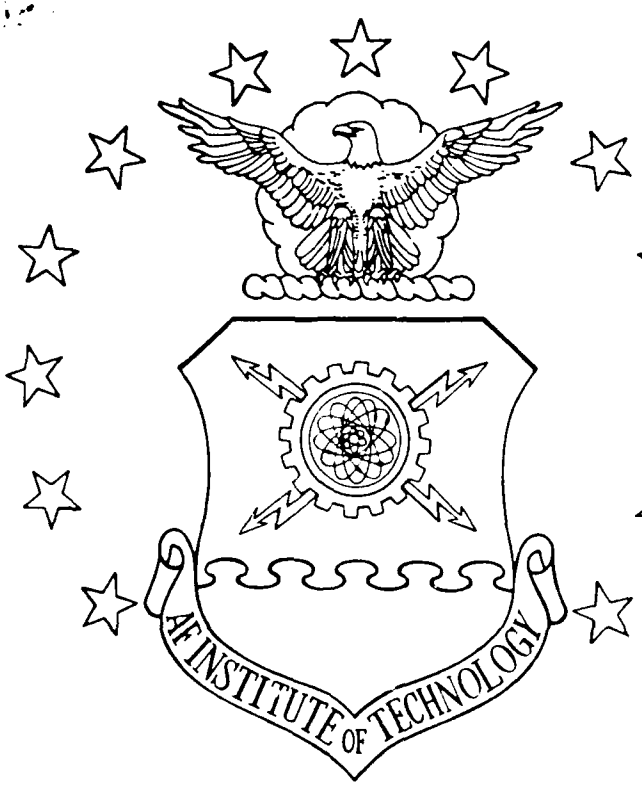

AUTOMATING THE AIR FORCE BASE-LEVEL REPORT OF DISCREPANCY PROGRAM: AN APPLICATION OF DATABASE MANAGEMENT TECHNIQUES

THESIS

James L. Johnson
Captain, USAF

AFIT/GLM/LSM/89S-33

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GLM/LSM/89S-33

AUTOMATING THE AIR FORCE BASE-LEVEL REPORT OF DISCREPANCY PROGRAM: AN APPLICATION OF DATABASE MANAGEMENT TECHNIQUES

THESIS

James L. Johnson
Captain, USAF

AFIT/GLM/LSM/89S-33

The contents of the document are technically accurate, and no sensitive items, detrimental ideas, or deleterious information is contained therein. Furthermore, the views expressed in the document are those of the author and do not necessarily reflect the views of the School of Systems and Logistics, the Air University, the United States Air Force, or the Department of Defense.

AFIT/GLM/LSM/89S-33

AUTOMATING THE AIR FORCE BASE-LEVEL REPORT OF DISCREPANCY PROGRAM: AN APPLICATION OF DATABASE MANAGEMENT TECHNIQUES

THESIS

Presented to the Faculty of the School of Logistics

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Logistics Management

James L. Johnson, B.S., M.S.

Captain, USAF

September 1989

Acknowledgements

I wish to extend my sincerest appreciation to Captain John E. Sullivan III, my thesis advisor, for his patience, understanding, and encouragement in conducting this study. I would also like to thank Dr. Charles Fenno for his assistance in helping me to learn and master many of the documentation techniques essential to conducting a research study. Thanks also to Carolyn Thompson and Sergeant William T. Jackson of the 2750th Logistics Squadron for helping me to gain in-sight into the functions of the Report of Discrepancy program.

Table of Contents

List of Figures

List of Programs

Abstract

The purpose of this study was to apply principles of database management to the Air Force Report of Discrepancy (ROD) Program. The overall goal was to reduce the number of manhours required to manage the program and to improve the reliability of information. In conducting this study, the researcher asked and answered four research questions. Each question constituted one of four steps in the development of this study. The four steps were to (1) evaluate the current system, (2) identify aspects of the current system that could be improved through automation, (3) choose a database management software (DBMS) package to automate those aspects identified in step 2, and (4) structure and implement a database application.

The study resulted in the development of a database application designed to automate aspects of the ROD program. This program could have universal application to all Air Force base supply organizations. Validation and/or distribution of the programs or findings will be managed by the Logistics Management Center at Gunter AFB, Alabama.

AUTOMATING THE AIR FORCE BASE-LEVEL REPORT OF DISCREPANCY PROGRAM: AN APPLICATION OF DATABASE MANAGEMENT TECHNIQUES

I. Introduction

Background

In August 1978, the House Appropriations Committee charged the military services were wasting millions of dollars each year due to poor supply procedures (Smith and Saengaram, 1980:1). The committee recommended that $155 million be cut from the services' budgets. The Air Force's share of the proposed budget cut was $50 million (Smith and Saengaram, 1980:1). As a result of the proposed budget cut, the Air Force implemented a number of programs aimed at eliminating waste within its supply system. Following a thorough review of supply procedures, shipment discrepancies, received by base supply activities from various vendors, were closely scrutinized. Although there were many types of shipments discrepancies, only those involving shortages were tracked (Smith and Saengaram, 1980:2). These discrepancies were easily discovered/identified, and the cost could be easily calculated (Smith and Saengaram, 1980:2). Subsequently, the Air Force expanded the program to include discrepancies

which were more difficult to account for, but yet have an impact upon supply operations.

General Issue

The Report of Discrepancy (ROD) Program is a vital link in the Air Force's effort to eliminate waste within its supply system. The ROD program provides a means of reporting shipment discrepancies to the responsible activity and provides the basis for corrective action to prevent recurrences (HQ USAF, 1987:Ch 5, 51). While the number of shipment discrepancies has steadily increased, the manual, repetitive procedures for managing the program have remained virtually unchanged. As a result, the current program may no longer be an effective tool for managing discrepancies.

Justification

This research was sponsored by the Logistics Management Center located at Gunter Air Force Base, Alabama. The automation of the manual, inefficient procedures in the Air Force Report of Discrepancy (ROD) program is long overdue (Kendall, 1988). In the past, a lot of attention has been placed on establishing procedures to track each shipment discrepancy reported. However, no efforts were made to review those procedures to determine if less time consuming and more efficient means could be found to eliminate the repetitive functions. Most of the procedures governing the ROD program were established years ago and have little, if

any, computerized support (Kendall, 1988). Therefore, it was reasonable to conclude:

1. The management of the ROD program could be significantly enhanced through automation.

2. Automation would result in manpower savings and improved efficiency of assigned personnel.

Problem Statement

The ROD program is a manual system requiring a clerk to maintain file folders containing information for each shipment discrepancy reported. Whenever information is required, the clerk must rummage through files to locate a particular report. If and when the report is found, pencil notations must be made concerning follow-up actions or recurring problems. As a result, the reports are often smudged or illegible. In addition, trend analysis is a manual process, making the task of identifying and implementing long-term corrective actions more difficult.

Research Objective

The purpose of this research was to apply principles of database management to the management of the reports of discrepancies. The overall goal was to reduce the number of manhours required to manage the program and to improve the reliability of information. Once the areas requiring automation were identified, a microcomputer application was

developed to eliminate the manual, repetitive work. Reasons for automation were the following.

1. Those interviewed believed the current procedures were time consuming and highly inefficient.

2. The researcher and those interviewed found automation to be a reasonable approach to improving the current system (Bailey, 1988:5).

3. The researcher found automation to be practical and well within time and resource constraints (Bailey, 1988:5).

Research Questions

In conducting this research, four research questions were investigated.

1. What is the current system?
2. What aspects of the current system can be improved through automation?
3. Which database management software (DBMS) package is most appropriate?
4. How can the DBMS be structured and implemented?

Scope

For the purpose of this study, the researcher examined only the procedures affecting base level supply operations within the Air Force. Implementation of all application programs designed and written by the researcher was accomplished at Wright-Patterson Air Force Base (WPAFB) Base

Supply activity. The researcher chose this organization because of its proximity to the Air Force Institute of Technology (AFIT). The programs developed during this research could have universal application to all Air Force base supply organizations. Validation and/or distribution of the program or findings will be managed by the Logistics Management Center at Gunter AFB, Alabama.

Limitation

The applications programs developed during this research were designed for use on stand-alone personal computers. Furthermore, the application programs were designed to meet the various information needs of several sections within the local supply activity. However, until automated interfaces between the various sections are established, a fully integrated Management Information System within the local supply activity will not be feasible.

Assumptions

The following assumptions were made concerning this research effort.

1. Those interviewed by the researcher were knowledgeable of the procedures involved in managing the ROD program, as it applies to their work area, and were considered experts.

2. Personnel interviewed could provide specific requirements for application programs to automate current manual procedures (Bailey, 1988:7).

3. Users of the programs, developed by the researcher, would have access to an IBM compatible computer system, and the database software needed to run the programs (Bailey, 1988:8).

Organization of Report

This research report consists of five chapters. Chapter I addresses the research problem, the research questions, the limitations of the research, and assumptions made during this study. Chapter II provides an introduction to databases, DBMS concepts, and the Air Force Report of Discrepancy (ROD) program. Chapter III describes the methodology of the research. Chapter IV reviews the findings of the research and provides answers to the research questions. Chapter V provides conclusions and provides recommendations for further research. The appendices provide information on the database files and application programs developed during this research.

Definitions

Key terms used in this paper are defined as follows.

1. Discrepancy Report. "A report of the receipt of an item which is deficient in some aspect and which is

officially reported on a Standard Form 364, Report of Discrepancy (ROD)" (Smith and Saengaram, 1980:4).

2. Shipment Discrepancy. "A requisition which is received and is found to contain a shortage or an overage, erroneous materials, hidden condition which affects its usefulness, missing or incomplete technical data markings, missing supply documentation, or a misdirected shipment which can reasonably be assumed to be the fault of the shipping activity" (Smith and Saengaram, 1980:4-5).

3. Receiving Activity. The activity in receipt of a discrepant shipment which subsequently files a report of discrepancy with the shipping activity.

4. Report Of Discrepancy Monitor. The person who administers the ROD program at the base level supply activity.

5. Shipping Activity. The activity from which the discrepant shipment originated.

6. Shortage. "When the quantity received is less than the quantity ordered or shown on the shipping document" (HQ USAF, 1986:Encl 1, 1).

7. Overage. "When the quantity received is greater than that ordered or shown on the shipping document" (HQ USAF, 1986: Encl 1, 1).

8. Information Attributes. Detailed information about an entity. For people, information attributes are things

such as name, rank, office symbol, social security number, etc. (Hodge and others, 1984:204).

9. Entity. A person, place, or thing (Pratt, 1988:9).

10. Attribute. "A property of an entity" (Pratt, 1988:9).

11. Database Management System (DBMS). "A program or collection of programs whose function is to manage a database on behalf of the people who use it" (Pratt, 1988:6).

II. Literature Review

Overview

In conducting this study, the researcher encountered a problem in locating background information concerning the Report of Discrepancy (ROD) program. For this reason, many important aspects of the program, such as when and why the program was instituted, were not addressed. Additionally, information concerning the way civilian institutions handle shipment discrepancies was equally scarce, and, when found, the procedures for reporting the discrepancies were not clearly documented. This chapter provides an introduction to databases, database management concepts, and the Air Force Report of Discrepancy (ROD) program.

Databases

Entities are persons, places, and things. Data are simply information concerning entities. This information or data consists of three parts: the information attribute or context, the data value, and the data attribute. An example of this is as follows:

Entity	Information Attribute	Data Value	Data Attribute
Automobile	Color	Brown	10 alpha characters
	Make	Chevrolet	20 alpha characters
	Type	4-door sedan	20 alpha characters
	Cost	$12,000	6 numeric characters

Data and attributes are two important terms in database design. Data are simply the information that goes into the database. Attributes are the types of data that make up the database. In a database, data are stored in data files, which are structures used to store data about some entity. According to Pratt, a data file is similar to an ordinary paper file kept in a filing cabinet in that the two may contain the same information (Pratt, 1988:10-11). However, a database is much more than a file. While a paper file may contain information about a particular entity and that entity's attributes, a database contains information about a number of entities, their attributes, and any relationships that exist with other entities (Pratt, 1988:10-11).

The data value and data attribute also play important roles in database design (Hodge and others, 1984:204). The data value is the actual values assigned and entered into the database concerning some entity. The data attribute is the mechanical function of the database that addresses how the data are physically stored within the database. The main purpose of the database management system and its accompanying applications software is to combine information about the entity, its data attributes and values, and to provide the information to the user in some useful format (Hodge and others, 1984:204).

Database Management Concepts

Databases provide six functions which allow the user to manipulate data more efficiently and effectively than ever before (Bailey, 1988:21). Functionally, databases (1) allow efficient entry and storage of data, (2) provide error and consistency checking functions as well as integrity constraints, which ensure certain conditions are before different phases of processing are completed, (3) provide some degree of protection against unauthorized access to data, (4) minimize the potential impact of software errors may have on the data, (5) provide more efficient use of internal storage space by allowing the data and the programs that manipulate the data to be stored separately, and (6) store data in a central location, which makes update and access by application programs quicker (Bailey, 1988:21-22).

To carry out these functions, databases perform six basic tasks which affect the contents of the database and provides the user with the required information (Bailey, 1988:22). "These tasks include (1) adding data to the database, (2) editing data already in the database, (3) deleting data from the database, (4) sorting the data according to the user's needs, (5) searching for information required by the user, and (6) printing information contained in the database" (Bailey, 1988:22).

Database management systems (DBMS) are categorized by the approach the DBMS takes in storing and manipulating

information about a variety of entities, the attributes of the entities, and the relationships between the entities (Pratt, 1988:19). "The vast majority of DBMS's follow one of three model models: the relational model, the network model, or the hierarchical model" (Pratt, 1988:19).

A relational model database is a data model in which the structure appears to be a collection of tables. Each table consists of rows (horizontal) which are called records and columns (vertical) which are called fields. Each record within a table is linked to the other records in that table by the fact they contain the same fields. Similarly, records stored in different tables are linked. According to Pratt, the strengths of a relational model systems are twofold (Pratt, 1988:28-29). First, they are simple and easy to use. The user does not need a high level of expertise concerning database design to access and use relational type databases, unlike the other models. Second, relational model systems offer a higher degree of data independence than the other models. "Data independence, which is the ability to make changes in the database structure without having to make changes in the programs that access the database, is one of the advantages of all types of DBMS's" (Pratt, 1988:28-29). However, relational models offer a higher degree than the other two models (Pratt, 1988:28-29).

Network model databases differ significantly from relational models. While relational models appear to be a collection of tables, network models appear to be a combination of record types (entities), fields within the record types (attributes), and "explicit" relationships between the record types (Pratt, 1988:29). The network model differs from the relational model in that the relationships within the network model are "explicit" whereas the relational model's relationships are "implicit"--derived by matching columns in various tables (Pratt, 1988:29).

According to Pratt, network database models present a number of advantages and disadvantages (Pratt, 1988:32-33). The main advantage of the network model is its efficiency, which enables it to handle large databases with large amounts of activity. Network models also provide integrity constraints that the other models lack. The main disadvantage of the network model is that it is more difficult to use than the other models. To obtain information from a network model, the user must access database files in the proper sequence. This requires the user to be familiar with the underlying database structure in order to obtain the desired output in an efficient manner. Because of the difficulty in using network models, making changes to the database is also difficult. Unlike relational database models, which allow the user to update

the database without affecting the programs that access the data, network model require that all programs that access the database also be updated (Pratt, 1988:32-33). This quality makes the network model highly complex--which, in turn, makes the network model more expensive in terms of disk space, computer time, and programmer time than the other models (Bailey, 1988:23). For example, consider the diagram in Figure 1.

