complaints?	___	___	___
10. Do these mechanisms provide feedback to the users?	___	___	___
11. Is there a formal evaluation established to determine system performance?	___	___	___
12. Are security measures taken to preclude unauthorized access to terminals and disks?	___	___	___
13. Have working spaces, equipment, etc., been identified or made available?	___	___	___

\* Answers in this column require action or some justification for departing from established guidelines.

### III. SOFTWARE DESIGN

Although microcomputers were introduced in the mid-1970s, they were not available for general use due to a lack of software. In 1978, a financial software package for microcomputers was marketed for under \$100.00. It was so popular that it sold 100,000 copies in just 18 months [Ref. 9:p. 22]. Despite the recent decline in microcomputer sales, they continue to represent a raw computing capacity that is just now being blessed with an abundance of quality programs making them easier and easier to use.

The checklist provided at the end of the chapter is intended to serve as a starting point for program design. The programmer must be knowledgeable in all aspects of the system to ensure that the end user's needs are satisfied. Only through experience and practice can this become a rewarding endeavor to the programmer.

#### A. ENVIRONMENT

It is important to note the differences among kinds of computers and the various environments in which software exists. There are three kinds of computers; mainframes, minis, and micros. The concepts of programming and software design are the same, regardless of the environment. Nevertheless, effective design requires an awareness of the various systems.

Large-scale computers are generally designed to handle problems that require massive computation or access to large files. The centralization of data files for an organization requires a single computer complex, and many of the problems that require high-speed computational ability need that capability for only a few seconds. A large scale computer allows many users who have similar problems to share a single resource, the central processing unit (CPU). The minicomputer was introduced as a relative low-cost component (priced around \$25,000 when mainframes were over \$250,000) and is simply a smaller version of a mainframe [Ref. 10: p. 3]. Because of the cost of the CPU and the economies of scale, it was practical to centralize operations by using the CPU as much as possible. Now that a microcomputer CPU is one of the least expensive components of a system, it is often practical to decentralize with many microcomputers as dedicated on-site processors.

#### B. LANGUAGE

Many different programming languages have been developed, ranging from low-level machine code to assembly languages and higher-level languages. Each language has certain features that make it useful for specific kinds of applications. There also exist program-generators that will write programs after stepping through a menu of selected questions. Regardless, complex programs when written in

any language will require extensive cost, documentation, maintenance, and training to use. The ability to write easy to use application software will give the user a wide degree of flexibility. It is strongly recommended that we stress in both our officer and enlisted ranks as much software training in high school and college as possible. Of those Zenith 120 users surveyed, nearly all responded that they would like to attend in depth courses in various programming languages.

### C. SELECTING SOFTWARE

Today there are literally thousands of software packages available from which users may choose. Many of the micro-computer programs available to users of the Z-120 system are inexpensive and easy to learn. However, selecting software for the system must be done with as much care as selecting and designing the system. The additional man-hour expense in operator time due to poor program structure and the cost of converting from the old system to a more contemporary package can be devastating. Software acquisition should consider the following requirements and features:

- user training
- help commands
- compatibility with hardware/operating system
- expansion capabilities
- user interaction

- language code
- cost
- ease of maintenance
- vendor reputation
- memory space
- error detection
- documentation

Most commercially available software is easy to use, even by those who have never used a computer. Today's software is written with the assumption that the user will not have a background in software design and that most users will never bother to read the manual. For the most popular software, there are commercially published texts that summarize the program's main features and allow the user to follow the 80/20 rule. That is, 80% of the users will only need about 20% of the instructions and commands found in their user manual.

NARDAC, San Diego does an excellent job of providing an introduction to several of the software programs made available with the Z-120. They have supplemented the user manuals with their own program text material complete with illustrations and example problems. The instructors are well qualified and professional. Many users of the Z-120 surveyed rated the course as the best they had attended. Many students have requested that indepth courses in specific subjects such as dBASE, Lotus, and Worstar be made

available which would allow them sufficient time to develop professional programming skills. NARDAC, San Diego is in the process of establishing several such courses.

The software packages provided to users of the Z-120 include Wordstar, Lotus 1,2,3, and dBASE II. Each represent state of the art material and have many of the desirable features required by its users. dBASE II is a software system that has found wide acceptance among many professional programmers. Unfortunately, most of the users of our Z-120s are not professional programmers and few have had limited, if any, previous programming experience. Additionally, not all the features of dBASE II are easy to use, nor do they work as expected. Moreover, it is not always advisable to use certain features of the system. Why then did we buy it?

Despite its faults and the varying degree of difficulty some users may have implementing dBASE II, it does contain many standard operations and several features not ordinarily associated with languages such as BASIC and COBOL. These features lie mostly in the areas of data storage, manipulation, and retrieval. Its advantages in these three areas make dBASE II very useful in the development of professional application software systems. Another major advantage is that dBASE II is a self-contained programming system. This means that all the facilities necessary for software development are available within the system itself [Ref. 11].

Thus, software whether strictly off-the-shelf like Wordstar or a programmable language like dBASE II should be designed to minimize the operator task. It should permit logical task sequences with a minimum of control actions, and should be made as simple as possible for tasks requiring real-time responses. The following discussion provides guidance to assist in developing application programs that promote fast, accurate data entry when using a language such as dBASE II.

#### D. DESIGN GOALS

Users often understand their system better than designers, and designers often have a considerable amount to learn before being able to construct a successful application system. To narrow this gap, there exist four goals that application system designers should strive to attain.

##### 1. Ease of Use

First, the application system should be as easy to use as you can make it. The user should be able to select appropriate functions from a menu, rather than be forced to use an instruction manual. All user instructions should be given in nontechnical terms. This does not imply there should be no documentation for the system, on the contrary, the first line of documentation is in the menu itself. When possible, attempt to use menus that allow the selection of functions that are built into the system [Ref. 11:pp.4-7].

## 2. Flexibility

Second, the design of the application system should be as flexible as possible to accommodate changes as they occur. Flexible must not be construed to mean kluge (sloppy) programming, only that the designer allow for as much generality as possible.

## 3. Efficiency

Third, because of the limitations of speed within microcomputers, it is necessary to avoid any practices that will cause unnecessary delay in the application program. Operations that take several seconds should be avoided and substituted with faster techniques when available.

## 4. Accuracy

Last, and of course not least, the application system must work correctly. Decentralization has given rise to thousands of pseudo-programmers. Simply because an answer to a question or problem appears on the CRT, does not make the answer correct. Application programs must be validated with manual techniques and then tested and retested after every modification to ensure correctness. The use of good design procedures will help alleviate this problem.

An excellent source on modern analysis and design techniques that leads to flexible design is a book by Tom DeMarco titled Structured Analysis and System Specification, Prentice-Hall, 1979.

## E. DIALOGUE

Dialogue refers to how the operator communicates with the system. There are four basic types of dialogues: computer initiated and guided, form filling, menu selection, and programming-like statements. The advantages of the computer initiated and guided format are that the computer tells the user what to do for each entry, and the user needs little training. The disadvantages include slow response time and reduced flexibility.

\*\*\*\*\*

Example of computer initiated dialogue:

Type in your name \_\_\_\_\_ <press return>

\*\*\*\*\*

The form-filling format also requires little training, but is not at all flexible.

\*\*\*\*\*

Example of form filling dialogue:

Part-Number\_\_\_\_\_<return> Shelf\_\_\_\_\_<return>

Name\_\_\_\_\_<return> Price\_\_\_\_\_<return>

\*\*\*\*\*

By far the most popular method in use today is menu selection. It requires little training and is designed so that the user has only to recognize the correct option. However, it is limited in scope, and does not make the most

efficient use of the system capabilities. Additionally, it is probably too slow for the experienced user.

\*\*\*\*\*

Example of menu dialogue:

Select one of the choices below.

1. Add Flight Time
2. Update a Record
3. Create a New Record
4. Quit

\*\*\*\*\*

The format that is the most concise, flexible, and powerful is the dialogue using programming language statements from a restricted list of commands. Unfortunately, these advantages are offset by the training required for the operator to learn both the language commands and the correct command format.

\*\*\*\*\*

Example of command dialogue in dBASE II:

JOIN to <file> FOR <exp> [FIELDS <field list>]

In this command, when two database files are open, they can be joined to create a third database. The <exp> indicates which records are to be selected from the opened databases, and the FIELDS clause indicates which fields are to be included in the third database. E.g.,

JOIN to TEMP1 for P.BUNO = S.BUNO

In this example two database files are open, PRIMARY and

SECONDARY. The JOIN command will concatenate common records (BUNO) from two files, and will place the concatenated records into a third database file, TEMP1, for easy access.

\*\*\*\*\*

## F. DESIGN PRINCIPLES

### 1. Diagnostics

If a failure occurs in the program, the system should be designed to allow the user to make corrections or restart the system as necessary. He should be able to distinguish among program errors, equipment failures, and operator errors. Also, the program should allow for orderly shutdown and the establishment of check points so that restorations without loss of computing are available.

### 2. Documentation

Documentation should accompany individual groups of data, modules, and messages to designate their content. Documentation should be located in a consistent fashion adjacent to the data group or message it describes. The relationship of the documentation to the message should be clearly visible. It should be highlighted or otherwise emphasized to facilitate user scanning and recognition. The technique used should be easily distinguished from that used to highlight code or critical messages. Finally, documentation should be clearly worded to avoid confusion about whether it is for a data entry field, a control option, a guidance message, or for other displayed purposes.

### 3. Data Display Format

Display formats should be designed to provide optimum transfer of information to the user. Although there should be a standard format for data texts and tables, the user should be able to modify these formats for personal use, so long as it does not affect other users or the results of the data. Normally, information should be displayed statically on the screen and not scrolled. If text is intended to be scanned, 35 characters per line columns (like in the newspaper) should be used. If text is to be read in detail, efficiency is increased if a 70-character line is used.

### 4. Grouping

Information should be placed in groups to permit the user to associate or compare like classes of data. It is best if columns are either left or right justified to establish boundaries of group areas. Spacing between groups and within groups to maintain information relationships is critical. Place items of greatest importance near the top or to the right of the screen whenever possible.

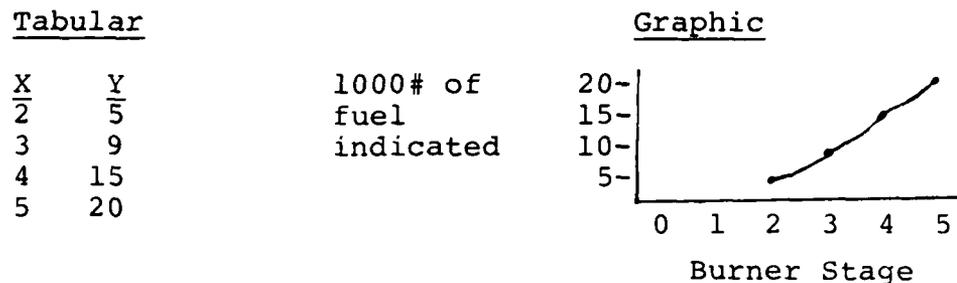
### 5. Data Presentation

Data should be presented to the operator in a readily usable format. Requiring the user to transpose, compute, interpolate, or mentally translate into other units or numerical bases should be avoided. The following are the minimum guidelines that will prove effective when designing programs for the Z-120.

- When presented in tabular form, alphanumeric data should be left justified, numeric data should be left justified by decimal point. E.g.,

<u>Alphanumeric</u>	<u>Numeric</u>	<u>Decimal</u>
A67 890 123	1234	46.24
A67 890 124	45	7.0
A67 890 125	456	2.3
A67 890 126	7	6.333

- Use graphic or tabular form for lists that must be scanned and compared to other items in a list. Ensure that the axes of all graphs are labeled. E.g.,



- The use of hyphens should be avoided.
- Periods should be placed after each item selection number, at the end of a sequence, and where necessary for clarification.
- Text should be displayed in both upper and lower case font.
- Text paragraphs should be separated by at least one open line.
- Frequently appearing commands should appear in the same place on the screen. When words are deleted from lists, consider not compressing the list, but leaving blanks instead so the user will find the command at the same place on the screen.
- When grouping alphabetic characters, make use of meaningful acronyms or short character strings rather than randomly selected characters that have little system relevance.