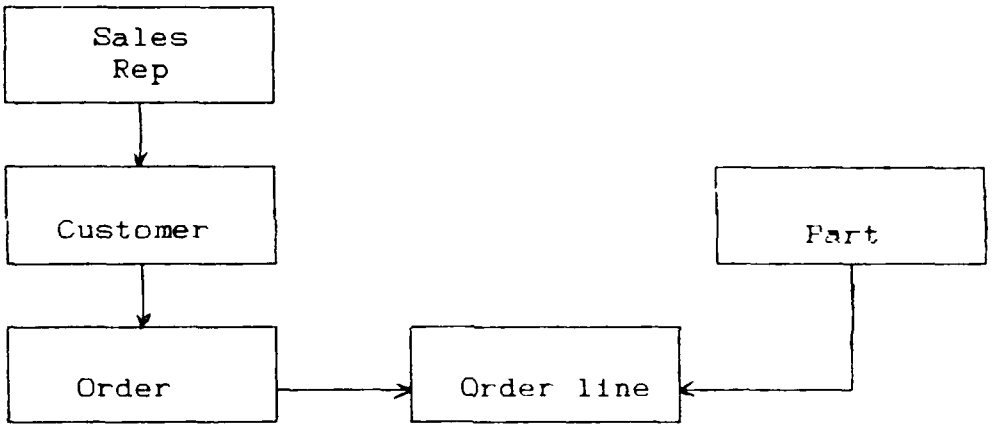

Figure 1. The Network Model (Pratt, 1988:30)

In Figure 1, the rectangles represent the record types in a database, and the arrows represent the relationships between the record types (Pratt, 1988:30). In a network data model, "the arrow goes from the 'one' part of the relationship, called the owner, to the 'many' part, called the member (Pratt, 1988:30). In the relationship shown by the arrow from sales representative (rep) to customer, sales rep is the owner and customer is the member. Since each sales rep may have more than one customer, but yet each

customer can have only one sales rep, a one-to-many relationship exists (Pratt, 1988:30).

A hierarchical model may be thought of as a family tree consisting of mothers and sons (Bailey, 1988:23). Each son can have only one mother, yet each mother can have more than one son. In a hierarchial model, a "child" record can be linked to only one "parent" record, but a "parent" record can be linked to many "child" records (Bailey, 1988:23). For example, consider the diagram shown in Figure 2.

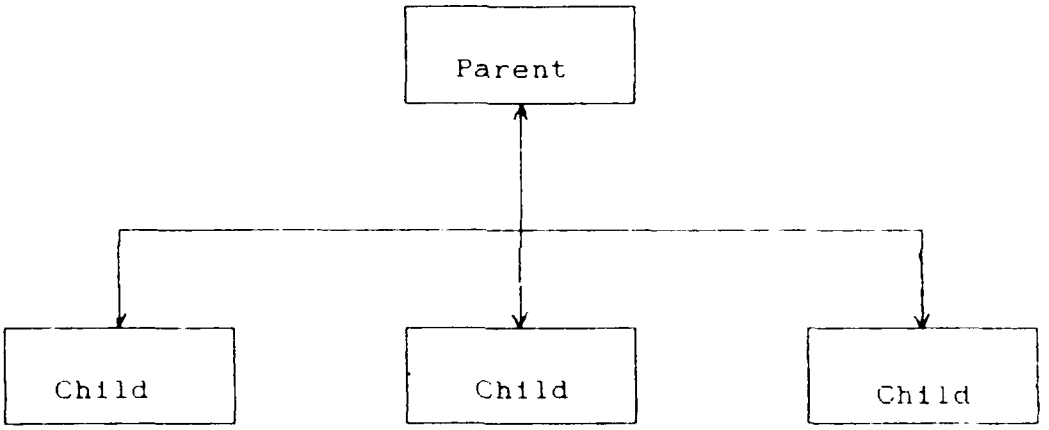

Figure 2. Hierarchical Data Model

The hierarchical model possesses the same advantages and disadvantages as the network model (Pratt, 1988:34). Pratt suggests that hierarchical and network models are "comparable" with the only distinction being that if the underlying database design is tree-like, then a hierarchical model would be required (Pratt, 1988:34). Bailey has presented a different view. Bailey suggests the major disadvantage of the hierarchical model is that while it

allows for vertical relationships between the "parent" and "child" records, it does not allow for relationships between the "child" records, which limits the hierarchical model's usefulness (Bailey, 1988:23).

There are a number of differences in the approaches the hierarchical and network data models use in storing and manipulating data. However, the most prominent difference is that the network model allows for "parent-child" and "child-child" relationships while the hierarchical model allows only "parent-child" relationships. Additionally, network models allow a "child" record to have more than one "parent", unlike the hierarchical model (Bailey, 1988:23). As a result, network models are more complex which make them more expensive in terms of disk space, computer time, and programmer time than the other models (Bailey, 1988:23).

Today with the aid of DBMS software, users often build databases and write programs to perform various functions on data. While building databases and writing programs are time consuming activities requiring considerable knowledge and skills, the DBMS software available today allow individuals with little or no formal training to accomplish these tasks (Bailey, 1988:3). However, there are still many areas yet to benefit from application of DBMS software which would eliminate repetitive tasks. One such area that may benefit from application of database management techniques is the Air Force Report of Discrepancy (ROD) Program.

Purpose of the ROD program

The purpose of the ROD Program is to provide a means of reporting shipment discrepancies to the responsible activity and to provide the basis for corrective action to prevent recurrences (HQ USAF, 1988:Ch 5, 51). All types of discrepancies are reported using a Standard Form 364, Report of Discrepancy (ROD). According to Air Force Manual (AFM) 67-1, Volume I, Part One, Chapter 5, Section D, Reporting of Item and Packaging Discrepancies, the SF Form 364 serves the following functions:

1. Supports adjustments of property and financial inventory accounting records.

2. Provides information which serves as a basis for claims against contractors, notification to shippers, and disposition instructions.

3. Provides visibility of preservation, packing, marking, and unitized load discrepancies with required corrective actions or recommended improvements.

4. Provides information for management evaluations. When used correctly, the SF Form 364 can help improve supply operations, and aid both the receiving and shipping activity in determining problem areas (Smith and Saengaram, 1980:2).

The receiving activity files a Report of Discrepancy (ROD) to record item or packaging discrepancies caused by the shipping activity (HQ USAF, 1988:Ch 9, 343). The ROD serves two basic purposes:

1. The ROD notifies the responsible shipping activity that a discrepancy exists, aids in analyzing and correcting discrepancies, and helps prevent recurrences (HQ USAF, 1988:Ch 9, 343).

2. The ROD serves as a supporting document for inventory accounting and financial adjustments. "A copy of the completed report is used to support adjustments to the reporting and shipping organizations' accountable inventory and financial accounting records" (HQ USAF, 1988:Ch 9, 343).

Reporting Activity ROD Submission Actions

The reporting activity has two basic responsibilities within the ROD program (HQ USAF, 1988:Ch 9, 343). First, it prepares a ROD any time an item is received as a result of a discrepancy attributable to the shipping activity. The discrepancies must meet at least one of the conditions specified in Air Force Regulation 400-54, Reporting of Item and Packaging Discrepancies. Second, the reporting activity maintains a suspense file for each discrepancy awaiting a reply. Once the reply is received, the suspense copy of the ROD is destroyed. If no reply is received after 45 calendar days, a follow-up ROD is forwarded to the shipping activity. A second follow-up is forwarded if no reply is received after 45 days from the initial follow-up. If no reply is received by the third follow-up and 160 days have passed since the initial submission of the ROD, the suspense copy

of the ROD is forwarded to the major command for resolution (HQ USAF, 1988:Ch 9, 343).

Shipping Activities ROD Reply Actions

Shipping activities are required to reply to each ROD submitted by base supply activities within 45 calendar days Each ROD not submitted by the reporting activity within this time limit is rejected by the shipping activity (HQ USAF, 1988:Ch 9, 344). Once the ROD is received, the shipping activity completes the reverse side to the SF Form 364, indicates whether a billing adjustment will be issued, and returns the reply--the original SF Form 364 with details of actions taken--to the reporting activity (HQ USAF, 1988:Ch 9, 344).

Analysis Requirements

For any program to be effective, a system for internal analysis must be in place to ensure the system is operating as designed. The ROD management program is no exception. AFM 67-1, Supply Procedures, requires each activity establish controls to ensure:

1. RODs are reported promptly.

2. RODs, received against the shipping installations, are investigated to determine the cause.

3. Corrective actions are taken to preclude recurrence.

4. Control and follow-up processing is maintained until the discrepancy/claim is resolved.

Additionally, AFM 67-1 requires each organization involved in the discrepancy reporting process prepare periodic summaries. The supply squadron uses these summaries in analyzing the types of errors being made and to take corrective action to prevent recurrences.

Within the base-level supply activity, the Procedures and Analysis (PA) Unit is the focal point of all reports of discrepancy submitted. PA coordinates the actions of all supply activities involved with the ROD program, ensures the responsible sections make follow-ups on outstanding RODs, and distributes replies to RODs to the appropriate section for action. Current procedures require the PA Unit to maintain a suspense file of outstanding RODs. PA uses the suspense file to perform trend and problem analysis of all RODs submitted by activities within the Chief of Supply complex.

Conclusion

The procedures for managing the Report of Discrepancy Program are simple and straight forth. However, the procedures governing the program require manual steps that are time consuming and of questionable efficiency. With the computer technology available today, some effort should be

made to automate the ROD program. Automation of the program would be beneficial to the Air Force in two ways.

1. It would eliminate the need to maintain bulky, time consuming file folders for each discrepancy report.

2. Automation would make the task of gathering information for use in trend and cause analysis a lot easier.

Preliminary research indicated automation of the ROD Program would not require any substantial outlay of funds to purchase new equipment because most of the required equipment--Z248 computer systems or equivalent IBM compatible systems--was already on hand. This being the case, all that remained was to develop a program to automate the manual, repetitive procedures involved in managing the ROD program.

III. Methodology

Overview

The researcher conducted this study in two phases. In phase I, the researcher identified a series of research questions. The aim of the research questions was to gain a working-knowledge of the ROD program and to help the researcher gain in-sight into any potential applications of database management techniques. The second phase involved actually designing the database and applying database management techniques to the ROD program.

Phase I

In conducting this study, the researcher established the following research questions. The methodologies for each varies slightly and will be discussed separately.

1. What is the current system?
2. What aspects of the current system can be improved through automation?
3. Which database management software (DBMS) package is most appropriate?
4. How can the DBMS be structured and implemented?

Research Question One. In addressing this question, the researcher performed a thorough literature review of the existing procedures concerning the ROD program, databases, and DBMS concepts. Literature reviewed included Air Force

manuals, inspection reports, staff assistance visit reports, and books on database management. Additionally, interviews with users in the field were conducted to gain first hand knowledge of the procedures used in managing the ROD program.

Research Question Two. The researcher observed the step-by-step procedures involved in managing the ROD program and with the aid of the ROD monitor at the local base supply identified aspects of the program that required automation.

Research Question Three. The researcher established a set of criteria to choose an appropriate software package. After applying the criteria, the researcher chose the software package that best met the established criteria.

Research Question Four. The researcher approached this question in two steps. First, the researcher considered how the data elements contained on the SF Form 364, Report of Discrepancy, were related to each other, if at all. In this step, the researcher sought to establish whether or not a one-to-many relationship or even a "parent" - "child" relationship existed. This determination was key in determining an appropriate database model. Second, the researcher determined which type of database model to use in building the database and proceeded to build the database.

Phase II

In this phase, the researcher followed a series of steps based on Jones's five steps in designing application programs. The steps were:

1. Define the problem
2. Design the program
3. Write the program
4. Verify the program
5. Document the program (Jones, 1987:215).

The first step was simply to define the problem the program was designed to solve. The problem could be as simple as the need to automate some procedure. In step 2, the researcher sought to develop a series of small programs or modules that would collectively form a larger program. In step 3, the researcher wrote the program code for the modules. The researcher verified the program by correcting the program and by consulting the users to ensure the program met their needs. In step 5, the researcher provided documentation by placing comments within the program code, which explained the functions of each module and, occasionally, the purpose of a line of program code.

The application program was implemented at WPAFB. The programs have been turned over to the Air Force Logistics Management Center (AFLMC) for further testing and distribution throughout the Air Force. AFLMC has assumed all responsibility for maintaining the programs.

IV. Findings

Overview

This study was accomplished in two phases. In phase I, the researcher established and answered a series of research questions. The aim of the research questions was to gain first-hand knowledge of the ROD program and in-sight into any potential applications of database management to the current system. The second phase involved the design and application of database management techniques to the ROD program in order to reduce the number of manhours required to manage the program and to improve the reliability of information. This chapter addresses the procedures used in conducting both phases of the research study.

Phase I

In conducting this study, the researcher established four research questions. The specific methodology employed in answering each research question varies slightly, so each will be discussed separately.

Research Question One. What is the current system?

In addressing this question, the researcher performed a thorough literature review of the existing procedures concerning the ROD program, databases, and DBMS concepts. Literature reviewed included Air Force manuals, inspections reports, staff assistance visit reports, and books on

database management. Additionally, interviews with users in the field were conducted to gain first hand knowledge of the procedures used in managing the ROD program. The findings from the literature review are addressed in Chapter II of this research study.

Research Question Two. What aspects of the current system can be improved through automation?

In addressing this question, the researcher observed the step-by-step procedures involved in maintaining the suspense files and in making the required follow-ups. Specific aspects requiring automation identified during the interviews were as follows:

1. The filing system.

2. The manual steps required to determine when follow-up action should be taken.

3. The steps to provide analysis of the timeliness of follow-ups.

In addition to providing information concerning the aspects needing automation, those interviewed provided input as to the built-in capabilities the program should possess. The capabilities identified were as follows:

1. Develop a menu-driven program to store information contained on the SF Form 364 internally, thereby eliminating the need to maintain hard copies of the RODs (Bates, 1988; Thompson, 1988).

2. Establish a simple reference system to allow easy update of existing files, addition of new files, and deletion of old files (Jackson, 1989).

3. Create an internal system to access different aspects of information contained in a ROD file, such as national stock number, date forwarded, or price (Wright, 1989).

4. Develop a capability to print information directly onto blank SF Form 364 (Jackson, 1989).

Research Question Three. Which database management software (DBMS) package is most appropriate?

In a similar study, Captain Bailey identified several criteria for selecting the most appropriate software package. Of his six criteria, the researcher chose only three because they were most applicable to this research project.

1. The software package had to be designed for use on a stand-alone microcomputer. This was essential because all programs would be developed using a microcomputer (Bailey, 1988:47).

2. The database management system (DBMS) had to allow users to develop their own programs. This capability would enable the user to develop a system which was easy to use and tailor-made to the user's specifications (Bailey, 1988:47).

3. The DBMS had to be available to all users of the ROD program. Ideally, the software would already be on hand in the user's offices or easily obtained (Bailey, 1988:47).

Of the many DBMS packages available on the market, the two most common to the Air Force are dBASE III Plus and Enable (Kendall, 1988). Of these two packages, dBASE III Plus has the strongest support within the Air Force. The Air Force Logistics Management Center (AFLMC)--the activity that sponsored this research--and the Air Force Standard Systems Center routinely use dBASE III Plus to build small stand-alone management information systems. Additionally, AFLMC has the capability to compile dBASE III Plus programs (Kendall, 1988). Compiling results in programs running faster using dBASE III Plus and allows users to legally share a dBASE III Plus application without purchasing additional software packages. As a result of these factors and AFLMC's recommendation, the researcher chose dBASE III Plus to develop a microcomputer application to automate the program.

Research Question Four. How can the DBMS be structured and implemented?

Structuring the Database. The first step in the development of a database was to determine the database's basic structure. This step required the researcher to determine which data elements or attributes should be contained in the database. The researcher contacted

potential users of the program at the local base supply activity to determine what information would be most useful in managing the ROD program. The researcher and those interviewed agreed the database should, as a minimum, contain as many of the data fields contained on the SF Form 364 as possible to facilitate a print-out function (Jackson, Thompson, and Wright, 1989). The actual structure of the database is contained in the data dictionary located in the appendix A.