Good	Poor
Valve 3261	L3XS 3261
Hose 4763	Hse 4763

- The system should round output to the last significant digit and not truncate it. E.g., computed 7.988

Good	Poor
rounded to 7.99	truncated to 7.98

#### 6. Cursor

The cursor should be positioned to the first character position of the first input field. As an input field is completed, the cursor automatically goes to the first character of the next field. Formats should be organized to minimize positioning movements of the cursor to specify where the keyed entry is to go. Align entries to minimize search time.

#### 7. Fixed Length Entries

If a fixed length word or collection of characters is to be entered, underscore should be used to indicate to the user the exact number of characters to enter. For example: Enter Julian Date \_ \_ \_ \_.

#### 8. Menus

The displayed menu should only contain options which are available to the user. If menu items are brief, they may be displayed in two columns.

#### 9. Information Density

The quantity of information displayed on the screen should be held to a minimum. Putting too much information on the screen leads to confusion and increases error rate. The screen should contain only data elements which the user may need. Even then, the screen may appear congested. A

common practice is to designate certain areas of the screen for certain types of information. Users should be able to temporarily remove data not needed from the screen. The screen should not normally be broken into many smaller windows, unless the user desires this option and is able to select it himself.

#### 10. Response Time

Time intervals between input and response tend to be dead time for the user. Decreases in mental efficiency occur in sudden drops rather than in linear fashion when delays exceed a given point. In general, delays greater than 15 seconds rule out conversational interaction. If delays of more than 15 seconds occur, the system should free the user from waiting for a response so that they can turn to other activities.

Response times that are between 5 and 15 seconds are generally too large for interactive conversation since it requires the user to retain information in short-term memory. Users subject to such delays tend to be inhibited in problem solving activity and frustrated in data entry. An operator is usually content to wait for a response for a period of up to 15 seconds after keying RETURN. Longer waiting periods, however, can lead to unproductive behavior or a shift from one task to another [Ref. 5].

SOFTWARE DESIGN CHECKLIST

	Yes	No	N/A
Language			
1. Are command words congruent; i.e., if U means up, does D mean down?	—	—	—
2. Is the language use logical? Consistent?	—	—	—
3. Are labels and messages; a. Distinct? b. Meaningful?	—	—	—
4. Is terminology that of the user and not the designer?	—	—	—
5. Is the user able to request help at any time and not lose data entered?	—	—	—
6. Was the language reviewed by at least a sample of the users?	—	—	—
7. Is nomenclature the same for similar or identical functions in all modes?	—	—	—
Memorization			
8. Is memorization of codes, sequences, etc., minimized?	—	—	—
9. Are codes designed to aid human memory?	—	—	—
10. Is it clear when the computer is waiting for a response or command?	—	—	—
Symbology			
11. Are the symbols standardized: a. Within the system? b. Among similar operations?	—	—	—
12. Are abbreviations, acronyms, and codes in plain, concise text?	—	—	—
13. Are abbreviations used only when they are significantly shorter?	—	—	—
14. Is each abbreviation unique?	—	—	—
15. Are standard fields used for: a. Time? b. Date? c. Numbers?	—	—	—

- |     |   |       |       |       |
|-----|---|-------|-------|-------|
| 16. | Is the standard displayed when the user is inputing this information?   | _____ | _____ | _____ |
| 17. | Are fields identified so that the user can easily recognize the data category?                                  | _____ | _____ | _____ |
| 18. | Do the numbered menu items start with one?  | _____ | _____ | _____ |
| 19. | Is the system designed so that user changes will not affect other users?  | _____ | _____ | _____ |
| 20. | Is the use of hyphens minimized?  | _____ | _____ | _____ |
| 21. | Is unnecessary punctuation avoided?   | _____ | _____ | _____ |
| 22. | Is information displayed so that the user does not have to note single and double space blanks?                 | _____ | _____ | _____ |
|     | Labels  |       |       |       |
| 23. | Does each data group, message, or frame contain a descriptive title, phrase, or word to designate its contents? | _____ | _____ | _____ |
| 24. | Are labels located next to what they describe?  | _____ | _____ | _____ |
| 25. | Are labels highlighted?   | _____ | _____ | _____ |
| 26. | Are labels unique among themselves?   | _____ | _____ | _____ |
|     | Format Design   |       |       |       |
| 27. | Are there fixed formats for:  |       |       |       |
|     | a. Data?  | _____ | _____ | _____ |
|     | b. Text?  | _____ | _____ | _____ |
|     | c. Tables?  | _____ | _____ | _____ |
| 28. | Can the user personalize the formats?   | _____ | _____ | _____ |
|     | Grouping  |       |       |       |
| 29. | Are like classes of information grouped?  | _____ | _____ | _____ |
| 30. | Are group boundaries clearly defined?   | _____ | _____ | _____ |
| 31. | Are comparable items directly over each other for easy comparison?  | _____ | _____ | _____ |
| 32. | Is each new item started on a new line when enumerating?  | _____ | _____ | _____ |

Data Presentation

- |     |  |     |     |     |
|-----|--|-----|-----|-----|
| 33. | Is data in usable, easy to read form?  | ___ | ___ | ___ |
| 34. | Are groups separated by one or more blank characters?  | ___ | ___ | ___ |
| 35. | Is identical data displayed in consistent, standardized form?  | ___ | ___ | ___ |
| 36. | Is alphanumeric data left justified?   | ___ | ___ | ___ |
| 37. | Are list left justified?   | ___ | ___ | ___ |
| 38. | Are list vertically aligned?   | ___ | ___ | ___ |
| 39. | Are subclassifications indented?   | ___ | ___ | ___ |
| 40. | Are periods placed:  |     |     |     |
|     | a. At the end of a sentence?   | ___ | ___ | ___ |
|     | b. After item selection number?  | ___ | ___ | ___ |
|     | c. Where necessary for clarification?  | ___ | ___ | ___ |
| 41. | Do frequently appearing commands and subcommands appear in the same place on the screen?                           | ___ | ___ | ___ |
| 42. | Are alphabetic acronyms pronounceable and meaningful?  | ___ | ___ | ___ |
| 43. | Is text in upper and lower font?   | ___ | ___ | ___ |
|     | Significant Digits   |     |     |     |
| 44. | Are numbers rounded to the last significant digit?   | ___ | ___ | ___ |
|     | Cursor   |     |     |     |
| 45. | Is the cursor easy to track and find?  | ___ | ___ | ___ |
| 46. | Is the text free from visual interference by the cursor?   | ___ | ___ | ___ |
| 47. | Is the cursor positioned at the first character position of the first input field at the appearance of each frame? | ___ | ___ | ___ |
| 48. | As each input field is completed, does the cursor go to the first character position of the next field?            | ___ | ___ | ___ |
| 49. | Are formats organized to minimize user positioning movements of the cursor?  | ___ | ___ | ___ |

Fixed Length Entries

50. Is the length of fixed length character entries indicated by underscores? \_\_\_\_\_

Coding

51. Are the symbols familiar to the user? \_\_\_\_\_
52. Is the information on the display only that which is necessary? \_\_\_\_\_
53. Is information density at a minimum? \_\_\_\_\_
54. Are certain areas of the display designated for certain types of info? \_\_\_\_\_
55. Are the users able to see the entire page on which they are working? \_\_\_\_\_
56. Is there a scroll or window function? \_\_\_\_\_

Response Time

57. Is error feedback given within 2.0 sec? \_\_\_\_\_
58. Are commands to interrupt an automatic process acknowledged within 2 seconds? \_\_\_\_\_
59. Is a complex command either implemented or is feedback concerning implementation less than 5 seconds? \_\_\_\_\_

Control Functions

60. Does the system dialogue prompt the user with the next step or alternatives rather than just ending? \_\_\_\_\_
61. Are auditory signals used to alert and direct the user's attention to the appropriate visual display? \_\_\_\_\_
62. Can auditory signals be removed? \_\_\_\_\_

#### IV. DATA DICTIONARY/DIRECTORY SYSTEM

There is nothing surprising or unique about a data dictionary/directory system. In fact, when compared to other processing concepts the data dictionary is really simple. A dictionary represents one method to collect, quantify, maintain, and publish information about data. This data about data is termed "meta data." When the management of definitions on which the system is built is automated, the standardization and control of data can be optimized. This effort will result in improved productivity and system reliability.

A data dictionary/directory system provides a mechanism to define and use information about data elements, records, files, and their relationships. Additionally, it is capable of defining other entities such as modules, forms, reports, screens, procedures, programs, and just about anything else that the user can think of which is relevant.

The dictionary at the end of this chapter uses dBASE II. It catalogs some of the existing software available today for squadron use.

##### A. FUNCTIONS

In the past few years, designers have made significant progress in the consistency and vigor with which they have applied modern techniques to the development of data

processing systems. To develop modular, flexible, user-maintainable application systems, designers must follow company guidelines. Today, the typical data processing shop has strict standards for programmers to follow that will ensure their products are consistent and cohesive. These gains in application program design have been primarily in the areas of program and procedure definition. Only recently has an attempt been made to define and document information about data and to concentrate on ways to use it effectively in the stages of system analysis, design, and maintenance.

#### 1. System Analysis

A DD/DS can be effective when used as a tool for system analysis such as requirements documentation, alternate systems comparison, data flow, and communication.

Traditionally, communication has taken the form of conversations, meetings, and telephone calls. This communication has been recorded as memos, minutes, and notes. Occasionally, we have developed manuals and references texts, but the value of this medium was limited in updating capabilities and circulation techniques [Ref. 14:p. 14]. What we have lacked is a central pool of information that can be shared by all departments in an organization. The data dictionary/directory system represents this medium.

#### 2. System Design

The DD/DS can support the system design process as well. It can be used to set standards, improve data

integrity, increase security, and provide for database documentation. As such, it is an efficient way of portraying system design details to the user. The DD/DS is used to generate files, segments, and record definitions for a variety of programming languages. By doing so, it centralizes the control of program data definitions. This aspect of the DD/DS ensures consistency of data use, and reduces data redundancy [Ref. 14:p. 17].

### 3. Program Development and Maintenance

Documentation using a data dictionary/directory is available to anyone in the system with a terminal. The DD/DS represents a dynamic documentation mechanism that keeps pace with the changing system. For example, before converting some element such as zip-code to nine digits, it would be beneficial to identify every occurrence of it in the system. The automated search and cross-referencing tools of a good DD/DS span multiple programs, systems, databases, report definitions, or any other entity type.

The use of a DD/DS will promote the development of standards concerning entity names, usage, and coding. The dictionary portion comprises those data definition generated from the meta data which are applied to various applications. The directory permits application access to meta data without knowledge of their physical locations. The directory thus enhances the versatility of the dictionary and enables the DD/DS to perform the following functions:

- Record Keeping. The DD/DS can document all the data items in one or more application systems. The objective is to determine common data elements and reduce data redundancy. Also, this record keeping will aid in future system maintenance and documentation.
- Cross-Reference. The DD/DS maintains a useful cross-reference between the entities contained in it. A cross-reference is kept between data elements, synonyms, programs, reports, files, records, and users.
- Indexing. Indexing provides a means to relate words to their definitions. They are indexed by key word, record number, or any other field.
- Standards and Control. The Database Administrator (DBA) can exert control over all the data elements by use of the DD/DS. He should institute procedures that review all data items for redundancy and conformance to standards before their implementation in an application system [Ref. 16].