The researcher structured the database to allow entry of all data fields on the SF Form 364, Report of Discrepancy. Each SF Form 364 has a distinctive report number to which all the other data fields relate. The researcher established the report number as the primary key, main access vehicle to the individual records, that would be used to load information into the database. The researcher found each report number relates to a number of data fields, but each data field relates to only one report number, which constituted a one-to-many relationship (See Figure 3). Of the three database models discussed in chapter II, the researcher chose to use the hierarchial model because it most closely matched the relationships of the data fields contained on the SF Form 364. For example, consider the diagram in Figure 3. The report number (the "parent"), relates to many data fields (the "child" records), while each of the "child" records related to only one "parent" or

report number. In the database, the report number serves as the "parent" for the data fields shown on the SF Form 364. For purposes of illustration, only four of the 25 data fields are shown.

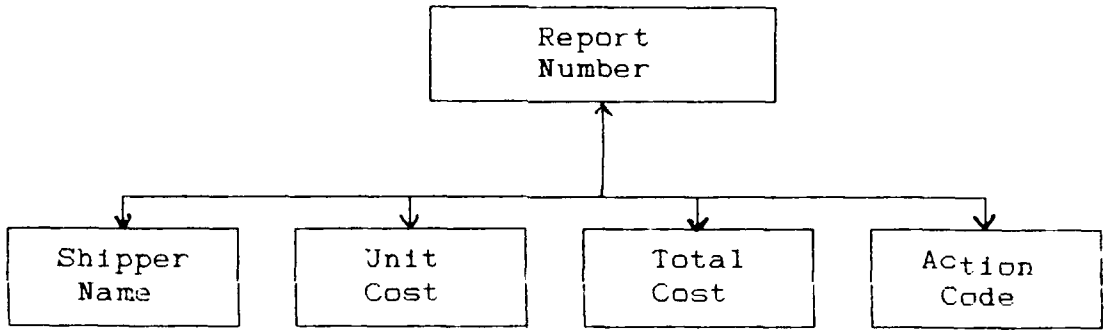

Figure 3. Database Structure

Phase II

During this phase, the researcher developed the computer applications to automate aspects of the ROD program. In developing the dBASE III Plus applications, the researcher followed Jones's five steps in designing programs. The steps were:

1. Define the problem
2. Design the program
3. Write the program
4. Verify the program
5. Document the program (Jones, 1987:215).

Defining the Problem. In this step, the researcher defined the problem the program was designed to solve. The researcher viewed the underlying problem as being the Report

of Discrepancy (ROD) program's manual, repetitive procedures used in managing the program. Furthermore, the researcher concluded that the problem could be solved through automation.

Designing the Program. The best programs are in fact a collection of smaller programs--called modules--that perform specific functions (Jones, 1987:217). Smaller programs divide the larger programing task into manageable sections which not only make it easier to organize the layout of the program, but also make it easier to locate errors in the program (Jones, 1987:217). In designing the program, the researcher outlined four smaller tasks or modules that would:

1. Allow new records to be added
2. Allow existing records to be changed
3. Delete records when no longer required
4. Produce reports when required.

After outlining the basic functions of the DBMS, the researcher contacted potential users in the field to verify these modules represented the basic functions that are performed in the day-to-day management of the ROD program. Following the verification, the researcher proceeded to write the program.

Writing the Program. In this step, the researcher wrote the basic program. The program began with a brief introduction to the program followed by an operations menu

which listed the functions identified during the design process. The applications programs developed during this study are contained in appendix C.

Verifying the Program. During this step, the researcher corrected errors in the program. Users at the local base supply activity were allowed to examine and experiment with the program to ensure the program met their needs. During the review by the users, the researcher made changes as necessary to make the system more responsive to the needs of the users.

Documenting the Program. Documentation was provided by written directions (a user's guide) explaining how the programs operate and by comments placed within the program code explaining the logic of various sections of the programs. The purpose of the documentation was to assist the reader in understanding the programs created during this study and to assist personnel assigned to the Logistics Management Center, should changes or modification of the programs become necessary. The user's guide is contained in appendix B.

V. Conclusions and Recommendations

Summary

The purpose of this research was to apply database management techniques to the management of the reports of discrepancies. The overall goal was to reduce the number of manhours required to manage the program and to improve the reliability of information. Using the knowledge acquired from an in-depth review of literature concerning the ROD program and detailed information provided by potential users, the researcher developed an application program to accomplish the overall goal. The application program, entitled Report of Discrepancy Information System (RODIS), could have universal application to all Air Force base supply organizations. Personnel assigned to the Air Force Logistics Management Center will test and, if necessary, further refine RODIS.

Conclusions

In this study, the researcher developed application programs designed to automate the manual procedures of the ROD program. While the program is a positive step towards improving the efficiency of assigned personnel, it is not a cure all. Instead, it is only a tool that, when used properly, could aid assigned personnel in performing their duties smarter and in a more timely manner.

As with any computer system, RODIS will only be as accurate as the information the user enters into the system. Simply stated, "Junk in, junk out!" For this program to benefit the user, the user must take care to ensure that the information entered into the system is as accurate as possible. Without user commitment to accuracy, RODIS will be little more than a collection of menu screens.

All testing and further development of RODIS will be conducted by personnel assigned to the Logistics Management Center at Gunter AFB, Alabama. RODIS was examined by personnel assigned the local supply organization. The response has been positive, and copies were given to the personnel for their use.

Recommendations

Recommendations for further study fall into two basic areas. The first deals with revising the SF Form 364. The second addresses establishing an automated link between the bases and the depots.

SF Form 364. A study should be conducted to determine how the SF Form 364 should be revised. During this study, the research observed that many of the blocks on the SF Form 364 were rarely used (blocks 5b, 6, 7a, and 7b). If this is the case, the form could be simplified by either consolidating or totally eliminating unnecessary blocks. With the capabilities of RODIS to print information directly

on to the SF Form 364, some effort should be made to adapt the forms so that they may be put through a standard printer. This would save additional man-hours by eliminating the need for the clerk to type the information on the SF Form 364. The clerk could print the information directly from the computer.

Automated Link. A study should be conducted to determine the feasibility of establishing some type of electronic link between the bases and the depots. Establishment of an electronics link would greatly reduce the paperwork load between the bases and the depots and provide more real-time status and disposition of damaged assets.

Appendix A: Report of Discrepancy Information System Data Dictionary

The RODIS data dictionary is comprised of two databases, ROD_FILE.DBF and INACTIVE.DBF. ROD_FILE.DBF contains all open reports while INACTIVE.DBF contains the closed/completed reports. Because the structure of the two is identical, the database structure is shown only once. The template column identifies restrictions placed on the characters RODIS will accept as inputs to the individual fields within the databases. An "A" means an alpha character is required. A "9" means a numeric character is required while a "X" means that either an alpha or numeric is acceptable.

ROD_FILE.DBF and INACTIVE.DBF

Field Name	Data Element	Type	Length	Template
Depot Name	DEPOTNAME	Char	30	
Deport Address	DEPOT_ADD	Char	30	
Report Number	RPT_NUMBER	Num	8	99999999
Routing Ident	ROUTING_ID	Char	3	
Shipper Name	SHIP_NAME	Char	28	
Shipper Address	SHIP_ADD	Char	28	
Requisition Nbr	REQ_NUMBER	Char	15	AA999999999999A
NSN/PN	NSN_OR_PN	Char	15	9999AAAAAAAAAAA
Nomenclature	NOUN	Char	20	
Unit of Issue	UI	Char	2	AA
Qty Shipped	QTY_SHIP	Num	5	99999
Qty Received	QTY_RCVD	Num	5	99999
Bad Quantity	BAD_QTY	Num	5	99999
Unit Cost	UNIT_COST	Num	7	9999.99
Total Cost	TOTAL_COST	Num	7	9999.99
Discrep Code	DSCP_CODE	Char	2	A9
Action Code	ACTIONCODE	Char	2	9A
Follow-up Date	DUE_FOLLUP	Date	8	
Last Follow-up	LAST_FOLL	Date	8	
Date Completed	DATE_COMP	Date	8	
Remarks 1	REMARKS1	Char	76	
Remarks 2	REMARKS2	Char	76	
Remarks 3	REMARKS3	Char	76	
Remarks 4	REMARKS4	Char	76	
Remarks 5	REMARKS5	Char	76	
Remarks 6	REMARKS6	Char	76	
Name/Rank	NAME_RANK	Char	50	
Duty Title	DUTY_TITLE	Char	50	

Appendix B: Report of Discrepancy Information System User's Guide

Overview

This appendix addresses the purposes/functions of the programs that comprise RODIS. The purpose of this appendix is to assist the reader in understanding and using the programs and to assist personnel assigned to the Logistics Management Center, should changes or modifications to the original programs become necessary.

Report of Discrepancy Information System (RODIS)

RODIS is a menu-driven system that prompts the user to select the action to be taken. Because of this built-in feature, most of the system's operations are self-explanatory. The system provides users the capability to electronically store, update, and print forms which were manually produced in the past. The system is composed of twelve individual programs, each performing a different function. Figure 4 depicts the hierarchy of RODIS programs.

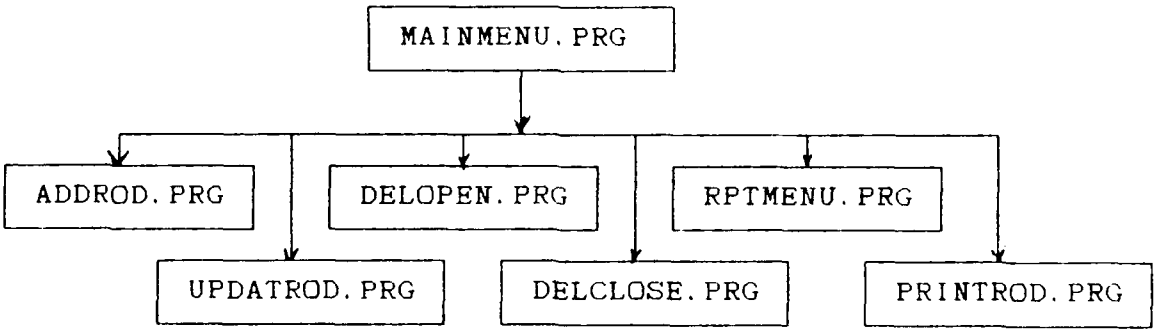

Figure 4. Hierarchy of RODIS Major Programs.

MAINMENU.PRG. This program provides an options menu from which the user selects. The user accesses RODIS by typing DO MAINMENU at the dot prompt and pressing ENTER. Afterwards, an introductory screen (along with a system prompt informing the user to press any key to continue) appears. After the user presses a key, the options menu screen appears. The user then enters the number for the desired function. Figure 5 displays the option screen.

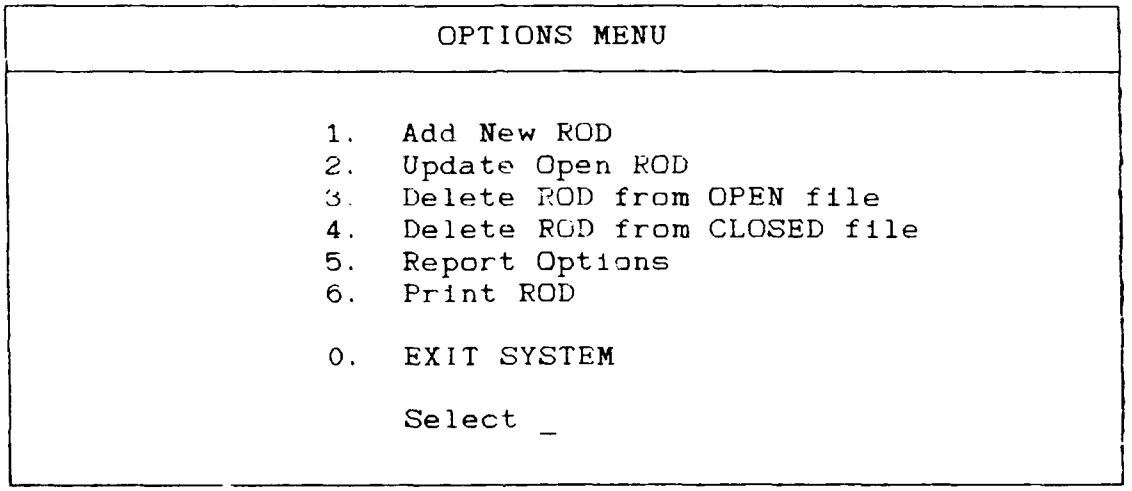

Figure 5. MAINMENU.PRG Options Menu.

ADDROD.PRG. The purpose of this program is to allow the user to add reports to the database. If the user enters the number "1" at the options menu, the ROD add screen appears. First, the user first enters the report number for the report. After this entry is made, the user enters the information into the appropriate data field as it would appear on the SF Form 364. The system accommodates data entry through the use of two input screens, which resemble

the SF Form 364. After the user enters data into the last data field, the computer prompts the user to decide whether or not to add the report to the database. If the user chooses yes, the system adds the report to the database and presents a new add screen. If the user enters the number 0 as the report number, the system returns to the options menu.

UPDATROD.PRG. This program allows the user to update a report within the database. If the user chooses the number "2" at the options menu, the ROD update screen appears. The user must enter a report number that is contained within the database. If not, the computer displays the following message, "REPORT NUMBER DOES NOT EXIST WITHIN DATABASE". This message will continually be displayed until the user either enters a report number that exists within the system or enters the number "0", which will cause the system to return to the options menu. If the user enters a report number that exists within the system, the information, as it was last stored, will appear on the screen. The system advances the cursor to each data field on the screen so the user can either leave the information as is (by pressing ENTER) or change the information (by typing the new information). The system functions as a word processor and accepts most keyboard functions, such as capitalization, deletion, and insertion.

DELOPEN.PRG. This program deletes reports from the database. If the user chooses the number "3" from the

options menu, the open ROD delete screen appears. The user must enter a report number that exists within the database or the system will display a message informing the user the report number does not exist within the database. When the user enters a valid report number, the system displays the report associated with the specified report number. The system does not allow the user to change any of the data fields. The system then prompts the user to decide whether or not to delete the file. If the user chooses yes, the system asks the user to verify the decision by making the same response. If the user responds yes to the second system prompt, the system deletes the file from the open database (ROD_FILE.DBF) and transfers the file to the closed (INACTIVE.DBF) database. The system informs the user the record was deleted and returns the user to the ROD delete screen. If the user enters the number "0" in the report number block, the system exits the delete function and returns to the options menu.

DELCLOSE.PRG. This program deletes closed/completed reports from the closed database. If the user chooses the number "4" from the options menu, the closed ROD delete screen appears. The procedures of this program are identical to those of DELOPEN.PRG discussed above.

RPTMENU.PRG. This program provides the user the option of producing five reports from the Report of Discrepancies (RODs) stored within the open database. If the user enters

the number "5" at the options menu, the system presents a menu screen listing five options (see Figure 6). The options are self-explanatory and are not discussed. After the user selects the desired function, the system prompts the user to specify whether the output should go to the printer or the screen. Following the users selection, the system directs the output to the specified device. The system then returns to the report options menu. If the user enters the number "0", the system returns to the options menu.

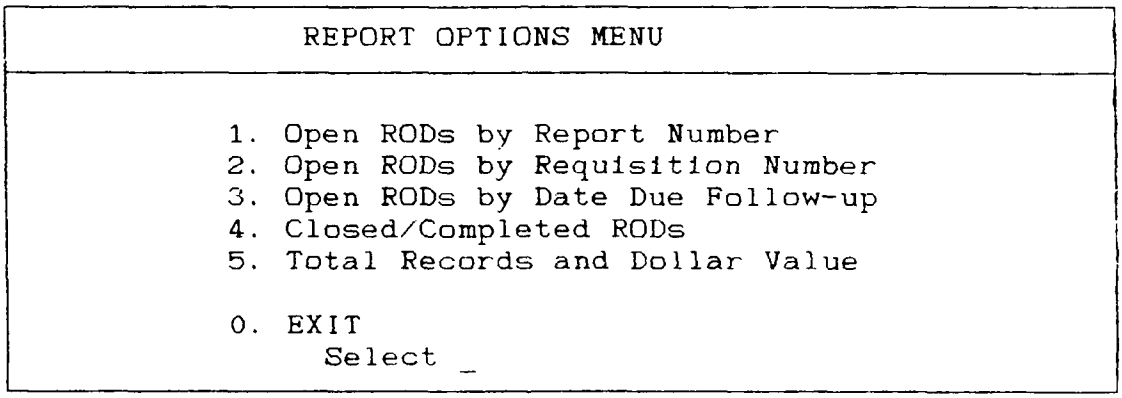

Figure 6. RPTMENU.PRG Options Menu.

PRINTROD.PRG. This program prints the information stored within the database for a specified report. If the user chooses the number "6" from the options menu, the print screen appears. The user must enter the number of a report that exists within the system. If the number does not exist, the system displays a message notifying the user the report number does not exist. After the user enters a valid number, the system displays the information contained in the report. The system then prompts the user to decide whether to print

the information. If the user enters yes, the system prompts the user to decide whether to send the output to the screen or the printer. Following the user's selection, the system sends the output to the device specified. The system then returns to the report options menu.

Appendix C: Report of Discrepancy Information System Program Code

```
*************************************************************
* Program...: MAINMENU.PRG                                  *
* Author....: Capt James L. Johnson                         *
* Date......: 6/30/1989                                     *
* Note......: Main menu.  This program provides a menu that *
*   lists six functions from which the users selects.       *
*************************************************************
* Change the environment by setting talk and bell off.
SET TALK OFF
SET BELL OFF

CLEAR

SET COLOR TO W/B

TEXT




   REPORT OF DISCREPANCY INFORMATION SYSTEM (RODIS)


    Welcome to the Report of Discrepancy Information System.
This is a menu driven information system designed to help
manage the base-level Report of Discrepancy (ROD) Program.
The programs for RODIS were written using dBASE III Plus and
designed to run on any IBM compatible computer system.




 ENDTEXT

 * Draw line around text.
 @ 1, 0 TO 20,79 DOUBLE

 WAIT SPACE(25) + "Press any key to continue"

 CLEAR
```

```
* Loop as long as user desires another action.
DO WHILE .T.
  * Display customer menu upon screen
  CLEAR
  @  5, 29 SAY "OPTIONS MENU"
  @  9, 26 SAY "1. Add New ROD"
  @ 10, 26 SAY "2. Update Open ROD"
  @ 11, 26 SAY "3. Delete ROD from OPEN file"
  @ 12, 26 SAY "4. Delete ROD from CLOSED file"
  @ 13, 26 SAY "5. Report Options"
  @ 14, 26 SAY "6. Print ROD"
  @ 16, 26 SAY "0. EXIT SYSTEM"
  @ 18, 29 SAY "Select"

  * Put double line around menu and use double line to
  *   separate title from menu choice.
  @ 4,0 TO 19,79 DOUBLE
  @ 6,1 TO 6,78 DOUBLE

  * Initialize selectnum to 0.  Prompt user for choice.
  selectnum = 0
  @ 18, 37 GET selectnum PICTURE "9"    RANGE 0,6
  READ
  CLEAR

  * Take appropriate action based on value of selectnum
  IF selectnum = 0
    * Return environment to original state and leave
    *   program.
    SET TALK ON
    SET BELL ON
    RETURN
  ENDIF

  IF selectnum = 1
    * Add new ROD
    @ 19, 10 SAY "Loading ADD program, please wait"
    SET PROCEDURE TO addrod
    DO interact
    SET PROCEDURE TO
  ENDIF

  IF selectnum = 2
    * Update Existing ROD
    @ 19, 10 SAY "Loading UPDATE program, please wait"
    SET PROCEDURE TO updatrod
    DO interact
    SET PROCEDURE TO
  ENDIF
```

```
  IF selectnum = 3
    * Delete ROD from OPEN file.
    @ 19, 10 SAY "Loading DELETE program, please wait"
    SET PROCEDURE TO delopen
    DO interact
    SET PROCEDURE TO
  ENDIF

  IF selectnum = 4
    * Delete ROD from Closed file.
    @ 19, 10 SAY "Loading DELETE program, please wait"
    SET PROCEDURE TO delclose
    Do interact
    SET PROCEDURE TO
  ENDIF

  IF selectnum = 5
    * Reports option
    @ 19, 10 SAY "Loading REPORT OPTIONS, please wait"
    DO rptmenu
  ENDIF

  IF selectnum = 6
    * Print option
    @ 19, 10 SAY "Loading PRINT program, please wait"
    SET PROCEDURE TO printrod
    DO interact
    SET PROCEDURE TO
  ENDIF

ENDDO
```

```
*************************************************************
* Program..: ADDROD.PRG                                     *
* Author...: Capt James L. Johnson                          *
* Date.....: 6/30/1989                                      *
* Note.....: Program adds Report of Discrepancies (ROD)     *
*   to the file using a input screen.                       *
*************************************************************
*************************************************************
* Procedure..: INTERACT                                     *
* Purpose....: This is the main procedure within this       *
*   procedure file.  It contains the overall interaction    *
*   with the user in adding records to the ROD file.        *
*************************************************************
PROCEDURE interact

* Open database file.
USE rod_file INDEX i_refnbr

* Continuous loop until user enters 0 at which time the
*   program exits the loop.
DO WHILE .T.
  CLEAR
  @  1, 28 SAY "ADD NEW ROD SCREEN"
  @  1, 70 SAY DATE()
  @  3, 26 SAY "Report Number:"
  @  3, 52 SAY "(0 TO EXIT)"
  @  5, 2  SAY "Depot:"
  @  5, 42 SAY "Shipper"
  @  6, 2  SAY "Name"
  @  6, 42 SAY "Name"
  @  7, 2  SAY "Address"
  @  7, 42 SAY "Address"
  @  9, 2  SAY "Requisition Number"
  @  9, 42 SAY "Routing Identifier"
  @ 11, 26 SAY "DISCREPANCY DATA"
  @ 12, 48 SAY "Qty"
  @ 12, 57 SAY "Qty"
  @ 12, 64 SAY "Disc"
  @ 12, 71 SAY "Unit"
  @ 13, 2  SAY "NSN/Part Number   Nomenclature"
  @ 13, 43 SAY "UI"
  @ 13, 48 SAY "Shipped"
  @ 13, 57 SAY "Rcvd"
  @ 13, 64 SAY "Qty"
  @ 13, 71 SAY "Cost"
  @ 16, 2  SAY "Total"
  @ 16, 12 SAY "Discrepancy"
  @ 16, 27 SAY "Action"
  @ 16, 37 SAY "Date Due"
  @ 16, 51 SAY "Date Last"
  @ 16, 70 SAY "Date"
  @ 17, 2  SAY "Cost"
```

```
@ 17, 15 SAY "Code"
@ 17, 28 SAY "Code"
@ 17, 37 SAY "Follow-up"
@ 17, 51 SAY "Follow-up"
@ 17, 64 SAY "Completed/Closed"

* Draw lines to separate input data.
@  2, 0 TO 20, 79 DOUBLE
@  4, 1 TO  4, 78 DOUBLE
@ 10, 1 TO 10, 78 DOUBLE
@ 15, 1 TO 15, 78

* Initialize memory variables for user inputs.
STORE 0 TO m_rptnbr
STORE SPACE(30) TO m_dptname
STORE SPACE(3) TO m_routid
STORE SPACE(30) TO m_dptadd
STORE SPACE(28) TO m_shipname
STORE SPACE(28) TO m_shipadd
STORE SPACE(2) TO m_ui
STORE 0 TO m_qtyship
STORE 0 TO m_qtyrcvd
STORE 0 TO m_badqty
STORE SPACE(15) TO m_reqnnbr
STORE SPACE(15) TO m_nsn
STORE SPACE(20) TO m_noun
STORE 0.00 TO m_unitcost
STORE 0.00 TO m_totlcost
STORE SPACE(2) TO m_dcspcode
STORE SPACE(8) TO m_duefoll
STORE SPACE(8) TO m_lastfoll
STORE SPACE(2) TO m_actioncd
STORE SPACE(8) TO m_datecomp
STORE SPACE(76) TO m_remarks1
STORE SPACE(76) TO m_remarks2
STORE SPACE(76) TO m_remarks3
STORE SPACE(76) TO m_remarks4
STORE SPACE(76) TO m_remarks5
STORE SPACE(76) TO m_remarks6
STORE SPACE(50) TO m_name
STORE SPACE(50) TO m_title

* Use initialized variables To clear the screen.
@  3, 41 SAY m_rptnbr PICTURE "99999999"
@  6, 10 SAY m_dptname
@  6, 51 SAY m_shipname
@  7, 10 SAY m_dptadd
@  7, 51 SAY m_shipadd
@  9, 22 SAY m_reqnnbr
@  9, 62 SAY m_routid
@ 14,  2 SAY m_nsn
@ 14, 20 SAY m_noun
```

```
@ 14, 43 SAY m_ui
@ 14, 48 SAY m_qtyship
@ 14, 57 SAY m_qtyrcvd
@ 14, 64 SAY m_badqty
@ 14, 71 SAY m_unitcost PICTURE "99999.99"
@ 18,  2 SAY m_totlcost PICTURE "99999.99"
@ 18, 16 SAY m_dcspcode
@ 18, 29 SAY m_actioncd
@ 18, 37 SAY m_duefoll PICTURE "@D"
@ 18, 51 SAY m_lastfoll PICTURE "@D"
@ 18, 69 SAY m_datecomp PICTURE "@D"

* Get report number from the user.
DO getrpt

* If report number = 0, exit loop.
IF m_rptnbr = 0
  EXIT
ENDIF

* Prepare data field for input.
m_dptname = rod_file->depotname
m_shipname = rod_file->ship_name
m_dptadd = rod_file->depot_add
m_shipadd = rod_file->ship_add
m_routid = rod_file->routing_id
m_reqnbr = rod_file->req_number
m_nsn = rod_file->nsn_or_pn
m_ui = rod_file->ui
m_qtyship = rod_file->qty_ship
m_qtyrcvd = rod_file->qty_rcvd
m_badqty = rod_file->bad_qty
m_totlcost = rod_file->total_cost
m_dcspcode = rod_file->dscp_code
m_duefoll = rod_file->due_follup
m_lastfoll = rod_file->last_foll
m_actioncd = rod_file->actioncode
m_datecomp = rod_file->date_comp
m_remarks1 = rod_file->remarks1
m_remarks2 = rod_file->remarks2
m_remarks3 = rod_file->remarks3
m_remarks4 = rod_file->remarks4
m_remarks5 = rod_file->remarks5
m_remarks6 = rod_file->remarks6
m_name = rod_file->name_rank
m_title = rod_file->duty_title

* Get info concerning other variables from the user.
DO getinfo

CLEAR
```

```
* Create second screen for input.
DO screen2

* Prompt user to decide whether to update database.
SET COLOR TO R+
WAIT "Add record to database? (Y or N)" TO userresp
SET COLOR TO W/B
IF UPPER(userresp) # "Y"
   CLEAR
ENDIF

* If the user input "Y", then update database.
IF UPPER(userresp) = "Y"
    * Add a blank record to the database file.
    APPEND BLANK
    * Input memory variable into corresponding database
    *   field.
    REPLACE rod_file->rpt_number WITH m_rptnbr
    REPLACE rod_file->depotname WITH m_dptname
    REPLACE rod_file->ship_name WITH m_shipname
    REPLACE rod_file->depot_add WITH m_dptadd
    REPLACE rod_file->ship_add WITH m_shipadd
    REPLACE rod_file->ui WITH m_ui
    REPLACE rod_file->qty_ship WITH m_qtyship
    REPLACE rod_file->qty_rcvd WITH m_qtyrcvd
    REPLACE rod_file->bad_qty WITH m_badqty
    REPLACE rod_file->routing_id WITH m_routid
    REPLACE rod_file->req_number WITH m_reqnnbr
    REPLACE rod_file->nsn_or_pn WITH m_nsn
    REPLACE rod_file->noun WITH m_noun
    REPLACE rod_file->unit_cost WITH m_unitcost
    REPLACE rod_file->total_cost WITH m_totlcost
    REPLACE rod_file->dscp_code WITH m_dcspcode
    REPLACE rod_file->due_follup WITH m_duefoll
     REPLACE rod_file->last_foll WITH m_lastfoll
    REPLACE rod_file->actioncode WITH m_actioncd
    REPLACE rod_file->date_comp WITH m_datecomp
    REPLACE rod_file->remarks1 WITH m_remarks1
    REPLACE rod_file->remarks2 WITH m_remarks2
    REPLACE rod_file->remarks3 WITH m_remarks3
    REPLACE rod_file->remarks4 WITH m_remarks4
    REPLACE rod_file->remarks5 WITH m_remarks5
    REPLACE rod_file->remarks6 WITH m_remarks6
    REPLACE rod_file->name_rank WITH m_name
    REPLACE rod_file->duty_title WITH m_title
    message = "Record ADDED to database"
  ELSE
    message = "Record NOT ADDED to database"
ENDIF

* Clear and display message.
CLEAR
```

```
  DO displmsg
  DO clearmsg
ENDDO

CLEAR
CLOSE DATABASES
RETURN
*************************************************************
* Procedure...: GETRPT                                      *
* Called From.: INTERACT                                    *
* Purpose.....: Obtain and validate the report number from *
*   the user.                                               *
*************************************************************
PROCEDURE getrpt

DO WHILE .T.
  * Obtain a reference number from the user and display.
  @  3, 41 GET m_rptnbr PICTURE "99999999"
  READ
  @  3, 41 SAY m_rptnbr PICTURE "99999999"

  * If report number = 0, exit from loop.  If not 0, then
  *   attempt to find the reference number.  If found,
  *   display message and exit the loop.
  IF m_rptnbr = 0
       EXIT
     ELSE
       SEEK m_rptnbr
       * For the add program, no record should exist.  This
       *   helps to ensure that two records do not have the
       *   same reference number.
       IF .NOT. FOUND ()
         EXIT
        ELSE
         message = "ROD ALREADY EXISTS FOR THIS REPORT NUMBER"
         DO displmsg
         DO clearmsg
         @ 20,  0 TO 20, 79 DOUBLE
       ENDIF
  ENDIF
ENDDO

RETURN
*************************************************************
* Procedure....: GETINFO                                    *
* Called From..: INTERACT                                   *
* Purpose......: To obtain and verify all other fields      *
*************************************************************
PROCEDURE getinfo

DO WHILE .T.
  * Obtain values for remaining fields from the user.
```

```
@  6, 10 GET m_dptname
@  7, 10 GET m_dptadd
@  6, 51 GET m_shipname
@  7, 51 GET m_shipadd
@  9, 22 GET m_reqnnbr PICTURE "AA999999999999A"
@  9, 62 GET m_routid
@ 14, 2  GET m_nsn PICTURE "9999AAAAAAAAAAA"
@ 14, 20 GET m_noun
@ 14, 43 GET m_ui PICTURE "AA"
@ 14, 48 GET m_qtyship PICTURE "99999"
@ 14, 57 GET m_qtyrcvd PICTURE "99999"
@ 14, 64 GET m_badqty PICTURE "99999"
@ 14, 71 GET m_unitcost PICTURE "99999.99"
@ 18,  2 GET m_totlcost PICTURE "99999.99"
@ 18, 16 GET m_dcspcode PICTURE "A9"
@ 18, 29 GET m_actioncd PICTURE "9A"
@ 18, 37 GET m_duefoll PICTURE "@D"
@ 18, 51 GET m_lastfoll PICTURE "@D"
@ 18, 69 GET m_datecomp PICTURE "@D"
READ

* Display data
@  6, 10 SAY m_dptname
@  7, 10 SAY m_dptadd
@  6, 51 SAY m_shipname
@  7, 51 SAY m_shipadd
@  9, 22 SAY m_reqnnbr
@  9, 62 SAY m_routid
@ 14, 2  SAY m_nsn
@ 14, 20 SAY m_noun
@ 14, 43 SAY m_ui
@ 14, 48 SAY m_qtyship
@ 14, 57 SAY m_qtyrcvd
@ 14, 64 SAY m_badqty
@ 14, 71 SAY m_unitcost PICTURE "99999.99"
@ 18,  2 SAY m_totlcost PICTURE "99999.99"
@ 18, 16 SAY m_dcspcode
@ 18, 29 SAY m_actioncd