## B. USERS

Analysts and designers have made tremendous progress toward improved database design including use of the end-user staff to assist in the development of automated data processing systems. The DD/DS is one more tool to increase user effectiveness in system development.

While we have made significant gains in user interaction and review during the phases of system development, this process still relies heavily on the limited data processing staff resource. Traditional development schemes view the user as only a reviewer or auditor of the development effort. In reality, the user often does not get involved in active design and development.

### C. CLASSIFICATION

A DD/DS can be classified as either passive or potentially active. A passive DD/DS does not require that any of the processes or system components depend on the DD/DS for their meta data, and there is no interface with any system component. Thus, changes in the DD/DS does not automatically result in corresponding changes in the software containing the appropriate entity descriptions, and the execution of applications programs does not include automatic checking with the DD/DS for correctness.

On the other hand, a potentially active DD/DS can be the source for data descriptions used in processing components such as compilers, assemblers, and database management systems (DBMS). A potentially active DD/DS can produce the meta base for a given program or process, but is not required to do so.

A truly active DD/DS will be the only source of meta data, and programs and processes will be fully dependent on the DD/DS for it. Presently, no truly active DD/DS systems exist.

A second classification of DD/DS is according to its dependence on other software for implementing its functions. In this respect, the DD/DS may be termed dependent or independent. A dependent DD/DS is one that uses existing DBMS facilities to implement the structure and organization of the meta data and its support functions. This type of

DD/DS requires the facilities of the general purpose software system to perform its functions. This type of DD/DS is said to be "in-line" and is an elaboration of an active DD/DS in that the DBMS directory serves as the directory for both the DD/DS and the DBMS.

An independent DD/DS is one that does not require any other general software systems to perform its meta data management functions. As a stand-alone DD/DS it may be either active or passive.

#### D. SOFTWARE DATA DICTIONARY/DIRECTORY EXAMPLE

The proliferation of application software created for use on the Zenith 120 has in itself created a need for some mechanism to control and coordinate its dissemination. A data dictionary represents one method to ensure that the programs being written are made available to users and to control the logical and physical design of the programs. The program should reside at some local clearing house or on an electronic bulletin board available to users. The Software Data Dictionary/Directory Program is presented in Appendix B and is an example of a simple data dictionary system. Its purpose is to assist users in locating existing software for use at the squadron level. Programs held in this DD/DS were submitted by Fleet users and have not been validated.

### 1. Program Design

This program is a passive and independent data dictionary. It uses a menu selection format that presents six options for the user to select, including one to terminate the session (Figure 4.1). There are five database files; MAINTENANCE, OPERATIONS, ADMINISTRATION, SAFETY, and PERSONNEL for the user to select an appropriate subject. As new programs become available they can easily be added to their respective subject area.

#### SOFTWARE DATA DICTIONARY/DIRECTORY SYSTEM

This DD/DS allows you to find programs you may want to use in your squadron. Simply type the number of the corresponding department you need.

1. MAINTENANCE PROGRAMS
2. OPERATIONS PROGRAMS
3. ADMINISTRATION PROGRAMS
4. SAFETY PROGRAMS
5. PERSONNEL PROGRAMS
6. EXIT FROM SYSTEM;

Figure 4.1

#### E. SUMMARY

It is not difficult to believe that data dictionary/directory systems will become increasingly necessary as a means of standardizing and controlling data, not only in the data resource management systems but in many non-DBMS systems.

The benefits derived from a dictionary are proportional to the size of the dictionary itself. A DD/DS that contains information about the data of only one subsystem is of limited value. However, as information about more systems and schemes are added, the data dictionary increases in worth. The more global cross-referencing and inquiries that can be done, the more valuable this information pool becomes.

The main advantage of a dictionary lies not in its ability to store and catalogue information about data, but in its ability to assist in the sundry information processing activities that occur in any organization.

## V. SQUADRON MAINTENANCE/MANAGEMENT SYSTEM

### A. REQUIREMENTS SPECIFICATION

A major tool for squadron maintenance managers is OPNAVINST 4790.2C. This instruction lists the numerous maintenance functions required in organizational level maintenance activities and the persons responsible for ensuring compliance. When properly followed, the instruction significantly improves performance and training of personnel, equipment readiness, safety, and management. However, only through proper communication and planning can it be effective. For example, aircraft inspections that the squadron must perform are preflight, postflight, turn-around, daily, special, conditional, calendar, phased, acceptance, transfer, and inventory.

A key player in this effort is the maintenance/material control officer (M/MCO). He is responsible to the maintenance officer for the overall productive effort and material support of the maintenance department and aircraft. Among his many duties are the maintenance of aircraft logs and records, monthly review of maintenance data, material requirements planning, assigning workload, and ensuring the full capability of the maintenance department. Obviously, the need to maintain accurate data is essential to the M/MCO. His ability to communicate both up and down the

chain of command would be greatly enhanced by a well-organized database system.

The Squadron Maintenance/Management System Program (SM/MS) (Appendix C) was selected as a model for this thesis because it provides a realistic example of an application program suitable for squadron use. Although the SM/MS program is not complete as it now stands, it employs many of the programming techniques discussed in the paper. With little additional programming, it can easily be expanded to include such functions as supply parts inventory, technical training, and deployment scheduling. SM/MS is written in dBASE II. For more description of terms and concepts, the reader is referred to the Ashton-Tate, dBASE II manual.

To the maximum extent possible, the program is designed to allow interactive processing by the user. It assumes that the user has little or no computer experience. Only access to squadron flight records (yellow sheets) is necessary. There is a significant degree of error checking and automatic processing to ensure minimal data handling by the user.

The system has four basic operations:

1. "Query Option"--An interactive request procedure to extract information about a particular database entity.
2. "Report Option"--A request procedure that will supply the user with some arithmetically computed result.
3. "Update Option"--An interactive procedure for inputting data or making an update to existing database

entities. Relationships across database files are automatically handled.

4. "DD/DS"--The Data Dictionary/Directory System provides the user access to the meaning of any term, module, element, program, or relationship used in the program.

## B. DATABASE DESIGN

Database management systems (DBMSs) are more adequately suited to meet the challenge of complex Navy administrative requirements because a DBMS is data-oriented rather than function-oriented. This enables the DBMS to integrate data in a manner which enables multiple users to view the same data in different frames of reference without having to create dedicated files or application programs. The database concept is designed for generality, flexibility, and extensibility involving the records and files that comprise it.

Many problems existing in the applications approach are overcome by the DBMS characteristic of centralization of data. First, centralized data helps avoid data redundancy, thus contributing to potential savings in storage space. Additionally, cost savings are also realized in the updating or modifying of the database because updates or modifications only have to be done once. Finally, and most importantly, data centralization results in increased data integrity.

A second major benefit of DBMS is the existence of logical and physical data independence. Logical independence means that the overall logical structure of the data may

be changed without changing the application programs. Physical data independence means that the physical layout and the organization of data may be changed without changing either the overall logical structure of the data or the applications programs [Ref. 15]. In summary, a DBMS yields advantages to management in more information from a given amount of data, in improved responses to information requirements, and in flexibility when answering unanticipated inquiries.

With logical and physical data independence, users may create multiple logical views of a single representation of the data. The logical design can be specified in a variety of ways. One such technique is called "data structure diagrams," or "Bachman diagrams," after the man who invented them. The single/double arrows notation is used to express relationships among records of the databases. This relationship is done through an intersection file created by the union of common records in the files [Ref. 16]. The Bachman diagram in Figure 5.1 only shows the relationships among records. The contents of these records is documented in the data dictionary. The intersection file is aircraft.

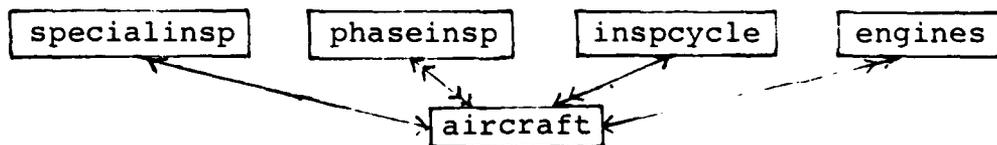


Figure 5.1

The arrows indicate relationships between the boxes which represent files in the database. Single arrows represent a one-to-one relationship, e.g., specialinsp and aircraft. In this case, one aircraft may require one special inspection and only one special inspection may be required on a single aircraft. One-to-many relationships are represented by double and single ended arrows, e.g., engines and aircraft. This relationship among files allows for many engines to be in the aircraft file, but only a single aircraft to be in the file for a specific engine. Finally, many-to-many relationships are represented by double ended arrows, e.g., phaseinsp and aircraft. In this instance, many aircraft may require any number of phase inspections and many phase inspections could be required on many aircraft.

## C. PROGRAMMING

### 1. Program Module Hierarchy

The Maintenance Program operates using a series of application modules, Figure 5.2, which access the database described above. The user need not be familiar with the names or even the functions, as they are explained interactively. A quick overview will indicate what to expect as you step through the procedures. All sessions begin with the "START" module and branch to the various options. All modules terminate to the main menu, Figure 5.3, which gives the user the option of reinitializing or ending the session.

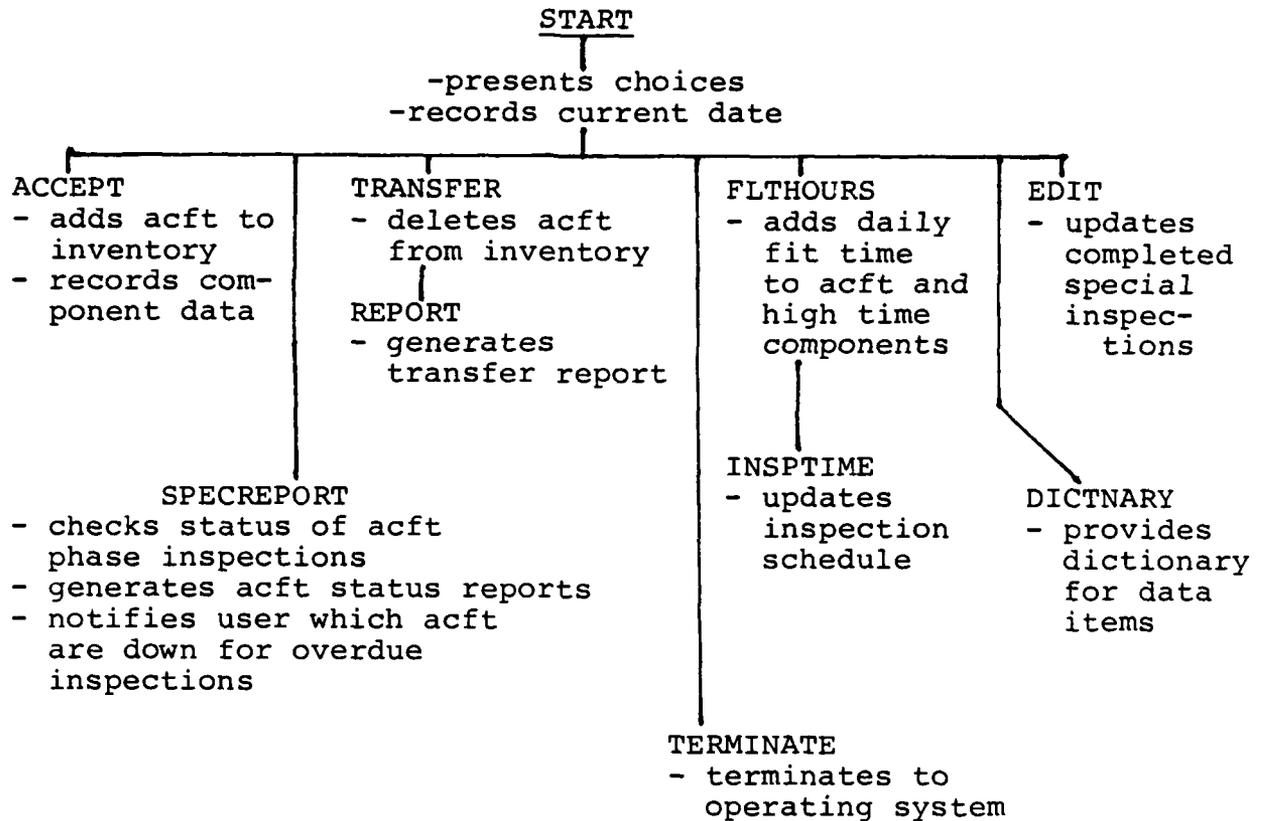


Figure 5.2

**PROGRAM MAIN MENU SELECTION**

This program accesses the aircraft inspection and component database. You can access or update the database by selecting one of the following options. Type the number corresponding to your selection.