@ 18, 37 SAY m_duefoll PICTURE "@D"
@ 18, 51 SAY m_lastfoll PICTURE "@D"
@ 18, 69 SAY m_datecomp PICTURE "@D"
EXIT

ENDDO

RETURN
```

```
*************************************************************
* Procedure..: Screen2                                      *
* Purpose....: Program provides a second data entry form.   *
*************************************************************
PROCEDURE Screen2
DO WHILE .T.
  @  1, 25 SAY "ROD ADD SCREEN (CONTINUED)"
  @  1, 70 SAY DATE()
  @  3,  2 SAY "Remarks:"
  @ 12, 30 SAY "PREPARING OFFICIAL"
  @ 13,  2 SAY "Name/Rank/Off Sym :"
  @ 14,  2 SAY "Duty Title/Phone  :"

  * initialize variables
  @  4, 2 SAY m_remarks1
  @  5, 2 SAY m_remarks2
  @  6, 2 SAY m_remarks3
  @  7, 2 SAY m_remarks4
  @  8, 2 SAY m_remarks5
  @  9, 2 SAY m_remarks6
  @ 13,23 SAY m_name
  @ 14,23 SAY m_title

  * draw lines
  @  2,0 TO 16, 79 DOUBLE
  @ 10,1 TO 10, 79

  * get info from user
  @  4, 2 GET m_remarks1
  @  5, 2 GET m_remarks2
  @  6, 2 GET m_remarks3
  @  7, 2 GET m_remarks4
  @  8, 2 GET m_remarks5
  @  9, 2 GET m_remarks6
  @ 13,23 GET m_name
  @ 14,23 GET m_title
  READ

  * display data
  @  4, 2 SAY m_remarks1
  @  5, 2 SAY m_remarks2
  @  6, 2 SAY m_remarks3
  @  7, 2 SAY m_remarks4
  @  8, 2 SAY m_remarks5
  @  9, 2 SAY m_remarks6
  @ 13,23 SAY m_name
  @ 14,23 SAY m_title
  EXIT
ENDDO
RETURN
```

```
*************************************************************
* Procedure...: DISPLMSG                                    *
* Called From.: Several procedures                          *
* Purpose.....: To display the contents of the variable     *
*   called "message"                                        *
*************************************************************
PROCEDURE displmsg

DO clearmsg
SET COLOR TO R+
@ 19, 10 SAY message
SET COLOR TO W/B
WAIT SPACE(10) + "Press any key to continue..."
RETURN

*************************************************************
* Procedure...: CLEARMSG                                    *
* Called From.: Several procedures                          *
* Purpose.....: To clear any message that is present on     *
*   line 19 of the data entry screen.                       *
*************************************************************
PROCEDURE clearmsg

@ 19, 10 SAY SPACE(68)
@ 20, 10 SAY SPACE(68)
RETURN
```

```
*************************************************************
* Program..: UPDATROD.PRG                                   *
* Author...: Capt James L. Johnson                          *
* Date.....: 6/30/1989                                      *
* Note.....: Program updates ROD files through use of an    *
*   input screen.                                           *
*************************************************************
*************************************************************
* Procedure..: INTERACT                                     *
* Purpose....: This is the main procedure within this       *
*    procedure file.  It contains the overall interaction   *
*    with the user in adding records to the ROD file.       *
*************************************************************
PROCEDURE interact

* Open database file.
USE rod_file INDEX i_refnbr

* Continuous loop until user enters 0 at which time the
*    program exits the loop.
DO WHILE .T.
  CLEAR
  @  1, 28 SAY "ROD UPDATE SCREEN"
  @  1, 70 SAY DATE()
  @  3, 26 SAY "Report Number:"
  @  3, 52 SAY "(0 TO EXIT)"
  @  5, 2  SAY "Depot:"
  @  5, 42 SAY "Shipper"
  @  6, 2  SAY "Name"
  @  6, 42 SAY "Name"
  @  7, 2  SAY "Address"
  @  7, 42 SAY "Address"
  @  9, 2  SAY "Requisition Number"
  @  9, 42 SAY "Routing Identifier"
  @ 11, 26 SAY "DISCREPANCY DATA"
  @ 12, 48 SAY "Qty"
  @ 12, 57 SAY "Qty"
  @ 12, 64 SAY "Disc"
  @ 12, 71 SAY "Unit"
  @ 13, 2  SAY "NSN/Part Number   Nomenclature"
  @ 13, 43 SAY "UI"
  @ 13, 48 SAY "Shipped"
  @ 13, 57 SAY "Rcvd"
  @ 13, 64 SAY "Qty"
  @ 13, 71 SAY "Cost"
  @ 16, 2  SAY "Total"
  @ 16, 12 SAY "Discrepancy"
  @ 16, 27 SAY "Action"
  @ 16, 37 SAY "Date Due"
  @ 16, 51 SAY "Date Last"
  @ 16, 70 SAY "Date"
  @ 17,  2 SAY "Cost"
```

```
@ 17, 15 SAY "Code"
@ 17, 28 SAY "Code"
@ 17, 37 SAY "Follow-up"
@ 17, 51 SAY "Follow-up"
@ 17, 64 SAY "Completed/Closed"

* Draw lines to separate input data.
@  2, 0 TO 20, 79 DOUBLE
@  4, 1 TO  4, 78 DOUBLE
@ 10, 1 TO 10, 78 DOUBLE
@ 15, 1 TO 15, 78

* Initialize memory variables for user inputs.
STORE 0 TO m_rptnbr
STORE SPACE(30) TO m_dptname
STORE SPACE(30) TO m_dptadd
STORE SPACE(28) TO m_shipname
STORE SPACE(28) TO m_shipadd
STORE SPACE(2) TO m_ui
STORE 0 TO m_qtyship
STORE 0 TO m_qtyrcvd
STORE 0 TO m_badqty
STORE SPACE(15) TO m_reqnnbr
STORE SPACE(3) TO m_routid
STORE SPACE(15) TO m_nsn
STORE SPACE(20) TO m_noun
STORE 0.00 TO m_unitcost
STORE 0.00 TO m_totlcost
STORE SPACE(2) TO m_dcspcode
STORE SPACE(8) TO m_duefoll
STORE SPACE(8) TO m_lastfoll
STORE SPACE(2) TO m_actioncd
STORE SPACE(8) TO m_datecomp
STORE SPACE(76) TO m_remarks1
STORE SPACE(76) TO m_remarks2
STORE SPACE(76) TO m_remarks3
STORE SPACE(76) TO m_remarks4
STORE SPACE(76) TO m_remarks5
STORE SPACE(76) TO m_remarks6
STORE SPACE(50) TO m_name
STORE SPACE(50) TO m_title

* Use initialized variables To clear the screen.
@  3, 41 SAY m_rptnbr PICTURE "99999999"
@  6, 10 SAY m_dptname
@  6, 51 SAY m_shipname
@  7, 10 SAY m_dptadd
@  7, 51 SAY m_shipadd
@  9, 22 SAY m_reqnnbr
@  9, 62 SAY m_routid
@ 14,  2 SAY m_nsn
@ 14, 20 SAY m_noun
```

```
@ 14, 43 SAY m_ui
@ 14, 48 SAY m_qtyship
@ 14, 57 SAY m_qtyrcvd
@ 14, 64 SAY m_badqty
@ 14, 71 SAY m_unitcost PICTURE "99999.99"
@ 18,  2 SAY m_totlcost PICTURE "99999.99"
@ 18, 16 SAY m_dcspcode
@ 18, 29 SAY m_actioncd
@ 18, 37 SAY m_duefoll PICTURE "@D"
@ 18, 51 SAY m_lastfoll PICTURE "@D"
@ 18, 69 SAY m_datecomp PICTURE "@D"

* Get report number from the user.
DO getrpt

* If report number = 0, exit loop.
IF m_rptnbr = 0
  EXIT
ENDIF

* Get info concerning other variables from the user.
DO getinfo

CLEAR

DO screen2

* Prompt user to decide whether to update database.
SET COLOR TO R+
WAIT "Update database? (Y or N)" TO userresp
SET COLOR TO W/B
IF UPPER(userresp) # "Y"
  CLEAR
ENDIF

* If the user input "Y", then update database.
IF UPPER(userresp) = "Y"
   REPLACE rod_file->rpt_number WITH m_rptnbr
   REPLACE rod_file->depotname WITH m_dptname
   REPLACE rod_file->ship_name WITH m_shipname
   REPLACE rod_file->depot_add WITH m_dptadd
   REPLACE rod_file->ship_add WITH m_shipadd
   REPLACE rod_file->ui WITH m_ui
   REPLACE rod_file->qty_ship WITH m_qtyship
   REPLACE rod_file->qty_rcvd WITH m_qtyrcvd
   REPLACE rod_file->bad_qty WITH m_badqty
   REPLACE rod_file->routing_id WITH m_routid
   REPLACE rod_file->req_number WITH m_reqnnbr
   REPLACE rod_file->nsn_or_pn WITH m_nsn
   REPLACE rod_file->noun WITH m_noun
   REPLACE rod_file->unit_cost WITH m_unitcost
   REPLACE rod_file->total_cost WITH m_totlcost
```

```
      REPLACE rod_file->dscp_code WITH m_dcspcode
      REPLACE rod_file->due_follup WITH m_duefoll
      REPLACE rod_file->last_foll WITH m_lastfoll
      REPLACE rod_file->actioncode WITH m_actioncd
      REPLACE rod_file->date_comp WITH m_datecomp
      REPLACE rod_file->remarks1 WITH m_remarks1
      REPLACE rod_file->remarks2 WITH m_remarks2
      REPLACE rod_file->remarks3 WITH m_remarks3
      REPLACE rod_file->remarks4 WITH m_remarks4
      REPLACE rod_file->remarks5 WITH m_remarks5
      REPLACE rod_file->remarks6 WITH m_remarks6
      REPLACE rod_file->name_rank WITH m_name
      REPLACE rod_file->duty_title WITH m_title
      message = "Record UPDATED within database"
    ELSE
      message = "Record NOT UPDATED within database"
    ENDIF

    * Clear and display message.
    CLEAR
    DO displmsg
    DO clearmsg
  ENDDO

  CLEAR
  CLOSE DATABASES
  RETURN
  ************************************************************
  * Procedure....: GETRPT                                    *
  * Called From.: INTERACT                                   *
  * Purpose.....: Obtain and validate the report number      *
  *   from the user.                                         *
  ************************************************************
  PROCEDURE getrpt

  DO WHILE .T.
    * Obtain a reference number from the user and display.
    @  3, 41 GET m_rptnbr PICTURE "99999999"
    READ
    @  3, 41 SAY m_rptnbr PICTURE "99999999"

    * If report number = 0, exit from loop.  If not 0, then
    *   attempt to find the reference number.  If found,
    *   display message and exit the loop.
    IF m_rptnbr = 0
      EXIT
    ELSE
      SEEK m_rptnbr
      IF FOUND ()
        m_dptname = rod_file->depotname
        m_shipname = rod_file->ship_name
        m_dptadd = rod_file->depot_add
```

```
      m_shipadd = rod_file->ship_add
      m_routid = rod_file->routing_id
      m_reqnnbr = rod_file->req_number
      m_nsn = rod_file->nsn_or_pn
      m_noun = rod_file->noun
      m_ui = rod_file->ui
       m_qtyship = rod_file->qty_ship
       m_qtyrcvd = rod_file->qty_rcvd
      m_badqty = rod_file->bad_qty
      m_unitcost = rod_file->unit_cost
      m_totlcost = rod_file->total_cost
      m_dcspcode = rod_file->dscp_code
      m_duefoll = rod_file->due_follup
       m_lastfoll = rod_file->last_foll
       m_actioncd = rod_file->actioncode
       m_datecomp = rod_file->date_comp
       m_remarks1 = rod_file->remarks1
      m_remarks2 = rod_file->remarks2
      m_remarks3 = rod_file->remarks3
      m_remarks4 = rod_file->remarks4
      m_remarks5 = rod_file->remarks5
      m_remarks6 = rod_file->remarks6
      m_name = rod_file->name_rank
      m_title = rod_file->duty_title
      EXIT
     ELSE
    message = "REPORT NUMBER DOES NOT EXIST WITHIN DATABASE"
       DO displmsg
      DO clearmsg
       @ 20, 0 TO 20, 79 DOUBLE
    ENDIF
ENDIF
ENDDO

RETURN
*************************************************************
* Procedure.....: GETINFO                                   *
* Called From...: INTFRACT                                  *
* Purpose.......: To obtain and verify all other fields     *
*    other than the key field.                              *
*************************************************************
PROCEDURE getinfo

DO WHILE .T.
  * Obtain values for remaining fields from the user.
  @  6, 10 GET m_dptname
  @  7, 10 GET m_dptadd
  @  6, 51 GET m_shipname
  @  7, 51 GET m_shipadd
  @  9, 22 GET m_reqnnbr PICTURE "AA999999999999A"
  @  9, 62 GET m_routid
  @ 14,  2 GET m_nsn
```

```
  @ 14, 20 GET m_noun
  @ 14, 43 GET m_ui PICTURE "AA"
  @ 14, 48 GET m_qtyship PICTURE "99999"
  @ 14, 57 GET m_qtyrcvd PICTURE "99999"
  @ 14, 64 GET m_badqty PICTURE "99999"
  @ 14, 71 GET m_unitcost PICTURE "99999.99"
  @ 18,  2 GET m_totlcost PICTURE "99999.99"
  @ 18, 16 GET m_dcspcode PICTURE "A9"
  @ 18, 29 GET m_actioncd PICTURE "9A"
  @ 18, 37 GET m_duefoll PICTURE "@D"
  @ 18, 51 GET m_lastfoll PICTURE "@D"
  @ 18, 69 GET m_datecomp PICTURE "@D"
  READ

  * Display data
  @  6, 10 SAY m_dptname
  @  6, 51 SAY m_shipname
  @ 7, 10 SAY m_dptadd
  @  7, 51 SAY m_shipadd
  @  9, 22 SAY m_reqnnbr PICTURE "AA999999999999A"
  @  9, 62 SAY m_routid
  @ 14, 2  SAY m_nsn
  @ 14, 20 SAY m_noun
  @ 14, 43 SAY m_ui
  @ 14, 48 SAY m_qtyship PICTURE "99999"
  @ 14, 57 SAY m_qtyrcvd PICTURE "99999"
  @ 14, 64 SAY m_badqty PICTURE "99999"
  @ 14, 71 SAY m_unitcost PICTURE "99999.99"
  @ 18,  2 SAY m_totlcost PICTURE "99999.99"
  @ 18, 16 SAY m_dcspcode
  @ 18, 29 SAY m_actioncd PICTURE "9A"
  @ 18, 37 SAY m_duefoll PICTURE "@D"
  @ 18, 51 SAY m_lastfoll PICTURE "@D"
  @ 18, 69 SAY m_datecomp PICTURE "@D"
  EXIT
ENDDO

RETURN

*************************************************************
* Procedure...: Screen2                                     *
* Purpose....: Program provides a data entry form to        *
*   allow entry of data that could not be entered in the    *
*   orignal screen.                                         *
*************************************************************
PROCEDURE screen2

DO WHILE .T.
  @  1, 25 SAY "ROD UPDATE SCREEN (CONTINUED)"
  @  1, 70 SAY DATE()
  @  3,  2 SAY "Remarks"
  @ 12, 30 SAY "PREPARING OFFICIAL"
```

```
  @ 13,  2 SAY "Name/Rank/Off Sym :"
  @ 14,  2 SAY "Duty Title/Phone  :"

  * Initialize variables
  @  4,  2 SAY m_remarks1
  @  5,  2 SAY m_remarks2
  @  6,  2 SAY m_remarks3
  @  7,  2 SAY m_remarks4
  @  8,  2 SAY m_remarks5
  @  9,  2 SAY m_remarks6
  @ 13, 23 SAY m_name
  @ 14, 23 SAY m_title

  * Draw lines
  @  2,  0 TO 16, 79 DOUBLE
  @ 10,  1 TO 10, 79

  * Get info from user
  @  4,  2 GET m_remarks1
  @  5,  2 GET m_remarks2
  @  6,  2 GET m_remarks3
  @  7,  2 GET m_remarks4
  @  8,  2 GET m_remarks5
  @  9,  2 GET m_remarks6
  @ 13, 23 GET m_name
  @ 14, 23 GET m_title
  READ

  * Display data
  @  4,  2 SAY m_remarks1
  @  5,  2 SAY m_remarks2
  @  6,  2 SAY m_remarks3
  @  7,  2 SAY m_remarks4
  @  8,  2 SAY m_remarks5
  @  9,  2 SAY m_remarks6
  @ 13, 23 SAY m_name
  @ 14, 23 SAY m_title
  EXIT
ENDDO
RETURN

*************************************************************
* Procedure....: DISPLMSG                                   *
* Called From.: Several procedures                          *
* Purpose.....: To display the contents of the variable     *
*    called "message"                                       *
*************************************************************
PROCEDURE displmsg

DO clearmsg
SET COLOR TO R+
@ 19, 10 SAY message
```

```
SET COLOR TO W/B
WAIT SPACE(10) + "Press any key to continue..."
RETURN

*************************************************************
* Procedure...: CLEARMSG                                    *
* Called From.: Several procedures                          *
* Purpose.....: To clear any message that is present on     *
*   line 19 on the data entry screen.                       *
*************************************************************
PROCEDURE clearmsg

@ 19, 10 SAY SPACE(68)
RETURN
```

```
*************************************************************
* Program..: DELOPEN.PRG                                    *
* Author...: Capt James L. Johnson                          *
* Date.....: 6/30/1989                                      *
* Note.....: Program deletes Report of Discrepancy (ROD)    *
*   files from the active file and transfers the files to   *
*   the inactive or completed file.                         *
*************************************************************
*************************************************************
* Procedure..: INTERACT                                     *
* Purpose....: This is the main procedure within this       *
* procedure file.  It contains the overall interaction with *
* the user in adding records to the ROD file.               *
*************************************************************
PROCEDURE interact

USE rod_file INDEX i_refnbr

* Continuous loop until user enters 0 at which time the
*   program exits the loop.
DO WHILE .T.
   CLEAR
   @  1, 26 SAY "OPEN ROD DELETE SCREEN"
   @  1, 70 SAY DATE()
   @  3, 11 SAY "Report Number to be deleted:"
   @  3, 52 SAY "(0 TO EXIT)"
   @  5, 2  SAY "Depot:"
   @  5, 42 SAY "Shipper"
   @  6, 2  SAY "Name"
   @  6, 42 SAY "Name"
   @  7, 2  SAY "Address"
   @  7, 42 SAY "Address"
   @  9, 2  SAY "Requisition Number"
   @  9, 42 SAY "Routing Identifier"
   @ 11, 26 SAY "DISCREPANCY DATA"
   @ 12, 48 SAY "Qty"
   @ 12, 57 SAY "Qty"
   @ 12, 64 SAY "Disc"
   @ 12, 71 SAY "Unit"
   @ 13, 2  SAY "NSN/Part Number   Nomenclature"
   @ 13, 43 SAY "UI"
   @ 13, 48 SAY "Shipped"
   @ 13, 57 SAY "Rcvd"
   @ 13, 64 SAY "Qty"
   @ 13, 71 SAY "Cost"
   @ 16, 2  SAY "Total"
   @ 16, 12 SAY "Discrepancy"
   @ 16, 27 SAY "Action"
   @ 16, 37 SAY "Date Due"
   @ 16, 51 SAY "Date Last"
   @ 16, 70 SAY "Date"
   @ 17,  2 SAY "Cost"
```

```
@ 17, 15 SAY "Code"
@ 17, 28 SAY "Code"
@ 17, 37 SAY "Follow-up"
@ 17, 51 SAY "Follow-up"
@ 17, 64 SAY "Completed/Closed"

* Draw lines to separate input data.
@  2, 0 TO 20, 79 DOUBLE
@  4, 1 TO  4, 78 DOUBLE
@ 10, 1 TO 10, 78 DOUBLE
@ 15, 1 TO 15, 78

* Initialize memory variables for user inputs.
STORE 0 TO m_rptnbr
STORE SPACE(30) TO m_dptname
STORE SPACE(30) TO m_dptadd
STORE SPACE(28) TO m_shipname
STORE SPACE(28) TO m_shipadd
STORE SPACE(2) TO m_ui
STORE 0 TO m_qtyship
STORE 0 TO m_qtyrcvd
STORE 0 TO m_badqty
STORE SPACE(15) TO m_reqnnbr
STORE SPACE(3) TO m_routid
STORE SPACE(15) TO m_nsn
STORE SPACE(20) TO m_nou
STORE 0.00 TO m_unitcost
STORE 0.00 TO m_totlcost
STORE SPACE(2) TO m_dcspcode
STORE SPACE(8) TO m_duefoll
STORE SPACE(8) TO m_lastfoll
STORE SPACE(2) TO m_actioncd
STORE SPACE(8) TO m_datecomp
STORE SPACE(76) TO m_remarks1
STORE SPACE(76) TO m_remarks2
STORE SPACE(76) TO m_remarks3
STORE SPACE(76) TO m_remarks4
STORE SPACE(76) TO m_remarks5
STORE SPACE(76) TO m_remarks6
STORE SPACE(50) TO m_name
STORE SPACE(50) TO m_title

* Use initialized variables To clear the screen.
@  3, 41 SAY m_rptnbr PICTURE "99999999"
@  6, 10 SAY m_dptname
@  6, 51 SAY m_shipname
@  7, 10 SAY m_dptadd
@  7, 51 SAY m_shipadd
@  9, 22 SAY m_reqnnbr
@  9, 62 SAY m_routid
@ 14,  2 SAY m_nsn
@ 14, 20 SAY m_noun
```

```
    @ 14, 43 SAY m_ui
    @ 14, 48 SAY m_qtyship
    @ 14, 57 SAY m_qtyrcvd
    @ 14, 64 SAY m_badqty
    @ 14, 71 SAY m_unitcost PICTURE "99999.99"
    @ 18,  2 SAY m_totlcost PICTURE "99999.99"
    @ 18, 16 SAY m_dcspcode
    @ 18, 29 SAY m_actioncd
    @ 18, 37 SAY m_duefoll PICTURE "@D"
    @ 18, 51 SAY m_lastfoll PICTURE "@D"
    @ 18, 69 SAY m_datecomp PICTURE "@D"

    * Get report number from the user.
    DO getrpt

    * If report number = 0, exit loop
    IF m_rptnbr = 0
       EXIT
    ENDIF

    DO showrest

    * Prompt user to decide whether to update database.
    SET COLOR TO R+
    WAIT "Delete ROD file? (Y or N)" TO userresp
    SET COLOR TO W/B
    IF UPPER(userresp) # "Y"
       CLEAR
    ENDIF

    * If the user input "Y", then update database.
    IF UPPER(userresp) = "Y"
       DO drecord
       IF m_rptnbr = 0
          EXIT
       ENDIF
       DELETE
       PACK
       message = "Record DELETED from database"
     ELSE
       message = "Record NOT DELETED from open file"
    ENDIF

    * Clear and display message.
     CLEAR
     DO displmsg
     DO clearmsg
ENDDO

CLEAR
CLOSE DATABASES
RETURN
```

```
*************************************************************
* Procedure....: GETRPT                                     *
* Called From.: INTERACT                                    *
* Purpose.....: Obtain and validate the report number from  *
*   the user.                                               *
*************************************************************
PROCEDURE getrpt

DO WHILE .T.
   * Obtain a reference number from the user and display.
   @  3, 41 GET m_rptnbr PICTURE "99999999"
   READ
   @  3, 41 SAY m_rptnbr PICTURE "99999999"

   * If report number = 0, exit from loop.  If not 0, then
   *   attempt to find the reference number.  If found,
   *   display message and exit the loop.
   IF m_rptnbr = 0
      EXIT
     ELSE
      SEEK m_rptnbr
       IF FOUND ()
         m_dptname = rod_file->depotname
         m_shipname = rod_file->ship_name
         m_dptadd = rod_file->depot_add
         m_shipadd = rod_file->ship_add
         m_routid = rod_file->routing_id
         m_reqnnbr = rod_file->req_number
         m_nsn = rod_file->nsn_or_pn
          m_noun = rod_file->noun
          m_ui = rod_file->ui
         m_qtyship = rod_file->qty_ship
         m_qtyrcvd = rod_file->qty_rcvd
          m_badqty = rod_file->bad_qty
         m_unitcost = rod_file->unit_cost
         m_totlcost = rod_file->total_cost
          m_dcspcode = rod_file->dscp_code
         m_duefoll = rod_file->due_follup
         m_lastfoll = rod_file->last_foll
         m_actioncd = rod_file->actioncode
         m_datecomp = rod_file->date_comp
          m_remarks1 = rod_file->remarks1
          m_remarks2 = rod_file->remarks2
         m_remarks3 = rod_file->remarks3
         m_remarks4 = rod_file->remarks4
         m_remarks5 = rod_file->remarks5
         m_remarks6 = rod_file->remarks6
         m_name = rod_file->name_rank
         m_title = rod_file->duty_title
         EXIT
       ELSE
     message = "REPORT NUMBER DOES NOT EXIST WITHIN DATABASE"
```

```
         DO displmsg
          DO clearmsg
         @ 20, 0 TO 20, 79 DOUBLE
    ENDIF
  ENDIF

ENDDO
RETURN

*************************************************************
* Procedure...: SHOWREST                                    *
* Purpose.....: Displays data contain in database fields.   *
*************************************************************
PROCEDURE showrest

* Display data
@  6, 10 SAY m_dptname
@  6, 51 SAY m_shipname
@  7, 10 SAY m_dptadd
@  7, 51 SAY m_shipadd
@  9, 22 SAY m_reqnnbr
@  9, 62 SAY m_routid
@ 14,  2 SAY m_nsn
@ 14, 20 SAY m_noun
@ 14, 43 SAY m_ui
@ 14, 48 SAY m_qtyship PICTURE "99999"
@ 14, 57 SAY m_qtyrcvd PICTURE "99999"
@ 14, 64 SAY m_badqty PICTURE "99999"
@ 14, 71 SAY m_unitcost PICTURE "99999.99"
@ 18,  2 SAY m_totlcost PICTURE "99999.99"
@ 18, 16 SAY m_dcspcode
@ 18, 29 SAY m_actioncd
@ 18, 37 SAY m_duefoll PICTURE "@D"
@ 18, 51 SAY m_lastfoll PICTURE "@D"
@ 18, 69 SAY m_datecomp PICTURE "@D"
RETURN

*************************************************************
* Procedure...: DRECORD                                     *
* Called From.: interact                                    *
* Purpose.....: Transfers records to inactive database prior*
*     to record being deleted from active or open database  *
*     file.                                                 *
*************************************************************
PROCEDURE drecord

USE inactive

* Clear and draw input screen
CLEAR
@  2, 27 SAY "ROD DELETE SCREEN"
@  4, 15 TO 8, 60 DOUBLE
```

```
@  6, 17 SAY "Reference number to be deleted:"
@  6, 49 SAY m_rptnbr PICTURE "99999999"

SET COLOR TO R+
@ 18, 1 SAY SPACE(1)
WAIT "ARE YOU SURE? (Y or N)" TO delchk
SET COLOR TO W/B
IF UPPER(delchk) # "Y"
   CLEAR
   m_rptnbr = 0
   EXIT
ENDIF

IF UPPER(delchk) = "Y"
   * Transfer record to inactive file
   APPEND FROM rod_file FOR RPT_NUMBER = m_rptnbr
ENDIF

* Reopen active database file.
USE rod_file INDEX i_refnbr

RETURN
*************************************************************
* Procedure...: DISPLMSG                                    *
* Called From.: Several procedures                          *
* Purpose.....: To display the contents of the variable     *
*    called "message".                                      *
*************************************************************
PROCEDURE displmsg

DO clearmsg
SET COLOR TO R+
@ 19, 10 SAY message
SET COLOR TO W/B
WAIT SPACE(10) + "Press any key to continue..."
RETURN

*************************************************************
* Procedure...: CLEARMSG                                    *
* Called From.: Several procedures                          *
* Purpose.....: To clear any message that is present on line*
*   19 on the data entry screen.                            *
*************************************************************
PROCEDURE clearmsg

@ 19, 10 SAY SPACE(68)
RETURN
```

```
*************************************************************
* Program...: DELCLOSE.PRG                                  *
* Author....: Capt James L. Johnson                         *
* Date......: 6/30/1989                                     *
* Note......: Program deletes Report of Discrepancy (ROD)   *
*   files from the CLOSED file.                             *
*************************************************************
*************************************************************
* Procedure..: INTERACT                                     *
* Purpose....: This is the main procedure within this       *
*   procedure file.  It contains the overall interaction    *
*   with the user in adding records to the ROD file.        *
*************************************************************
PROCEDURE interact

USE inactive INDEX i_delnbr

* Continuous loop until user enters 0 at which time the
*   program exits the loop.
DO WHILE .T.
   CLEAR
   @  1, 25 SAY "CLOSED ROD DELETE SCREEN"
   @  1, 70 SAY DATE()
   @  3, 11 SAY "Report Number to be deleted:"
   @  3, 52 SAY "(0 TO EXIT)"
   @  5, 2  SAY "Depot:"
   @  5, 42 SAY "Shipper"
   @  6, 2  SAY "Name"
   @  6, 42 SAY "Name"
   @  7, 2  SAY "Address"
   @  7, 42 SAY "Address"
   @  9, 2  SAY "Requisition Number"
   @  9, 42 SAY "Routing Identifier"
   @ 11, 26 SAY "DISCREPANCY DATA"
   @ 12, 48 SAY "Qty"
   @ 12, 57 SAY "Qty"
   @ 12, 64 SAY "Disc"
   @ 12, 71 SAY "Unit"
   @ 13, 2  SAY "NSN/Part Number   Nomenclature"
   @ 13, 43 SAY "UI"
   @ 13, 48 SAY "Shipped"
   @ 13, 57 SAY "Rcvd"
   @ 13, 64 SAY "Qty"
   @ 13, 71 SAY "Cost"
   @ 16, 2  SAY "Total"
   @ 16, 12 SAY "Discrepancy"
   @ 16, 27 SAY "Action"
   @ 16, 37 SAY "Date Due"
   @ 16, 51 SAY "Date Last"
   @ 16, 70 SAY "Date"
   @ 17,  2 SAY "Cost"
   @ 17, 15 SAY "Code"
```

```
@ 17, 28 SAY "Code"
@ 17, 37 SAY "Follow-up"
@ 17, 51 SAY "Follow-up"
@ 17, 64 SAY "Completed/Closed"

* Draw lines to separate input data.
@  2, 0 TO 20, 79 DOUBLE
@  4, 1 TO  4, 78 DOUBLE
@ 10, 1 TO 10, 78 DOUBLE
@ 15, 1 TO 15, 78

* Initialize memory variables for user inputs.
STORE 0 TO m_rptnbr
STORE SPACE(30) TO m_dptname
STORE SPACE(30) TO m_dptadd
STORE SPACE(28) TO m_shipname
STORE SPACE(28) TO m_shipadd
STORE SPACE(2) TO m_ui
STORE 0 TO m_qtyship
STORE 0 TO m_qtyrcvd
STORE 0 TO m_badqty
STORE SPACE(15) TO m_reqnnbr
STORE SPACE(3) TO m_routid
STORE SPACE(15) TO m_nsn
STORE SPACE(20) TO m_noun
STORE 0.00 TO m_unitcost
STORE 0.00 TO m_totlcost
STORE SPACE(2) TO m_dcspcode