1. ACCEPT AN AIRCRAFT
2. TRANSFER AN AIRCRAFT
3. ADD FLIGHT TIME TO AN AIRCRAFT
4. UPDATE A COMPLETED SPECIAL INSPECTION
5. PRINT A REPORT OF UPCOMING AIRCRAFT INSPECTIONS
6. USE THE DATA DICTIONARY
7. To TERMINATE this session enter "7"

Figure 5.3

## 2. Programming Techniques

Because users are not required or expected to have much computer experience, terms and procedures used are as simple as possible. Note that even terms in the menu for selecting desired options are not abbreviated. Also, the menu allows the user to either return to the main program or end the session before any actions are required.

Each command module begins with an introductory section for documentation. Notice that the section is well highlighted for quick reference. Additional documentation is located throughout the program modules and is also highlighted. For fixed length entries such as Julian date, only enough spaces are provided to accommodate exactly four digits.

Although the program was written by three authors, consistent indentations for DO WHILE and IF-ELSE statements are the same making the program logic easy to follow. Also, note that acronyms are consistent throughout the program. The DD/DS was most helpful when defining these entities.

Because the dBASE "REPORT" function is not conducive for use with more than one file, most of the modules have their own embedded display routines for print and/or screen display. The embedded routine approach also allows for simpler program maintenance. In almost all cases, the

dBASE II REPLACE and LIST commands were used to output required information.

The language and abbreviations used are common in the aviation community. However, some experience in Navy maintenance procedures is required to ensure database integrity. Modifying values and program functions cannot be done using the MAINTENANCE PROGRAM. These functions are considered in the domain of an individual trained for that responsibility. The UPDATE modules do allow values within the database files to be changed. This would be necessary and a regular procedure for almost any real world user.

## VI. CONCLUSIONS AND RECOMMENDATIONS

### A. CONCLUSIONS

Increased Fleet activity and reduced manning levels has resulted in a severe administrative burden on personnel that has significantly affected performance and morale. The purpose of microcomputer automation is to help eliminate much of this burden.

Administrative database requirements and managerial decision support needs can be met with microcomputer technology. Microcomputer hardware and software provide the qualified user with a tremendous amount of increased productivity potential. Microcomputers are a feasible solution to numerous applications because hardware and software are inexpensive, available, and easy to use. Recent studies have shown microcomputer hardware to be environmentally tolerant, portable, and reliable.

The research conducted during this study documents the serious need for proper implementation of the Zenith 120 microcomputer at the squadron level. The result of such an implementation effort would be the increased utilization of microcomputer assets currently onboard.

### B. RECOMMENDATIONS

The recommendations provided fall into three broad categories; command attention, DD/DS, and education.

1. Command Attention

Because microcomputers already exist in hundreds of commands, Type commanders are encouraged to establish policy that supports microcomputer use. Centralized control, even of decentralized computing, must be maintained. Standardized data formats, communication channels, and additional hardware should be provided by designated program managers. These program managers will become critical links in a command's ability to utilize their computer assets. These should not be collateral duty assignments, but billets filled by specially trained officers that are capable and motivated. These assignments could be filled by graduates of the computer systems and computer technology curriculums at the Naval Postgraduate School with the XX95P code.

Additionally, squadrons should be provided with approved application programs for maintenance, operations, safety, administration, and personnel departments. A central clearing house of approved programs maintained at a level no lower than the functional Wing commander is essential to control and maintain these programs.

2. DD/DS

A second important recommendation is the establishment of a Data Dictionary/Directory System for application programs. This central repository of data definitions, controls, and standards would ease considerably the plethora of terms used in locally generated programs at the squadron

level. The DD/DS would provide a common link between all squadrons and would promote the sharing of application programs, ease training requirements, and reduce data ambiguity. As confidence in the automation process increases, the DD/DS would become the primary tool for future system development.

### 3. Education

The operation and control of the data environment created by the microcomputer, will greatly impact the way in which the Navy currently does business. At the minimum, the person in charge of the microcomputer within the squadron should be an officer. He should be responsible for the overall use of the system, including development of software, data integrity, and security. Because most personnel are not experienced in each of these areas, sufficient training is essential. Current NARDAC courses offer an excellent opportunity for filling a part of this need. However, these classes should be expanded to include such areas as relational database, fourth generation languages, decision support systems, and artificial intelligence. Additionally, program texts should be made available to users of the system that are easy to read and designed for on-line instruction.

Other areas previously mentioned in the thesis that could increase awareness and improve utilization include; a west coast version of "Chips Ahoy," an electronic

bulletin board, a station user group, modems, and letter quality printers.

The intention of this thesis was to help foster better understanding and acceptance of the microcomputer in the Navy. That goal is accomplished by contributing to the ongoing implementation and validation effort. Evaluation and implementation of this study's recommendations will allow greater employment of the Zenith 120. Applications of microcomputer technology to many squadron activities is available now. This work will assist others in recommendations and further analysis of computer use.

APPENDIX A

REQUIREMENTS ANALYSIS SURVEY

This appendix contains the survey and cover letter sent to sixty users of the Zenith 120 in West Coast squadrons. Results are summarized within the example survey.

20 April 1985

From: LCDR G. A. Barnett  
SMC #1691  
Naval Postgraduate School  
Monterey, CA. 93943-5100

To: Fleet User

Subj: Zenith 120 Training and Implementation Questionnaire

1. Recently, you attended all or part of a Z-120 microcomputer course provided by NARDAC, San Diego. The course was designed as a "hands-on" learning experience and to provide as much practical knowledge as possible. At the end of the course, you were asked to complete an evaluation form to assist in future course development and improvement. Now that you have had an opportunity to use what you have learned, NARDAC has asked me to assist them in determining how effective your training was. Your effort in this endeavor is greatly appreciated.

2. Some questions are of a general nature and are designed to determine how your command is utilizing the computer. Please feel free to comment both positively and negatively as this information will be used to assist in the implementation of future computer systems and courses.

3. Your participation is an opportunity for you to provide a candid response to the systems you receive and will be of high value to the Navy. Thank you for your cooperation and effort in this research.

4. Please return your questionnaire and comments to:

LCDR G. A. Barnett  
SMC #1691  
Naval Postgraduate School  
Monterey, CA. 93943-5100

If you have any further questions and would like to contact me at NPS my telephone number is (A)878-2174 or (C)408-646-2174.

## Section I. USER INFORMATION FORM

The purpose of this evaluation is to improve the computer systems and training which you receive. If you put your name on the questionnaire, feedback concerning the problems you have identified will be provided.

Date \_\_\_\_\_ (day) \_\_\_\_\_ (month) \_\_\_\_\_ (year)  
Experience with this system \_\_\_\_\_ (years) \_\_\_\_\_ (months)  
Name/Rank \_\_\_\_\_  
Branch/Division \_\_\_\_\_  
Job Title \_\_\_\_\_  
Organization \_\_\_\_\_

## Section II. QUESTIONNAIRE

Please check your choice or fill in the answer.

1. Approximately how many hours per week do you use the computer? Avg. = 9.1
2. How many people in your organization have you trained to use the computer? Avg. = 2.5
3. Is there a formal training plan established in your command for the computer system? yes = 3, no = 28
4. Is one planned for the future? yes = 6, no = 18, n/a = 5
5. Would you like to attend an indepth software development course? yes = 18, no = 13
6. Do you use the NARDAC handouts in conjunction with the commercial user manuals? yes = 23, no = 8
7. Are the examples in the NARDAC handouts adequate? yes = 23, no = 8
8. Are there enough examples in the handouts? yes = 18, no = 13
9. Was too much information covered in the time you had for the Z-120 course? yes = 12, no = 19 How much time should have been allowed? Avg. = 10 (days)
10. Would you like to take a follow-on course for any specific subject? yes = 19, no = 12  
Which one? Lotus, dBASE, Wordstar How long? Avg. = 5, 3, 2
11. Do you utilize any commercial software besides what was provided with the system? yes = 13, no = 18

12. If yes what are you using? Condor, Sidekick, Word, Peach
13. Has the system enabled you to perform your task more efficiently? yes = 30, no = 1
14. Is system down time a problem? yes = 0, no = 31
15. Where is the system physically located in your command?  
All departments represented.
16. Is there sufficient room for peripheral equipment, such as printers, telephone, etc.? yes = 30, no = 1
17. Is lighting adequate and not glaring on the screen?  
yes = 29, no = 2
18. Are you using double sided/double density diskettes?  
yes = 29, no = 2
19. Have you or anyone in your command contacted anyone with regard to developing programs which may be more complex than you wish to attempt? yes = 7, no = 21
20. Have you or anyone in your command been contacted by anyone to determine your needs? yes = 6, no = 25
21. Have any problems with using the system been identified and documented? yes = 10, no = 21
22. Is the printer satisfactory for all of your needs?  
yes = 10, no = 21
23. Are you presently using any programs that you feel would benefit other commands? yes = 10, no = 21
24. Are there any specific areas that you feel a software program could provide significant help, but do not know who to contact? yes = 24, no = 7

### Section III. SUGGESTIONS AND COMMENTS

Experience has shown that users are by far the best source for information on what they need. Please list any problems or recommendations you feel need to be addressed. Thank you for your time and cooperation.

LCDR G. A. Barnett

## APPENDIX B

### SOFTWARE DATA DICTIONARY/DIRECTORY SYSTEM

This appendix contains the user manual, programming code, file structure of the SDD/DS.

To initialize the Software Program:

1. Turn on the computer, CRT, and printer.
2. Load the Software Program disk in drive B.
3. Load the system operating disk in drive A.
4. "Boot" the system and enter the data as <MM/DD/YY>.
5. When the A> appears type: dbase.
6. The computer will load the database system program and respond with a "." as the program cue.
7. Type: set default to B.
8. Type: do software.
9. Use the menu to select the desired subject.

Instructions at this point are displayed on the screen.

If you make an error while entering a choice, the screen will return you to the main menu. If you desire to print a listing of available programs, use the print screen option.

To exist the system:

10. Select option 6.
11. Remove the disks from the disk drives.
12. Turn off the printer, CRT, and computer.

```

***** SOFTWARE CMD *****
*****
*AUTHOR: LCDR BARNETT *
*PURPOSE: This program is a passive and independent data*
* dictionary/directory system that allows the user to *
* search and find application programs for squadron *
* use. It uses a menu selection format and presents *
* six options for the user to select, including one to*
* terminate the session. *
*INPUT FILES USED: MAINTENANCE, OPERATIONS, ADMIN, *
* SAFETY, and PERSONNEL. *
*LOCAL VARIABLES USED: SELECT1 *
*****

```

```

CLEAR
SET BELL OFF
SET TALK OFF

```

```

***** Screen format presented here *****

```

```

STORE 0 TO SELECT1
DO WHILE SELECT1 < 6
ERASE
@ 5, 5 SAY "SOFTWARE DATA DICTIONARY/DIRECTORY SYSTEM"
@ 7, 5 SAY "This DD/DS allows you to find programs you may"
@ 8, 5 say "want to use in your squadron. Simply type the"
@ 9, 5 say "number of the corresponding department you need."
@ 11,10 SAY "1. MAINTENANCE PROGRAMS"
@ 12,10 SAY "2. OPERATIONS PROGRAMS"
@ 13,10 SAY "3. ADMINISTRATION PROGRAMS"
@ 14,10 SAY "4. SAFETY PROGRAMS"
@ 15,10 SAY "5. PERSONNEL PROGRAMS"
@ 16,10 SAY "6. EXIT FROM SYSTEM";
GET SELECT1 PICTURE "#"
READ

```

```

***** OPTION SELECTION *****

```

```

IF SELECT1 = 1
ERASE
USE MAINTENANCE
LIST
? "TYPE ANY KEY TO CONTINUE"
WAIT TO CONTINUE
ENDIF
IF SELECT1 = 2
ERASE
USE OPERATIONS
LIST
? "TYPE ANY KEY TO CONTINUE"
WAIT TO CONTINUE

```

```

ENDIF
IF SELECT 1 = 3
    ERASE
    USE ADMIN
    LIST
    ? "TYPE ANY KEY TO CONTINUE"
    WAIT TO CONTINUE
ENDIF
IF SELECT1 = 4
    ERASE
    USE SAFETY
    LIST
    ? "TYPE ANY KEY TO CONTINUE"
    WAIT TO CONTINUE
ENDIF
IF SELECT1 = 5
    ERASE
    USE PERSONNEL
    LIST
    ? "TYPE ANY KEY TO CONTINUE"
    WAIT TO CONTINUE
ENDIF
ENDDO
IF SELECT 1 = 6
    ERASE
    RETURN
ENDIF

```

\*\*\*\*\* ERROR ROUTINE \*\*\*\*\*

```

DO WHILE (SELECT1 <> 1) .AND. (SELECT1 <> 2) .AND. (SELECT1 <> 3);
    .AND. (SELECT1 <> 4) .AND. (SELECT1 <> 5) .AND. (SELECT1 <> 6)
    ERASE
    @ 10, 8 SAY "ERRONEOUS INPUT, TRY AGAIN"
    @ 12,10 SAY "Type the NUMBER corresponding to your selection."
    @ 14,10 SAY "1. MAINTENANCE PROGRAMS"
    @ 15,10 SAY "2. OPERATIONS PROGRAMS"
    @ 16,10 SAY "3. ADMIN PROGRAMS"
    @ 17,10 SAY "4. SAFETY PROGRAMS"
    @ 18,10 SAY "5. PERSONNEL PROGRAMS"
    @ 19,10 SAY "6. EXIT FROM SYSTEM";
    GET SELECT1 PICTURE "#"
    READ
    IF SELECT1 = 1
        ERASE
        USE MAINTENANCE
        LIST
        ? "TYPE ANY KEY TO CONTINUE"
        WAIT TO CONTINUE
    ENDIF
    IF SELECT1 = 2

```

```
        ERASE
        USE OPERATIONS
        LIST
        ? "TYPE ANY KEY TO CONTINUE"
        WAIT TO CONTINUE
    ENDIF
    IF SELECT1 = 3
        ERASE
        USE ADMIN
        LIST
        ? "TYPE ANY KEY TO CONTINUE"
        WAIT TO CONTINUE
    ENDIF
    IF SELECT1 = 4
        ERASE
        USE SAFETY
        LIST
        ? "TYPE ANY KEY TO CONTINUE"
        WAIT TO CONTINUE
    ENDIF
    IF SELECT1 = 5
        ERASE
        USE PERSONNEL
        LIST
        ? "TYPE ANY KEY TO CONTINUE"
        WAIT TO CONTINUE
    ENDIF
    IF SELECT1 = 6
        ERASE
        RETURN
    ENDIF
ENDDO
```

# Software Data Dictionary File Structure

use maintenance  
list structure

STRUCTURE FOR FILE: B:MAINTENA.DBF  
NUMBER OF RECORDS: 00003  
DATE OF LAST UPDATE: 01/01/80  
PRIMARY USE DATABASE

FLD	NAME	TYPE	WIDTH	DEC
001	TITLE	C	025	
002	AUTHOR	C	025	
003	ADDRESS	C	025	
004	LANGUAGE	C	025	
005	DESCRIP	C	075	
006	OPSYS	C	010	
** TOTAL **			00186	

use operations  
list structure

STRUCTURE FOR FILE: B:OPERATIO.DBF  
NUMBER OF RECORDS: 00002  
DATE OF LAST UPDATE: 01/01/80  
PRIMARY USE DATABASE

FLD	NAME	TYPE	WIDTH	DEC
001	TITLE	C	025	
002	AUTHOR	C	025	
003	ADDRESS	C	025	
004	LANGUAGE	C	025	
005	DESCRIP	C	075	
006	OPSYS	C	010	
88 TOTAL **			00186	

use safety  
list structure

STRUCTURE FOR FILE: B:SAFETY .DBF  
NUMBER OF RECORDS: 00000  
DATE OF LAST UPDATE: 01/01/80  
PRIMARY USE DATABASE

FLD	NAME	TYPE	WIDTH	DEC
001	TITLE	C	025	
002	AUTHOR	C	025	
003	ADDRESS	C	025	
004	LANGUAGE	C	025	
005	DESCRIP	C	075	
006	OPSYS	C	010	
** TOTAL **			00161	

```

use admin
list structure
STRUCTURE FOR FILE: B:ADMIN .DBF
NUMBER OF RECORDS: 00005
DATE OF LAST UPDATE: 01/01/80
PRIMARY USE DATABASE
FLD      NAME      TYPE WIDTH  DEC
001     TITLE     C      025
002     AUTHOR    C      025
003     ADDRESS   C      025
004     LANGUAGE  C      025
005     DESCRIP   C      075
006     OPSYS     C      010
** TOTAL **                00186

```

```

use personnel
list structure
STRUCTURE FOR FILE: B:PERSONNE.DBF
NUMBER OF RECORDS: 00001
DATE OF LAST UPDATE: 01/01/80
PRIMARY USE DATABASE
FLD      NAME      TYPE WIDTH  DEC
001     TITLE     C      025
002     AUTHOR    C      025
003     ADDRESS   C      025
004     LANGUAGE  C      025
005     DESCRIP   C      075
006     OPSYS     C      010
** TOTAL **                00161

```

Example of SDD/DS Maintenance Programs

```

00001 FLIGHT HOURS          LT J.W. MARTIN      VS-33
      DBASE II             UPDATES FLIGHT HOURS USING
                          YELLOW SHEET FORMAT
                          MS-DOS
00002 MAINTENANCE PROGRAM  LTJG WARNER         VF-211
      DBASE II             UPDATES INSPECTIONS SCHEDULE
                          MS-DOS
00003 MAINTENANCE SCHEDULING LCDR BARNETT        VF-21
      DBASE                DSS FOR SCHEDULING PHASE AND
                          SPECIAL INSPECTIONS
                          MS-DOS

TYPE ANY KEY TO CONTINUE
WAITING

```

Example of SDD/DS Operations Programs

00001 TRAINING DATABASE  
DBASE II

CDR J. COMPTON CINCLANTFLT  
ALLOWS INPUT AND QUERY OF  
TRAINING RECORDS.

MS-DOS

00002 BYDATE.NDX  
DBASE

NARDAC, NORFOLK  
CALENDAR.DBF FOR PRINTER  
SETUP

DOS

TYPE ANY KEY TO CONTINUE  
WAITING

## APPENDIX C

### SQUADRON MAINTENANCE/MANAGEMENT SYSTEM

This appendix contains the SM/MS user manual, program code, file structure, and examples of screen and printer output. All instructions are presented on the screen. Once the system is loaded and initialized, the CRT will display instructions to step the user through the procedures. The user is asked before he begins a module if this is really where he wants to be. If he does not desire to perform the requested operation, a no answer N will return him to the main menu.

To initialize the Maintenance Program:

1. Turn on the computer, CRT, and printer.
2. Load the Maintenance Program disk in drive B.
3. Load the system operating disk in drive A.
4. "Boot" the system and enter the date as <MM/DD/YY>.  
Be sure to enter the correct date as it is used later in the program.
5. When the A> appears type: <dbase> return  
The computer will load the database system program and respond with a "." as the program cue.
6. Type <set default to B>. return
7. Type <DO START>. return
8. Enter the Julian date.
9. Use the menu to select the desired operation.

Instructions at this point are displayed on the screen. If you make an error while entering a choice or entering data, the program will allow you to correct it or return to the main menu. This double error checking may seem redundant, but all files are updated and calculation made based on the data that you insert. It is well worth the extra time to ensure data integrity.

If you see A> return to step 5 above and reinitiate the program. If you do not desire to re-enter the program, remove the system disk, turn off the printer, the computer, and the CRT.

```

***** START CMD *****
*****
* AUTHOR: LCDR BARNETT *
* PURPOSE: This program initializes the aircraft and *
* component update process. It accepts the Julian *
* Date and presents seven options to the user, *
* including one to terminate session. *
* INPUT FILES USED: None *
* OUTPUT FILES USED: None *
* ROUTINES CALLED: This program calls ACCEPT, TRANSFER, *
* FLTHOURS, EDIT, and DICTNARY *
* ROUTINES CALLED BY: None *
* LOCAL VARIABLES: SELECT1 *
* MAJOR CONTROL POINTS: The value of SELECT1 determines *
* which program is CALLED. *
*****

```

```

CLEAR
SET BELL OFF
SET TALK OFF

```

```

***** Julian Date entered here *****

```

```

STORE 0 TO JULDATE
ERASE
@ 5,10 SAY "ENTER TODAY'S JULIAN DATE" GET JULDATE PICTURE '####'
@ 7,10 SAY 'A VALID JULIAN DATE IS BETWEEN 5000 AND 5365'
@ 20,20 SAY 'UPDATE THIS SCREEN ON 1 JAN 198X'
READ
DO WHILE (JULDATE < 5000) .OR. (JULDATE > 5365)
  ERASE
  @ 5,10 SAY 'DATE MUST BE BETWEEN 5000 and 5365'
  @ 7,10 SAY 'PLEASE REENTER DATE' GET JULDATE PICTURE '####'
  READ
ENDDO

```

```

***** SCREEN FORMAT/OPTION PRESENTED HERE *****

```

```

STORE 0 TO SELECT1
DO WHILE SELECT1 < 7
ERASE
@ 5,10 SAY 'PROGRAM MAIN MENU SELECTION'
@ 7,5 SAY 'This program accesses the aircraft inspection'
@ 8,5 SAY 'and component database. You can access or update the'
@ 9,5 SAY 'database by selecting one of the following options.'
@ 10,5 SAY 'Type the number corresponding to your selection.'
@ 12,10 SAY '1. ACCEPT AN AIRCRAFT'
@ 13,10 SAY '2. TRANSFER AN AIRCRAFT'
@ 14,10 SAY '3. ADD FLIGHT TIME TO AN AIRCRAFT'
@ 15,10 SAY '4. UPDATE A COMPLETED SPECIAL INSPECTION'
@ 16,10 SAY '5. PRINT A REPORT OF UPCOMING AIRCRAFT INSPECTIONS'