STORE SPACE(8) TO m_duefoll
STORE SPACE(8) TO m_lastfoll
STORE SPACE(2) TO m_actioncd
STORE SPACE(8) TO m_datecomp
STORE SPACE(76) TO m_remarks1
STORE SPACE(76) TO m_remarks2
STORE SPACE(76) TO m_remarks3
STORE SPACE(76) TO m_remarks4
STORE SPACE(76) TO m_remarks5
STORE SPACE(76) TO m_remarks6
STORE SPACE(50) TO m_name
STORE SPACE(50) TO m_title

* USE initialized variables To clear the screen.
@  3, 41 SAY m_rptnbr PICTURE "99999999"
@  6, 10 SAY m_dptname
@  6, 51 SAY m_shipname
@  7, 10 SAY m_dptadd
@  7, 51 SAY m_shipadd
@  9, 22 SAY m_reqnnbr
@  9, 62 SAY m_routid
@ 14,  2 SAY m_nsn
@ 14, 20 SAY m_noun
@ 14, 43 SAY m_ui
```

```
    @ 14, 48 SAY m_qtyship
    @ 14, 57 SAY m_qtyrcvd
    @ 14, 64 SAY m_badqty
    @ 14, 71 SAY m_unitcost PICTURE "99999.99"
    @ 18,  2 SAY m_totlcost PICTURE "99999.99"
    @ 18, 16 SAY m_dcspcode
    @ 18, 29 SAY m_actioncd
    @ 18, 37 SAY m_duefoll PICTURE "@D"
    @ 18, 51 SAY m_lastfoll PICTURE "@D"
    @ 18, 69 SAY m_datecomp PICTURE "@D"

    * Get report number from the user.
    DO getrpt

    * If report number = 0, exit loop
    IF m_rptnbr = 0
      EXIT
    ENDIF

    DO showrest

    * Prompt user to decide whether to update database.
    SET COLOR TO R+
    WAIT "Delete ROD file? (Y or N)" TO userresp
    SET COLOR TO W/B
    IF UPPER(userresp) # "Y"
      CLEAR
    ENDIF

    * If the user input "Y", then update database.
    IF UPPER(userresp) = "Y"
      DO drecord
      IF m_rptnbr = 0
        EXIT
      ENDIF
      message = "Record DELETED from database"
     ELSE
      message = "Record NOT DELETED from open file"
    ENDIF

    * Clear and display message
    CLEAR
    DO displmsg
    DO clearmsg
ENDDO

CLEAR
CLOSE DATABASES
RETURN
```

```
*************************************************************
* Procedure...: GETRPT                                      *
* Called From.: INTERACT                                    *
* Purpose.....: Obtain and validate the report number from  *
*    the user.                                              *
*************************************************************
PROCEDURE getrpt

DO WHILE .T.
   * Obtain a reference number from the user and display.
   @  3, 41 GET m_rptnbr PICTURE "99999999"
   READ
   @  3, 41 SAY m_rptnbr PICTURE "99999999"

   * If report number = 0, exit from loop.  If not 0, then
   *   attempt to find the reference number.  If found,
   *   display message and exit the loop.
   IF m_rptnbr = 0
      EXIT
     ELSE
      SEEK m_rptnbr
      IF FOUND ()
         m_dptname = inactive->depotname
         m_shipname = inactive->ship_name
         m_dptadd = inactive->depot_add
         m_shipadd = inactive->ship_add
         m_routid = inactive->routing_id
         m_reqnnbr = inactive->req_number
         m_nsn = inactive->nsn_or_pn
         m_noun = inactive->noun
         m_ui = inactive->ui
         m_qtyship = inactive->qty_ship
         m_qtyrcvd = inactive->qty_rcvd
         m_badqty = inactive->bad_qty
         m_unitcost = inactive->unit_cost
         m_totlcost = inactive->total_cost
         m_dcspcode = inactive->dscp_code
         m_duefoll = inactive->due_follup
         m_lastfoll = inactive->last_foll
         m_actioncd = inactive->actioncode
         m_datecomp = inactive->date_comp
         m_remarks1 = inactive->remarks1
         m_remarks2 = inactive->remarks2
         m_remarks3 = inactive->remarks3
         m_remarks4 = inactive->remarks4
         m_remarks5 = inactive->remarks5
         m_remarks6 = inactive->remarks6
         m_name = inactive->name_rank
         m_title = inactive->duty_title
         EXIT
        ELSE
     message = "REPORT NUMBER DOES NOT EXIST WITHIN DATABASE"
```

```
          DO displmsg
          DO clearmsg
          @ 20, 0 TO 20, 79 DOUBLE
        ENDIF
    ENDIF
ENDDO
RETURN

*************************************************************
* Procedure...: SHOWREST                                    *
* Purpose.....: Displays data contain in database fields.   *
*************************************************************
PROCEDURE showrest

* Display data.
@  6, 10 SAY m_dptname
@  6, 51 SAY m_shipname
@  7, 10 SAY m_dptadd
@  7, 51 SAY m_shipadd
@  9, 22 SAY m_reqnnbr
@  9, 62 SAY m_routid
@ 14,  2 SAY m_nsn
@ 14, 20 SAY m_noun
@ 14, 43 SAY m_ui
@ 14, 48 SAY m_qtyship PICTURE "99999"
@ 14, 57 SAY m_qtyrcvd PICTURE "99999"
@ 14, 64 SAY m_badqty PICTURE "99999"
@ 14, 71 SAY m_unitcost PICTURE "99999.99"
@ 16,  2 SAY m_totlcost PICTURE "99999.99"
@ 18, 16 SAY m_dcspcode
@ 18, 29 SAY m_actioncd
@ 18, 37 SAY m_duefoll PICTURE "@D"
@ 18, 51 SAY m_lastfoll PICTURE "@D"
@ 18, 69 SAY m_datecomp PICTURE "@D"
RETURN

*************************************************************
* Procedure...: DRECORD                                     *
* Called From.: interact                                    *
* Purpose.....: Verifies user's choice of reference number  *
*     to be deleted from CLOSED database.                   *
*************************************************************
PROCEDURE drecord

* Clear and draw input screen.
CLEAR
@  2, 27 SAY "ROD DELETE SCREEN"
@  4, 15 TO 8, 60 DOUBLE
@  6, 17 SAY "Reference number to be deleted:"
@  6, 43 SAY m_rptnbr PICTURE "99999999"

SET COLOR TO R+
```

```
@ 18, 1 SAY SPACE(1)
WAIT "ARE YOU SURE? (Y or N)" TO delchk
SET COLOR TO W/B
IF UPPER(delchk) # "Y"
    CLEAR
    m_rptnbr = 0
    EXIT
ENDIF

IF UPPER(delchk) = "Y"
    DELETE
    PACK
ENDIF

RETURN
*************************************************************
* Procedure.. : DISPLMSG                                    *
* Called From.: Several procedures                          *
* Purpose.....: To display the contents of the variable     *
*    called "message".                                      *
*************************************************************
PROCEDURE displmsg

DO clearmsg
SET COLOR TO R+
@ 19, 10 SAY message
SET COLOR TO W/B
WAIT SPACE(10) + "Press any key to continue..."
RETURN

*************************************************************
* Procedure...: CLEARMSG                                    *
* Called From.: Several procedures                          *
* Purpose.....: To clear any message that is present on line*
*   19 on the data entry screen.                            *
*************************************************************
PROCEDURE clearmsg

@ 19, 10 SAY SPACE(68)
RETURN
```

```
*************************************************************
* Program...: RPTMENU.PRG                                   *
* Author....: Capt James L. Johnson                         *
* Date......: 6/30/1989                                     *
* Note......: This program produces a menu that provides the *
*   user the option of producing up to five reports.  If the*
*   user selects the number 0, the system returns to the    *
*   options menu.                                           *
*************************************************************
CLEAR

* Loop as long as user desires another action.
DO WHILE .T.
  * Display customer menu upon screen.
  CLEAR
  @  5, 26 SAY "REPORT OPTIONS MENU"
  @  9, 22 SAY "1. Open RODs by Report Number"
  @ 10, 22 SAY "2. Open RODs by Requisition Number"
  @ 11, 22 SAY "3. Open RODs by Date Due Follow-up"
  @ 12, 22 SAY "4. Closed/Completed RODs"
  @ 13, 22 SAY "5. Total Records and Dollar Value"
  @ 15, 22 SAY "0. EXIT"
  @ 17, 26 SAY "Select"

  * Put double line around menu and use double line to
  *   separate title from menu choice.
  @ 4,0 TO 18,79 DOUBLE
  @ 6,1 TO 6,78 DOUBLE

  * Initialize selectnum to 0.  Prompt user for choice.
  selectnum = 0
  @ 17, 34 GET selectnum PICTURE "9" RANGE 0,5
  READ
  @ 17, 34 SAY selectnum
  CLEAR

  * Take appropriate action based on value of selectnum.
  IF selectnum = 0
    EXIT
  ENDIF

  IF selectnum = 1
    * List open RODs.
    @ 19, 10 SAY "Loading program, please wait"
    DO listrod
  ENDIF

  IF selectnum = 2
    * List open RODs by requisition number.
    @ 19, 10 SAY "Loading program, please wait"
    DO reqnseq
  ENDIF
```

```
  IF selectnum = 3
    * List open RODs by date due follow-up.
    @ 19, 10 SAY "Loading program, please wait"
    DO pastsus
  ENDIF

  IF selectnum = 4
    * List close/completed RODs.
    @ 19, 10 SAY "Loading program, please wait"
    DO comprod
  ENDIF

  IF selectnum = 5
    * Compute total number of records and dollar value.
    @ 19, 10 SAY "Loading program, please wait"
    DO totals
  ENDIF
ENDDO
RETURN
```

```
*************************************************************
* Program..: PRINTROD.PRG                                   *
* Author...: Capt James L. Johnson                          *
* Date.....: 6/30/1989                                      *
* Note.....: Program prints a copy of the ROD after the user*
*    specifies the report number.                           *
*************************************************************
*************************************************************
* Procedure..: INTERACT                                     *
* Purpose....: This is the main procedure within this       *
*   procedure file.  It contains the overall interaction    *
*   with the user in adding records to the ROD file.        *
*************************************************************
PROCEDURE interact

* Open database file.
USE rod_file INDEX i_refnbr

* Continuous loop until user enters 0 at which time the
*   program exits the loop.
DO WHILE .T.
  CLEAR
  @  1, 28 SAY "ROD PRINT SCREEN"
  @  1, 70 SAY DATE()
  @  3, 26 SAY "Report Number:"
  @  3, 52 SAY "(0 TO EXIT)"
  @  5, 2  SAY "Depot:"
  @  5, 42 SAY "Shipper"
  @  6, 2  SAY "Name"
  @  6, 42 SAY "Name"
  @  7, 2  SAY "Address"
  @  7, 42 SAY "Address"
  @  9, 2  SAY "Requisition Number"
  @  9, 42 SAY "Routing Identifier"
  @ 11, 26 SAY "DISCREPANCY DATA"
  @ 12, 48 SAY "Qty"
  @ 12, 57 SAY "Qty"
  @ 12, 64 SAY "Disc"
  @ 12, 71 SAY "Unit"
  @ 13, 2  SAY "NSN/Part Number   Nomenclature"
  @ 13, 43 SAY "UI"
  @ 13, 48 SAY "Shipped"
  @ 13, 57 SAY "Rcvd"
  @ 13, 64 SAY "Qty"
  @ 13, 71 SAY "Cost"
  @ 16, 2  SAY "Total"
  @ 16, 12 SAY "Discrepancy"
  @ 16, 27 SAY "Action"
  @ 16, 37 SAY "Date Due"
  @ 16, 51 SAY "Date Last"
  @ 16, 70 SAY "Date"
  @ 17,  2 SAY "Cost"
```

```
@ 17, 15 SAY "Code"
@ 17, 28 SAY "Code"
@ 17, 37 SAY "Follow-up"
@ 17, 51 SAY "Follow-up"
@ 17, 64 SAY "Completed/Closed"

* Draw lines to separate input data.
@  2, 0 TO 20, 79 DOUBLE
@  4, 1 TO  4, 78 DOUBLE
@ 10, 1 TO 10, 78 DOUBLE
@ 15, 1 TO 15, 78

* Initialize memory variables for user inputs.
STORE 0 TO m_rptnbr
STORE SPACE(30) TO m_dptname
STORE SPACE(30) TO m_dptadd
STORE SPACE(28) TO m_shipname
STORE SPACE(28) TO m_shipadd
STORE SPACE(2) TO m_ui
STORE 0 TO m_qtyship
STORE 0 TO m_qtyrcvd
STORE 0 TO m_badqty
STORE SPACE(15) TO m_reqnnbr
STORE SPACE(3) TO m_routid
STORE SPACE(15) TO m_nsn
STORE SPACE(20) TO m_noun
STORE 0.00 TO m_unitcost
STORE 0.00 TO m_totlcost
STORE SPACE(2) TO m_dcspcode
STORE SPACE(3) TO m_duefoll
STORE SPACE(8) TO m_lastfoll
STORE SPACE(2) TO m_actioncd
STORE SPACE(8) TO m_datecomp
STORE SPACE(76) TO m_remarks1
STORE SPACE(76) TO m_remarks2
STORE SPACE(76) TO m_remarks3
STORE SPACE(76) TO m_remarks4
STORE SPACE(76) TO m_remarks5
STORE SPACE(76) TO m_remarks6
STORE SPACE(50) TO m_name
STORE SPACE(50) TO m_title

* Use initialized variables To clear the screen.
@  3, 41 SAY m_rptnbr PICTURE "99999999"
@ 6, 10 SAY m_dptname
@  6, 51 SAY m_shipname
@  7, 10 SAY m_dptadd
@  7, 51 SAY m_shipadd
@  9, 22 SAY m_reqnnbr
@  9, 62 SAY m_routid
@ 14,  2 SAY m_nsn
@ 14, 20 SAY m_noun
```

```
  @ 14, 43 SAY m_ui
  @ 14, 48 SAY m_qtyship
  @ 14, 57 SAY m_qtyrcvd
  @ 14, 64 SAY m_badqty
  @ 14, 71 SAY m_unitcost PICTURE "99999.99"
  @ 18,  2 SAY m_totlcost PICTURE "99999.99"
  @ 18, 16 SAY m_dcspcode
  @ 18, 29 SAY m_actioncd
  @ 18, 37 SAY m_duefoll PICTURE "@D"
  @ 18, 51 SAY m_lastfoll PICTURE "@D"
  @ 18, 69 SAY m_datecomp PICTURE "@D"

  * Get report number from the user.
  DO getrpt

  * If report number = 0, exit loop
  IF m_rptnbr = 0
    EXIT
  ENDIF

  * Display information on screen.
  DO showinfo

  * Prompt user to decide whether to update database.
  SET COLOR TO R+
  WAIT "Print copy of ROD? (Y or N)" TO userresp
  SET COLOR TO W/B
  IF UPPER(userresp) # "Y"
    EXIT
  ENDIF

  * If the user input "Y", then print ROD.
  IF UPPER(userresp) = "Y"
    DO prntrod
  ENDIF
ENDDO

CLEAR
CLOSE DATABASES
RETURN

*************************************************************
* Procedure....: GETRPT                                     *
* Called From.: INTERACT                                    *
* Purpose.....: Obtain and validate the report number from  *
*    the user.                                              *
*************************************************************
PROCEDURE getrpt