```

```
@ 17,10 SAY '6. USE THE DATA DICTIONARY'  
@ 18,10 SAY '7. To TERMINATE this session enter "7"';  
GET SELECT1 PICTURE '#'
```

```
READ
```

```
***** OPTION SELECT *****
```

```
IF SELECT1 = 1  
  ERASE  
  DO ACCEPT  
ENDIF  
IF SELECT1 = 2  
  ERASE  
  DO TRANSFER  
ENDIF  
IF SELECT1 = 3  
  ERASE  
  DO FLTHOURS  
ENDIF  
IF SELECT1 = 4  
  ERASE  
  DO EDIT  
ENDIF  
IF SELECT1 = 5  
  ERASE  
  DO SPECREPORT  
ENDIF  
IF SELECT1 = 6  
  ERASE  
  DO DICTNARY  
ENDIF  
ENDDO  
IF SELECT1 = 7  
  ERASE  
  RETURN  
ENDIF
```

```
***** ERROR ROUTINE *****
```

```
DO WHILE (SELECT1 <> 1) .AND. (SELECT1 <> 2) .AND. (SELECT1 <> 3);  
  .AND. (SELECT1 <> 4) .AND. (SELECT1 <> 5) .AND. (SELECT1 <> 6);  
  .AND. (SELECT1 <> 7)  
  ERASE  
  @ 10,8 SAY 'ERRONEOUS INPUT, TRY AGAIN'  
  @ 12,10 SAY 'Type the NUMBER corresponding to your selection.'  
  @ 14,10 SAY '1. ACCEPT AN AIRCRAFT'  
  @ 15,10 SAY '2. TRANSFER AN AIRCRAFT'  
  @ 16,10 SAY '3. ADD FLIGHT TIME TO AN AIRCRAFT'  
  @ 17,10 SAY '4. UPDATE A COMPLETED SPECIAL INSPECTION'  
  @ 18,10 SAY '5. PRINT REPORT OF FUTURE AIRCRAFT INSPECTIONS'
```

```
@ 19,10 SAY '6. USE THE DATA DICTIONARY'  
@ 20,10 SAY '7. To TERMINATE this session enter a "7"';  
GET SELECT1 PICTURE '#'  
READ  
IF SELECT1 = 1  
    ERASE  
    DO ACCEPT  
ENDIF  
IF SELECT1 = 2  
    ERASE  
    DO TRANSFER  
ENDIF  
IF SELECT1 = 3  
    ERASE  
    DO FLTHOURS  
ENDIF  
IF SELECT1 = 4  
    ERASE  
    DO EDIT  
ENDIF  
IF SELECT1 = 5  
    ERASE  
    DO SPECREPORT  
ENDIF  
IF SELECT1 = 6  
    ERASE  
    DO DICTNARY  
ENDIF  
IF SELECT1 = 7  
    QUIT  
ENDIF  
ENDDO  
QUIT
```

```

***** ACCEPT CMD *****
*****
* AUTHOR: LCDR BARNETT *
* PURPOSE: This program allows the user to add a new air- *
* craft to the AIRCRAFT file and its components to their *
* files *
* INPUT FILES USED: AIRCRAFT *
* OUTPUT FILES USED: AIRCRAFT and ENGINES *
* ROUTINES CALLED: None *
* ROUTINES CALLED BY: START *
* LOCAL VARIABLES: Local variables include ANSWER, *
* MEMENGL, SERNO, MEMBUNO and MEMENG2 *
* CONTROL POINTS: The variable ANSWER is the control *
* point in this module. *
*****

```

\*\*\*\*\* Gives user a chance to exit program \*\*\*\*\*

```

USE AIRCRAFT
INDEX ON BUNO TO BUNO
DO WHILE T
  ERASE
  ? "DO YOU WISH TO ENTER A NEW AIRCRAFT RECORD? (Y/N)"
  ACCEPT TO ANSWER
  IF $(ANSWER,1,1) <> "Y"
    RETURN
  ENDIF
  USE AIRCRAFT INDEX BUNO
  STORE " " TO MEMBUNO
  @ 5,5 SAY "ENTER AIRCRAFT BUNO" GET MEMBUNO
  READ
  FIND &MEMBUNO
  IF # <> 0
    ? "A RECORD FOR THIS BUNO ALREADY EXISTS"
    ? "UNABLE TO ENTER ANOTHER RECORD FOR THIS AIRCRAFT"
    ? "TYPE ANY KEY TO CONTINUE"
    WAIT TO CONTINUE

```

\*\*\*\*\* Allows user to update AIRCRAFT file \*\*\*\*\*

```

ELSE
  APPEND BLANK
  REPLACE BUNO WITH MEMBUNO
  EDIT CURRENT
  STORE ENGLSERNO TO MEMENGL
  STORE ENG2SERNO TO MEMENG2

```

\*\*\*\*\* Allows user to update ENGINES file \*\*\*\*\*

```
USE ENGINES
  APPEND BLANK
  REPLACE SERNO WITH MEMENG1
  ? " THE FOLLOWING SCREENS UPDATE ENGINE DATA"
  ? "TYPE ANY KEY TO CONTINUE"
WAIT TO CONTINUE
EDIT CURRENT
APPEND BLANK
REPLACE SERNO WITH MEMENG2
EDIT CURRENT
```

\*\*\*\*\* Allows user to update PHASEINSP file \*\*\*\*\*

```
USE PHASEINSP
  APPEND BLANK
  REPLACE BUNO WITH MEMBUNO
  ? "NEXT SCREEN UPDATES THE PHASE INSP FILE"
  ? "TYPE ANY KEY TO CONTINUE"
WAIT TO CONTINUE
EDIT CURRENT
```

\*\*\*\*\* Allows user to update SPECIALINSP file \*\*\*\*\*

```
USE SPECIALINSP
  APPEND BLANK
  REPLACE BUNO WITH MEMBUNO
  ? "NEXT SCREEN UPDATES THE SPECIAL INSP FILE"
  ? "PRESS ANY KEY TO CONTINUE"
WAIT TO CONTINUE
EDIT CURRENT
```

```
ENDIF
ENDDO
```

```

***** TRANSFER CMD *****
*****
* AUTHOR: Jim Compton *
* PURPOSE: This program will allow the user to delete all *
* data about an aircraft and also provides a printed *
* AIRCRAFT TRANSFER REPORT. *
* INPUT FILES USED: AIRCRAFT *
* OUTPUT FILES USED: AIRCRAFT and ENGINES *
* ROUTINES CALLED: REPORT *
* ROUTINES CALLED BY: START *
* LOCAL VARIABLES: Local variables include ANSWER, *
* MEMENG1, MEMENG2, SERNO, and MEMBUNO. *
* CONTROL POINTS: The variable ANSWER is the control *
* point in this module. *
*****

```

```

***** Allows user to exit program *****

```

```

USE AIRCRAFT
INDEX ON BUNO TO BUNO
DO WHILE T
  ERASE
  ? 'DO YOU WISH TO TRANSFER AN AIRCRAFT? (Y/N) '
  ACCEPT TO ANSWER
  IF $(ANSWER,1,1) <> "Y"
    RETURN
  ENDIF
  USE AIRCRAFT INDEX BUNO
  STORE " " TO MEMBUNO
  @ 5,5 SAY "ENTER AIRCRAFT BUNO TO BE DELETED" GET MEMBUNO
  READ
  FIND &MEMBUNO
  IF # = 0
    ? 'AN AIRCRAFT RECORD DOES NOT EXIST FOR THIS BUNO '
    ? 'PRESS RETURN TO CONTINUE '
    WAIT TO CONTINUE
    RETURN
  ENDIF
  IF # > 0
    @ 10,5 SAY "TURN PRINTER ON FOR PERMANENT RECORD"
    @ 11,5 SAY "PRESS ANY KEY WHEN PRINTER IS READY"
    WAIT TO CONTINUE

```

```

***** CALLS the REPORT program *****

```

```

DO REPORT

```

\*\*\*\*\* Record deletion part of this program begins here \*\*\*\*\*

```
SELECT PRIMARY
USE AIRCRAFT INDEX BUNO
FIND &MEMBUNO
  STORE ENGLSERNO TO MEMENG1
  STORE ENG2SERNO TO MEMENG2
  STORE TRANSERNO TO MEMTRANS
  STORE ROTORSERNO TO MEMROTOR
  STORE GEAR1SERNO TO MEMGEAR1
  STORE GEAR2SERNO TO MEMGEAR2
  STORE GEAR3SERNO TO MEMGEAR3
```

```
SELECT SECONDARY
USE ENGINES
INDEX ON SERNO TO SERNO
FIND &MEMENG1
  DELETE NEXT 1
  FIND &MEMENG2
  DELETE NEXT 1
  PACK
```

```
USE PHASEINSP
INDEX ON BUNO TO BUNO
FIND &MEMBUNO
  DELETE NEXT 1
  PACK
```

```
USE SPECIALINSP
INDEX ON BUNO TO BUNO
FIND &MEMBUNO
  DELETE NEXT 1
  PACK
```

```
SELECT PRIMARY
USE AIRCRAFT
INDEX ON BUNO TO BUNO
FIND &MEMBUNO
  DELETE NEXT 1
  PACK
```

```
ENDIF
ENDDO
RETURN
```

```

***** REPORT CMD *****
*****
* AUTHOR: Jim Compton *
* PURPOSE: Print the TRANSFER REPORT. *
* INPUT FILES USED: AIRCRAFT and ENGINES *
* OUTPUT FILES USED: REPORT1 and REPORT 2 *
* ROUTINES CALLED: None *
* ROUTINES CALLED BY: TRANSFER *
* VARIABLES USED: None *
* CONTROL POINTS: None *
*****

```

```

ERASE
SET PRINT ON
SELECT PRIMARY
USE AIRCRAFT
SELECT SECONDARY
USE ENGINES
JOIN TO REPORT1 FOR P.ENGLSERNO = S.SERNO .AND. P.BUNO = MEMBUNO;
FIELD P.BUNO,P.FLIGHTHRS,P.ENGLSERNO,S.CURRENTTIME,P.ENG2SERNO
SELECT PRIMARY
USE REPORT1
SELECT SECONDARY
USE ENGINES
JOIN TO REPORT2 FOR P.ENG2SERNO = S.SERNO .AND. P.BUNO = MEMBUNO;
FIELD P.BUNO,P.FLIGHTHRS,P.ENGLSERNO,P.CURRENTTIME,P.ENG2SERNO ;
S.CURRENTTIME

```

\*\*\*\*\* The report title and headings are printed \*\*\*\*\*

```

? '                AIRCRAFT TRANSFER REPORT'
? '                JULIAN DATE IS'
?? +JULDATE
? '
? '
? ' ACFT   FLIGHT ENG 1   ENG 1   ENG 2   ENG2'
? ' BUNO   HOURS  SERNO   TIME    SERNO   TIME'
? '

```

\*\*\*\*\* Data printed \*\*\*\*\*

```

USE REPORT2
LIST OFF
?
SET PRINT OFF
RETURN

```

```

***** SPECREPORT CDM *****
*****
* AUTHOR: LCDR ROWSON *
* PURPOSE: To print a report which lists aircraft and *
* their upcoming special inspection requirements. *
* INPUT FILES: SPECIALINSP *
* OUTPUT FILES: None *
* ROUTINES CALLED: None *
* ROUTINES CALLED BY: START *
* LOCAL VARIABLES: Local variables include ANSWER, *
* MEMQUEST, TEMPBOUND, and MEMREC. *
* CONTROL POINTS: Variable ANSWER controls entry to this *
* module. *
*****

```

```

SET TALK OFF
ERASE
STORE 0 TO MEMQUEST
ERASE
DO WHILE T
  ? 'DO YOU WISH A REPORT OF AIRCRAFT DUE FOR SPECIAL INSP
  (Y/N) '
  ACCEPT TO ANSWER
  IF $(ANSWER,1,1) <> 'Y'
    RETURN
  ENDIF
  @ 7,5 SAY 'ENTER THE REQUESTED PERIOD IN DAYS' GET MEMQUEST
  READ
  ? 'ENSURE THAT PRINTER IS ON'
  ? 'TYPE ANY KEY TO CONTINUE'
  WAIT TO CONTINUE
  ERASE
  SET PRINT ON

```

\*\*\*\*\* Report title and headings are printed \*\*\*\*\*

```

?'          AIRCRAFT SPECIAL INSPECTION REPORT'
?'          JULIAN DATE IS',+JULDATE
?
?'          BELOW IS THE STATE OF SPECIAL INSPECTIONS FOR THE NEXT'
?'          ',+MEMQUEST,'DAYS'
?
?
?' BUNO          TYPE INSP          DUE DATE '
?
SET PRINT OFF
USE SPECIALINSP
GOTO TOP
DO WHILE .NOT. EOF

```

```

STORE BUNO TO TEMPBUNO
STORE # + 1 TO MEMREC
ERASE
  IF JULDATE + MEMQUEST > DAY112 + 112
    STORE (DAY112 + 112) TO STORE112
    SET PRINT ON
    LIST FOR BUNO = TEMPBUNO OFF FIELD BUNO
    ??'    112 DAY', STORE 112
    ?
    SET PRINT OFF
    GOTO MEMREC
  ELSE
    IF JULDATE + MEMQUEST > DAY56 + 56
      STORE (DAY56 + 56) TO STORE56
      SET PRINT ON
      LIST FOR BUNO = TEMPBUNO OFF BUNO
      ??'    056 DAY', STORE56
      ?
      SET PRINT OFF
      GOTO MEMREC
    ELSE
      IF JULDATE + MEMQUEST > DAY28 + 28
        STORE (DAY28 + 28) TO STORE28
        SET PRINT ON
        LIST FOR BUNO = TEMPBUNO OFF BUNO
        ??'    028 DAY', STORE28
        ?
        SET PRINT OFF
        GOTO MEMREC
      ELSE
        IF JULDATE + MEMQUEST > DAY14 + 14
          STORE (DAY14 + 14) TO STORE14
          SET PRINT ON
          LIST FOR BUNO = TEMPBUNO OFF BUNO
          ??'    014 DAY', STORE14
          ?
          SET PRINT OFF
          GOTO MEMREC
        ELSE
          IF JULDATE + MEMQUEST > DAY7 + 7
            STORE (DAY7 + 7) TO STORE7
            SET PRINT ON
            LIST FOR BUNO = TEMPBUNO OFF BUNO
            ??'    007 DAY ', STORE7
            ?
            SET PRINT OFF
            GOTO MEMREC
          ELSE
            SET PRINT ON
            LIST FOR BUNO = TEMPBUNO OFF BUNO
            ?? '    NONE REQD'
            ?
            SET PRINT OFF
            GOTO MEMREC

```

```
                ENDIF  
            ENDIF  
        ENDIF  
    ENDIF  
    ENDDO  
RETURN  
ENDDO
```

```

***** FLTHOURS CMD *****
*****
* AUTHOR: Jim Compton *
* PURPOSE: Updates flight hours on aircraft and on high *
*           time components *
* INPUT FILES USED: AIRCRAFT *
* OUTPUT FILES USED: AIRCRAFT and ENGINES *
* ROUTINES CALLED: INSPTIME *
* ROUTINES CALLED BY: START *
* LOCAL VARIABLES: Local variables include MEMBUNO, *
*                   FLTHRS, TOTALHRS, ENGL, ENG2, ENGLTT, and ENG2TT. *
* CONTROL POINTS: MEMBUNO, which determines a valid BUNO *
*                   is the control point for this module. *
*****

```

```

SET BELL OFF
SET TALK OFF
ERASE

```

```

*** Allows user to input aircraft BUNO and flight time ***

```

```

STORE 0 TO FLTHRS
ACCEPT 'WHAT IS THE ACFT BUNO NUMBER?' TO MEMBUNO
@ 13,5 SAY 'HOW MANY FLIGHT HOURS WERE FLOWN?';
   GET FLTHRS PICTURE '99.9'
READ
USE AIRCRAFT
INDEX ON BUNO TO BUNO
FIND &MEMBUNO
IF # = 0
   ? 'AN AIRCRAFT RECORD FOR THIS BUNO DOES NOT EXIST'
   ? 'TYPE ANY KEY TO CONTINUE'
   WAIT TO CONTINUE
   RETURN
ENDIF

```

```

***** Updates aircraft and component times *****

```

```

GO TOP
DO WHILE .NOT. EOF
   IF BUNO = MEMBUNO
      ERASE
      STORE FLIGHTHRS + FLTHRS TO TOTALHRS
      REPLACE FLIGHTHRS WITH TOTALHRS
      STORE ENGLSERNO TO ENGL
      STORE ENG2SERNO TO ENG2
   ENDIF
   SKIP
ENDDO
DO INSPTIME

```

```
USE ENGINES
GO TOP
DO WHILE .NOT. EOF
    IF ENGL = SERNO
        ERASE
        STORE CURRENTIME + FLTHRS TO ENGLTT
        REPLACE CURRENTIME WITH ENGLTT
    ENDIF
    SKIP
ENDDO
GO TOP
DO WHILE .NOT. EOF
    IF ENG2 = SERNO
        ERASE
        STORE CURRENTIME + FLYHRS TO ENG2TT
        REPLACE CURRENTIME WITH ENG2TT
    ENDIF
    SKIP
ENDDO
? "FLIGHT TIME FIELDS HAVE BEEN UPDATED FOR THE AIRCRAFT,"
? "ENGINES."
? "PRESS ANY KEY TO CONTINUE."
WAIT TO CONTINUE
RETURN
```

```

***** INSPTIME CMD *****
*****
* AUTHOR: LCDR ROWSON *
* PURPOSE: Determines if an aircraft is subject to being *
*   inducted into a phase inspection when acft flight *
*   time is updated. A message is sent to the screen *
*   for the users information, including notification *
*   that an aircraft is down until a particular *
*   inspection. *
* INPUT FILES: AIRCRAFT, PHASEINSP, and INSPECYCLE. *
* OUTPUT FILES: ACINSP (Created by the JOIN command) *
* ROUTINES CALLED: None *
* ROUTINES CALLED BY: FLTHOURS *
* LOCAL VARIABLES: Local variables include PHASE, MEMTIME, *
*   and NEXTPHASE *
* CONTROL POINTS: None *
*****

```

```

SELECT PRIMARY
USE AIRCRAFT
SELECT SECONDARY
USE PHASEINSP
JOIN TO ACINSP FOR P.BUNO = S.BUNO .AND. P.BUNO = MEMBUNO FIELD;
P.BUNO,P.FLIGHTHRS,S.BASETIME,S.TYPE:COMP
USE INSPECYCLE
STORE PHASE TO PHASE
USE ACINSP
STORE (FLIGHTHRS - BASETIME) TO MEMTIME
  IF TYPE: COMP = "A"
    STORE "B" TO NEXTPHASE
  ENDIF
  IF TYPE:COMP = "B"
    STORE "C" TO NEXTPHASE
  ENDIF
  IF TYPE:COMP = "C"
    STORE "D" TO NEXTPHASE
  ENDIF
  IF TYPE:COMP = "D"
    STORE "A" TO NEXTPHASE
  ENDIF

```

\*\*\*\*\* Message is sent to screen \*\*\*\*\*

```

DO CASE
CASE MEMTIME < 90
  ERASE
  ?'AIRCRAFT HAS ', (PHASE-MEMTIME)
  @ 1,28 SAY 'HOURS UNTIL THE NEXT INSPECTION,'
  ?
  ? 'WHICH IS A/AN "',NEXTPHASE

```

```

@ 3,18 SAY '" PHASE INSPECTION.'
?
? 'TYPE ANY KEY TO CONTINUE'
WAIT TO CONTINUE
CASE (MEMTIME > 90) .AND. (MEMTIME <= 110)
ERASE
?'AIRCRAFT HAS HAD',MEMTIME
@ 1,29 SAY 'HOURS SINCE LAST INSPECTION'
?
?'AND IS DUE FOR A/AN "',NEXTPHASE
@ 3,24 SAY '" PHASE INSPECTION'
?
? 'TYPE ANY KEY TO CONTINUE'
WAIT TO CONTINUE
CASE MEMTIME > 110
ERASE
?'AIRCRAFT HAS HAD ', MEMTIME
@ 1,29 SAY ' HOURS SINCE LAST INSPECTION'
?
?'AND IS DOWN UNTIL A/AN "', NEXTPHASE
@ 3,27 SAY '" PHASE INSPECTION IS COMPLETED'
?
? 'TYPE ANY KEY TO CONTINUE'
WAIT TO CONTINUE
ENDCASE
RETURN

```

```

***** EDIT CDM *****
*****
* AUTHOR: LCDR BARNETT *
* PURPOSE: This program enables the user to update the *
* SPECIALINSP file when special inspections are *
* completed. *
* INPUT FILES USED: SPECIALINSP *
* OUTPUT FILES USED: SPECIALINSP *
* ROUTINES CALLED: None *
* ROUTINES CALLED BY: START *
* LOCAL VARIABLES: Local variables ANSWER, TEMPBOUND *
* INSPDATE and TYPEINSP *
* CONTROL POINTS: ANSWER controls entry into the module *
* and the value of TYPEINSP determines which *
* inspection type if updated. *

```

\*\*\*\*\* Allows user to exit program \*\*\*\*\*

```

DO WHILE T
  ERASE
  ? 'DO YOU NEED TO UPDATE AN AIRCRAFT SPECIAL INSPECTION? (Y/N)'
  ACCEPT TO ANSWER
  IF $(ANSWER,1,1) <> "Y"
    RETURN
  ENDIF

```

\*\*\*\*\* User enters the BUNO of aircraft inspected \*\*\*\*\*

```

USE SPECIALINSP
INDEX ON BUNO TO BUNO
STORE ' ' TO TEMPBUNO
@ 5,5 SAY 'ENTER THE AIRCRAFT BUNO' GET TEMPBUNO
READ
FIND &TEMPBUNO
IF # = 0
  ? 'NO RECORD FOR THIS AIRCRAFT ON FILE'
  ? 'TYPE ANY KEY TO CONTINUE'
  WAIT TO CONTINUE

```

\*\*\*\*\* User enters date and type inspection completed \*\*\*\*\*

```

ELSE
  STORE 0 TO INSPDATE
  @ 7,5 SAY 'ENTER DATE INSPECTION COMPLETED' GET INSPDATE ;
  PICTURE '9999'
  READ
  STORE 0 TO TYPE INSP
  @ 9,5 SAY 'ENTER TYPE INSP COMPLETED' GET TYPEINSP ;
  PICTURE '999'
  READ

```

```

IF (TYPEINSP <> 7) .AND. (TYPEINSP <>14) .AND.;
  (TYPEINSP <> 28) .AND. (TYPEINSP <> 56 .AND. (TYPEINSP ;
  <>112)
@ 11,5 SAY 'INVALID INSP TYPE TRY AGAIN.' GET TYPEINSP ;
  PICTURE '999'
READ
ENDIF
  IF TYPEINSP = 112
    REPLACE DAY112 WITH INSPDATE
    REPLACE DAY56 WITH INSPDATE
    REPLACE DAY28 WITH INSPDATE
    REPLACE DAY14 WITH INSPDATE
    REPLACE DAY7 WITH INSPDATE
  ELSE
    IF TYPEINSP = 56
      REPLACE DAY56 WITH INSPDATE
      REPLACE DAY28 WITH INSPDATE
      REPLACE DAY14 WITH INSPDATE
      REPLACE DAY7 WITH INSPDATE
    ELSE
      IF TYPEINSP = 28
        REPLACE DAY28 WITH INSPDATE
        REPLACE DAY14 WITH INSPDATE
        REPLACE DAY7 WITH INSPDATE
      ELSE
        IF TYPEINSP = 14
          REPLACE DAY14 WITH INSPDATE
          REPLACE DAY7 WITH INSPDATE
        ELSE
          IF TYPEINSP = 7
            REPLACE DAY7 WITH INSPDATE
          ENDIF
        ENDIF
      ENDIF
    ENDIF
  ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
ENDOF
RETURN

```

```

***** DICTNARY CMD *****
*****
* AUTHOR: LCDR BARNETT *
* PURPOSE: This program module is for the user to define *
* data items, terms, programs, and other modules. *
* INPUT FILES: DDS *
* OUTPUT FILES: None *
* ROUTINES CALLED: None *
* ROUTINES CALLED BY: START *
* LOCAL VARIABLES: ANSWER and MEMDATA *
* CONTROL POINTS: Variable ANSWER controls entry to this *
* module. *
*****

```

```

USE DDS
INDEX ON WORD TO WORD

```

```

***** Allows user to exit program *****

```

```

DO WHILE T
  ERASE
  ? 'IF YOU DO NOT KNOW THE MEANING OF A WORD THIS IS
  THE PLACE'
  ? 'DO YOU NEED TO LOOK UP A WORD? (Y/N)'
  ACCEPT TO ANSWER
  IF $(ANSWER,1,1) <> "Y"
  RETURN
ENDIF

```

```

***** User inputs word of unknown meaning *****

```

```

USE DDS INDEX WORD
STORE " " TO MEMDATA
@ 5,5 SAY "ENTER THE WORD YOU NEED" GET MEMDATA
READ
FIND &MEMDATA
IF # <> 0
  USE DDS
  @ 20,1 SAY "YOUR WORD WITH ITS SYNONYM(S),
  DEFINITION,; PROGRAMS WHERE"
  @ 21,1 SAY "IT IS USED AND FILE WHERE IT IS FORMATTED"
  ?
  LIST FOR WORD = MEMDATA OFF WORD
  LIST FOR WORD = MEMDATA OFF SYNONYM
  LIST FOR WORD = MEMDATA OFF DEFINITION
  LIST FOR WORD = MEMDATA OFF PROGRAMS
  LIST FOR WORD = MEMDATA OFF FILE
  WAIT TO CONTINUE
ELSE
  @ 10,5 SAY "SORRY, THAT IS NOT IN MY MEMORY"
  @ 11,5 SAY "TRY AGAIN, PRESS ANY KEY TO CONTINUE"
  WAIT TO CONTINUE
ENDIF
ENDDO

```

SM/MS File Structure

USE PHASEINSP  
 LIST STRUCTURE  
 STRUCTURE FOR FILE: B:PHASEINS.DBF  
 NUMBER OF RECORDS: 00005  
 DATE OF LAST UPDATE: 01/01/80  
 PRIMARY USE DATABASE

FLD	NAME	TYPE	WIDTH	DEC
001	BUNO	C	006	
002	TYPE:COMP	C	001	
003	BASETIME	N	007	001
** TOTAL **			00015	

USE AIRCRAFT  
 LIST STRUCTURE  
 STRUCTURE FOR FILE: B:AIRCRAFT.DBF  
 NUMBER OF RECORDS: 00005  
 DATE OF LAST UPDATE: 01/01/80  
 PRIMARY USE DATABASE

FLD	NAME	TYPE	WIDTH	DEC
001	BUNO	C	006	
002	ACPTDATE	N	004	
003	FLIGHTHRS	N	007	001
004	ENGLSERNO	C	006	
005	ENG2SERNO	C	006	
** TOTAL **			00030	

USE ENGINES  
 LIST STRUCTURE  
 STRUCTURE FOR FILE: B:ENGINES.DBF  
 NUMBER OF RECORDS: 00014  
 DATE OF LAST UPDATE: 01/01/80  
 PRIMARY USE DATABASE

FLD	NAME	TYPE	WIDTH	DEC
001	SERNO	C	006	
002	POSITION	C	001	
003	OVHAULTIME	N	007	001
004	INSTALTIME	N	007	001
005	CURRENTIME	N	007	001
** TOTAL **			00029	

```

USE SPECIALINSP
LIST STRUCTURE
STRUCTURE FOR FILE:  B:SPECIALI.DBF
NUMBER OF RECORDS:  00005
DATE OF LAST UPDATE: 01/01/80
SECONDARY USE DATABASE
FLD      NAME      TYPE WIDTH  DEC
001     BUNO      C      006
002     DAY7      N      004
003     DAY14     N      004
004     DAY28     N      004
005     DAY56     N      004
006     DAY112    N      004
** TOTAL **                00027

```

```

USE DDS
LIST STRUCTURE
STRUCTURE FOR FILE:  B:DDS      .DBF
NUMBER OF RECORDS:  00073
DATE OF LAST UPDATE: 01/01/80
SECONDARY USE DATABASE
FLD      NAME      TYPE WIDTH  DEC
001     WORD      C      020
002     SYNONYM   C      050
003     DEFINITION C      060
004     PROGRAMS  C      050
005     FILE      C      015
** TOTAL **                00196

```

## SM/MS Examples

### PROGRAM MAIN MENU SELECTION

This computer program accesses the aircraft inspection and component database. You can access or update the database by selecting one of the following options. Just type the number corresponding to your selection.

1. ACCEPT AN AIRCRAFT
2. TRANSFER AN AIRCRAFT
3. ADD FLIGHT TIME TO AN AIRCRAFT
4. UPDATE A COMPLETED SPECIAL INSPECTION
5. PRINT A REPORT OF UPCOMING AIRCRAFT INSPECTIONS
6. USE THE DATA DICTIONARY
7. To TERMINATE this session enter "7"

DO YOU WISH TO ENTER A NEW AIRCRAFT RECORD? (Y/N)

:Y

ENTER AIRCRAFT BUNO:153009:

BUNO :153009:  
ACPTDATE :5160:  
FLIGHTHRS :11500.0:  
ENGLSERNO :900001:  
ENG2SERNO :900002:

THE FOLLOWING SCREENS UPDATE ENGINE DATA  
TYPE ANY KEY TO CONTINUE

Update Flight hours flown

HOW MANY FLIGHT HOURS WERE FLOWN?: 3.0:

AIRCRAFT HAS 22.0 HOURS UNTIL THE NEXT INSPECTION,  
WHICH IS A/AN "D" PHASE INSPECTION.

TYPE ANY KEY TO CONTINUE

WAITING

FLIGHT TIME FIELDS HAVE BEEN UPDATED FOR THE AIRCRAFT,  
ENGINES, TRANSMISSION, AND ROTORHEAD.

PRESS ANY KEY TO CONTINUE.

WAITING

AD-A161 254

AN EVALUATION AND IMPLEMENTATION STUDY OF THE ZENITH  
120 MICROCOMPUTER IN WEST COAST FLEET COMMANDS(U) NAVAL  
POSTGRADUATE SCHOOL MONTEREY CA G A BARNETT SEP 85

2/2

UNCLASSIFIED

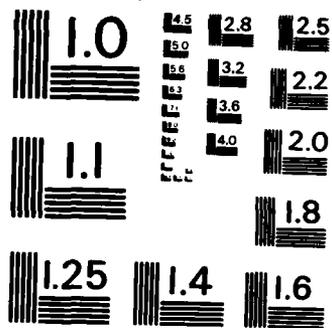
F/G 9/2

NL

END

FORM

DTIC



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

AIRCRAFT Special Inspection Example

DO YOU NEED TO UPDATE AN AIRCRAFT SPECIAL INSPECTION? (Y/N)  
:Y

ENTER THE AIRCRAFT BUNO:153009:

ENTER DATE INSPECTION COMPLETED:5160:

ENTER TYPE INSP COMPLETED: 7:

THE NEXT SCREEN UPDATES THE SPECIAL INSP FILE  
PRESS ANY KEY TO CONTINUE

RECORD # 00005  
BUNO :153009:  
DAY7 :5155:  
DAY14 :5155:  
DAY28 :5155:  
DAY56 :5155:  
DAY112 :5155:

Aircraft Report Example

DO YOU WISH A REPORT OF AIRCRAFT DUE FOR SPECIAL INSP (Y/N)  
:Y

ENTER THE REQUESTED PERIOD IN DAYS: 20:

AIRCRAFT SPECIAL INSPECTION REPORT  
JULIAN DATE IS 5160

THE FOLLOWING IS THE STATUE OF SPECIAL INSPECTIONS FOR  
THE NEXT 20 DAYS

BUNO	TYPE INSP	DUE DATE
153001	112 DAY	5166
153005	112 DAY	5177
153006	112 DAY	5167
153008	014 DAY	5178
153009	014 DAY	5169

Data Dictionary/Directory Example

IF YOU DO NOT KNOW THE MEANING OF A WORD THIS IS THE PLACE  
DO YOU NEED TO LOOK UP A WORD? (Y/N)

ENTER THE WORD YOU NEED: JULIAN :

YOUR WORD WITH ITS SYNONYM(S), DEFINITION, PROGRAMS WHERE  
IT IS USED AND FILE WHERE IT IS FORMATTED

JULIAN  
JULDATE, MENJUL  
THE CALENDAR BASED ON AUGUSTUS JULIUS  
START, EDIT  
N/A  
WAITING

## LIST OF REFERENCES

1. Work Unit Summaries, Defense Technical Information Center, DLA, Alexandria, Virginia, April 1985.
2. "America's Offices Enter A New Age," Nation's Business, V. 69, N. 7, p. 48-54, July 1981.
3. Noyce, R. N., "Microelectronics," Scientific American, V. 237, N. 3, p. 65, September 1977.
4. Senn, James A., Information Systems in Management, Wadsworth Publishing Company, 1978.
5. Hendricks, Daniel E., Human Engineering Guidelines For Management Information Systems, U.S. Army Material Development and Readiness Command, Aberdeen, Maryland, 1983.
6. Sullivan, J., Telephone Interview, 12 June, 1985 Navy Regional Data Automation Center, Norfolk, Virginia.
7. Leo, J., "Coping With The Computer," Discover 1980, pp 95-97, December 1, 1980.
8. Barnett, G. A., Can Computers Save the Squadron?, paper presented for MN 4125, Naval Postgraduate School, Monterey, California, December 1984.
9. "Should Software Vendors Start Thinking Micro?", Interface, V. 6, N. 2, p. 22, Summer 1981.
10. Ogden, C. A., Software Design for Microcomputers, Prentice-Hall, 1978.
11. Dinestein, Nelson T., dBASE II for the Programmer, Scott, Foresman and Company, 1984.
12. Zenith Data Systems, Z-100 PC Series Operations Manual, 1984.
13. Brandt, Richard, Telephone Interview, 8 June, 1985, University of Utah, Salt Lake City, Utah.
14. Durell, William, "Disorder to Discipline," Journal of Systems Management, pp. 12-19, May 1983.
15. Martin, James, Computer Data-base Organization, Englewood Cliffs, New Jersey: Prentice Hall, Inc., 1981.

16. Kroenke, David, Database Processing, Second Edition, Chicago, Illinois: Science Research Associates, 1983.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5100	2
2. Department Chairman, Code 54 Department of Administrative Sciences Naval Postgraduate School Monterey, California 93943-5100	1
3. Curricular Officer, Code 37 Computer Technology Department Naval Postgraduate School Monterey, California 93943-5100	1
4. LCDR S. Vance VF-126 NAS Miramar San Diego, California 92149	1
5. Commanding Officer Attn: Lt. L. MacAlpine Nardac San Diego, California 92135	1
6. Professor D. Dolk, Code 54Dk Naval Postgraduate School Monterey, California 93943-5100	1
7. LCDR P. Fischbeck, Code 55Fb Naval Postgraduate School Monterey, California 93943-5100	1
8. LCDR B. Barnett VF-21 Nas Miramar San Diego, California 92149	1
9. Mr. Sam Anderson, Code 20 COMNAVAIRPAC U.S. Naval Air Station, North Island San Diego, California 92135	1
10. LT. Brad Vannoy HC-3 NAS North Island San Diego, California 92135	1

11. Defense Technical Information Center  
Cameron Station  
Alexandria, Virginia 22304-6145

2

**END**

**FILMED**

**12-85**

**DTIC**