DO WHILE .T.
  * Obtain a reference number from the user and display.
  @  3, 41 GET m_rptnbr PICTURE "99999999"
```

```
READ
@  3, 41 SAY m_rptnbr PICTURE "99999999"

* If report number = 0, exit from loop.  If not 0, then
*   attempt to find the reference number.  If found,
*   display message and exit the loop.
IF m_rptnbr = 0
  EXIT
 ELSE
  SEEK m_rptnbr
  IF FOUND ()
    m_dptname = rod_file->depotname
    m_shipname = rod_file->ship_name
   m_dptadd = rod_file->depot_add
   m_shipadd = rod_file->ship_add
   m_routid = rod_file->routing_id
   m_reqnnbr = rod_file->req_number
   m_nsn = rod_file->nsn_or_pn
   m_noun = rod_file->noun
   m_ui = rod_file->ui
   m_qtyship = rod_file->qty_ship
   m_qtyrcvd = rod_file->qty_rcvd
   m_badqty = rod_file->bad_qty
   m_unitcost = rod_file->unit_cost
   m_totlcost = rod_file->total_cost
   m_dcspcode = rod_file->dscp_code
   m_duefoll = rod_file->due_follup
   m_lastfoll = rod_file->last_foll
   m_actioncd = rod_file->actioncode
   m_datecomp = rod_file->date_comp
    m_remarks1 = rod_file->remarks1
   m_remarks2 = rod_file->remarks2
   m_remarks3 = rod_file->remarks3
   m_remarks4 = rod_file->remarks4
   m_remarks5 = rod_file->remarks5
   m_remarks6 = rod_file->remarks6
   m_name = rod_file->name_rank
   m_title = rod_file->duty_title
   EXIT
  ELSE
  message = "REPORT NUMBER DOES NOT EXIST WITHIN DATABASE"
    DO displmsg
    DO clearmsg
    @ 20, 0 TO 20, 79 DOUBLE
  ENDIF
 ENDIF
ENDDO

RETURN
```

```
*****************************************************************
* Procedure....: GETINFO                                        *
* Called From..: INTERACT                                       *
* Purpose......: To obtain and verify all other fields other*
*    than the key field.                                        *
*****************************************************************
PROCEDURE showinfo

* Display data
@  6, 10 SAY m_dptname
@  6, 51 SAY m_shipname
@  7, 10 SAY m_dptadd
@  7, 51 SAY m_shipadd
@  9, 22 SAY m_reqnnbr PICTURE "AA999999999999A"
@  9, 62 SAY m_routid
@ 14, 2  SAY m_nsn
@ 14, 20 SAY m_noun
@ 14, 43 SAY m_ui
@ 14, 48 SAY m_qtyship PICTURE "99999"
@ 14, 57 SAY m_qtyrcvd PICTURE "99999"
@ 14, 64 SAY m_badqty PICTURE "99999"
@ 14, 71 SAY m_unitcost PICTURE "99999.99"
@ 18,  2 SAY m_totlcost PICTURE "99999.99"
@ 18, 16 SAY m_dcspcode
@ 18, 29 SAY m_actioncd PICTURE "9A"
@ 18, 37 SAY m_duefoll PICTURE "@D"
@ 18, 51 SAY m_lastfoll PICTURE "@D"
@ 18, 69 SAY m_datecomp PICTURE "@D"
RETURN

*****************************************************************
* Procedure...: PRNTROD                                         *
* Called From.: Interact                                        *
* Purpose.....: To print copy of a file.                        *
*****************************************************************
PROCEDURE prntrod

* Initialize varibles.
STORE SPACE(36) TO m_unit
STORE SPACE(36) TO m_base

* Get into concerning user's organization.
CLEAR
@  4, 31 SAY "BASE INFORMATION"
@  6, 19 TO 12, 69 DOUBLE
@  8, 22 SAY "Unit    :"
@  8, 32 SAY m_unit
@ 10, 22 SAY "Address:"
@ 10, 32 SAY m_base
@  8, 32 GET m_unit
@ 10, 32 GET m_base
READ
```

```
CLEAR

* Display message while printing.
@ 10, 22 SAY "Printing ROD, please wait"

SET DEVICE TO PRINT

* Print information.
@  4, 45 SAY DATE()
@  4, 62 SAY rod_file->rpt_number
@  7,  4 SAY rod_file->depotname
@  7, 44 SAY m_unit
@  8,  4 SAY rod_file->depot_add
@  8, 44 SAY m_base
@ 10,  4 SAY rod_file->ship_name
@ 11,  4 SAY rod_file->ship_add
@ 14, 60 SAY rod_file->req_number
@ 20,  4 SAY rod_file->nsn_or_pn
@ 20, 32 SAY rod_file->ui
@ 20, 36 SAY rod_file->qty_ship
@ 20, 42 SAY rod_file->qty_rcvd
@ 20, 49 SAY rod_file->bad_qty
@ 20, 54 SAY rod_file->unit_cost
@ 20, 69 SAY rod_file->dscp_code
@ 20, 73 SAY rod_file->actioncode
@ 21,  4 SAY rod_file->noun
@ 21, 61 SAY rod_file->total_cost
@ 28,  4 SAY rod_file->remarks1
@ 29,  4 SAY rod_file->remarks2
@ 30,  4 SAY rod_file->remarks3
@ 31,  4 SAY rod_file->remarks4
@ 32,  4 SAY rod_file->remarks5
@ 33,  4 SAY rod_file->remarks6
@ 53,  4 SAY rod_file->name_rank
@ 54,  4 SAY rod_file->duty_title
EJECT
SET DEVICE TO SCREEN
RETURN

************************************************************
* Procedure....: DISPLMSG                                  *
* Called From.: Several procedures                         *
* Purpose.....: To display the contents of the variable    *
*   called "message"                                       *
************************************************************
PROCEDURE displmsg

DO clearmsg
SET COLOR TO R+
@ 19, 10 SAY message
SET COLOR TO W/B
WAIT SPACE(10) + "Press any key to continue..."
```

```
RETURN

*************************************************************
* Procedure...: CLEARMSG                                    *
* Called From.: Several procedures                          *
* Purpose.....: To clear any message that is present on line*
*   19 on the data entry screen.                            *
*************************************************************
PROCEDURE clearmsg

@ 19, 10 SAY SPACE(68)
RETURN
```

```
*************************************************************
* Program...: LISTROD.PRG                                   *
* Author....: Capt James L. Johnson                         *
* Date......: 6/30/89                                       *
* Purpose...: Program produces a listing of open ROD        *
*    files in report number sequence.                       *
*************************************************************
CLEAR

USE rod_file

@ 10, 25 SAY "Counting records, please wait..."
COUNT TO m_count
CLEAR
IF m_count = 0
  @ 18, 20 SAY "Cannot produce report, no files exist"
  WAIT SPACE(20) + "Press any key to continue..."
  CLEAR
  CLOSE DATABASES
  RETURN
ENDIF

IF m_count = 1
  COPY TO TEMP
 ELSE
  @ 10, 25 SAY "Sorting data, please wait"
  SORT ON rpt_number TO TEMP
ENDIF

USE TEMP

* Initialize page number to 1.
pagenum = 1
linenum = 70

* Initialize variable
STORE SPACE(1) TO m_choice

* Prompt user for choice of screen or printer.
CLEAR
@  2, 27 SAY "PRINTER OPTION"
@  4, 15 TO 8, 52 DOUBLE
@  6, 17 SAY "Send output to printer (Y or N)"
@  6, 50 SAY m_choice
@  6, 50 GET m_choice PICTURE "A"
READ
@  6, 50 SAY m_choice

IF UPPER(m_choice) = "Y"
  * Direct output to printer.
  SET DEVICE TO PRINT
 ELSE
```

```
  m_choice = "N"
  SET DEVICE TO SCREEN
ENDIF

CLEAR

* Loop until end of file.
DO WHILE .NOT. EOF()
  * If more than 55 lines have been printed, print page
  *    heading, set line number to 7, and increase page
  *    number.
  IF linenum > 55
    @  1,  2 SAY DATE()
    @  1, 25 SAY "Report of Discrepancy"
    @  1, 72 SAY "Page"
    @  1, 77 SAY pagenum PICTURE "99"
    @  2, 30 SAY "Master List"
    @  3, 26 SAY "(By Report Number)"
    @  4, 0  SAY "Report"
    @  4, 11 SAY "Requisition"
    @  4, 46 SAY "Dscp"
    @  4, 53 SAY "Total"
    @  4, 60 SAY "Dscp"
    @  4, 65 SAY "Act"
    @  4, 70 SAY "Date Due"
    @  5, 0  SAY "Number"
    @  5, 13 SAY "Number"
    @  5, 28 SAY "NSN/Part nbr"
    @  5, 47 SAY "Qty"
    @  5, 54 SAY "Cost"
    @  5, 60 SAY "Code"
    @  5, 65 SAY "Code"
    @  5, 70 SAY "Follow-up"

    * Draw lines
    @  6,  0 SAY "=================================================="
    @  6, 39 SAY "=======================  ============="
    linenum = 7
    pagenum = pagenum + 1
  ENDIF

  * Print information
  @ linenum,  0 SAY TEMP->rpt_number
  @ linenum, 11 SAY TEMP->req_number
  @ linenum, 28 SAY TEMP->nsn_or_pn
  @ linenum, 45 SAY TEMP->bad_qty
  @ linenum, 52 SAY TEMP->total_cost
  @ linenum, 61 SAY TEMP->dscp_code
  @ linenum, 66 SAY TEMP->actioncode
  @ linenum, 70 SAY TEMP->due_followup

  * Go to next line and read next record
```

```
    linenum = linenum + 1
    IF linenum > 20
      IF m_choice = "N"
        WAIT SPACE(25) + "Press any key to continue..."
        CLEAR
        linenum = 1
      ENDIF
    ENDIF
    SKIP
  ENDDO

  IF UPPER(m_choice) = "Y"
    EJECT
    SET DEVICE TO SCREEN
  ENDIF

  @ 19, 1 SAY SPACE(1)
  WAIT
  CLEAR
  CLOSE DATABASES
  ERASE TEMP.DBF
  RETURN
```

```
*************************************************************
* Program...: REQNSEQ.PRG                                   *
* Author....: Capt James L. Johnson                         *
* Date......: 6/30/89                                       *
* Purpose...: Program produces a listing of open ROD        *
*    files in requisition number sequence.                  *
*************************************************************
CLEAR

USE rod_file

@ 10, 25 SAY "Counting records, please wait..."
COUNT TO m_count
CLEAR
IF m_count = 0
  @ 18, 20 SAY "Cannot produce report, no files exist"
  WAIT SPACE(20) + "Press any key to continue..."
  CLEAR
  CLOSE DATABASES
  RETURN
ENDIF

IF m_count = 1
  COPY TO TEMP
 ELSE
  @ 10, 25 SAY "Sorting data, please wait"
  SORT ON req_number TO TEMP
ENDIF

USE TEMP

* Initialize page number to 1.
pagenum = 1
linenum = 66

* Initialize variable
STORE SPACE(1) TO m_choice

* Prompt user for choice of screen or printer.
CLEAR
@  2, 27 SAY "PRINTER OPTION"
@  4, 15 TO 8, 52 DOUBLE
@  6, 17 SAY "Send output to printer (Y or N)"
@  6, 50 SAY m_choice
@  6, 50 GET m_choice PICTURE "A"
READ
@  6, 50 SAY m_choice

IF UPPER(m_choice) = "Y"
  * Direct output to printer.
  SET DEVICE TO PRINT
 ELSE
```

```
  m_choice = "N"
  SET DEVICE TO SCREEN
ENDIF

CLEAR

GOTO TOP

* Loop until end of file.
DO WHILE .NOT. EOF()
  * If more than 55 lines have been printed, print page
  *    heading, set line number to 7, and increase page
  *    number.
  IF linenum > 55
    @  1,  2 SAY DATE()
    @  1, 25 SAY "Report of Discrepancy"
    @  1, 72 SAY "Page"
    @  1, 77 SAY pagenum PICTURE "99"
    @  2, 30 SAY "Master List"
    @  3, 24 SAY "(Requisition Number Sequence)"
    @  4, 0  SAY "Report"
    @  4, 11 SAY "Requisition"
    @  4, 46 SAY "Dscp"
    @  4, 53 SAY "Total"
    @  4, 60 SAY "Dscp"
    @  4, 65 SAY "Act"
    @  4, 70 SAY "Date Due"
    @  5, 0  SAY "Number"
    @  5, 13 SAY "Number"
    @  5, 28 SAY "NSN/Part nbr"
    @  5, 47 SAY "Qty"
    @  5, 54 SAY "Cost"
    @  5, 60 SAY "Code"
    @  5, 65 SAY "Code"
    @  5, 70 SAY "Follow-up"

    * Draw lines
    @  6,  0 SAY "========================================="
    @  6, 39 SAY "========================================"
    linenum = 7
    pagenum = pagenum + 1
  ENDIF

  * Print information
  @ linenum,  0 SAY TEMP->rpt_number
  @ linenum, 11 SAY TEMP->req_number
  @ linenum, 28 SAY TEMP->nsn_or_pn
  @ linenum, 45 SAY TEMP->bad_qty
  @ linenum, 52 SAY TEMP->total_cost
  @ linenum, 61 SAY TEMP->dscp_code
  @ linenum, 66 SAY TEMP->actioncode
  @ linenum, 70 SAY TEMP->due_follup
```

```
  * Go to next line and read next record
  linenum = linenum + 1
  IF linenum > 20
    IF m_choice = "N"
      WAIT SPACE(25) + "Press any key to continue..."
      CLEAR
      linenum = 1
    ENDIF
  ENDIF
  SKIP
ENDDO

IF UPPER(m_choice) = "Y"
  EJECT
  SET DEVICE TO SCREEN
ENDIF

@ 19, 1 SAY SPACE(1)
WAIT
CLEAR
CLOSE DATABASES
ERASE TEMP.DBF
RETURN
```

```
*************************************************************
* Program...: PASTSUS.PRG                                   *
* Author....: Capt James L. Johnson                         *
* Date......: 6/30/89                                       *
* Purpose...: Program produces a listing of open ROD        *
*    files in date due follow-up sequence.                  *
*************************************************************
CLEAR

USE rod_file

@ 10, 25 SAY "Counting records, please wait..."
COUNT TO m_count
CLEAR
IF m_count = 0
  @ 18, 20 SAY "Cannot produce report, no files exist"
  WAIT SPACE(20) + "Press any key to continue..."
  CLEAR
  CLOSE DATABASES
  RETURN
ENDIF

IF m_count = 1
  COPY TO TEMP
 ELSE
  @ 10, 25 SAY "Sorting data, please wait..."
  SORT ON due_followup TO TEMP
ENDIF

USE TEMP

* Initialize page number to 1.
pagenum = 1
linenum = 70

* Initialize variable
STORE SPACE(1) TO m_choice

* Prompt user for choice of screen or printer.
CLEAR
@  2, 27 SAY "PRINTER OPTION"
@  4, 15 TO 8, 52 DOUBLE
@  6, 17 SAY "Send output to printer (Y or N)"
@  6, 50 SAY m_choice
@  6, 50 GET m_choice PICTURE "A"
READ
@  6, 50 SAY m_choice

IF UPPER(m_choice) = "Y"
  * Direct output to printer.
  SET DEVICE TO PRINT
 ELSE
```

```
  * Direct output to screen.
  m_choice = "N"
  SET DEVICE TO SCREEN
ENDIF

CLEAR

* Loop until end of file.
DO WHILE .NOT. EOF()
  * If more than 55 lines have been printed, print page
  *    heading, set line number to 7, and increase page
  *    number.
  IF linenum > 55
    @  1,  2 SAY DATE()
    @  1, 25 SAY "Report of Discrepancy"
    @  1, 72 SAY "Page"
    @  1, 77 SAY pagenum PICTURE "99"
    @  2, 30 SAY "Master List"
    @  3, 24 SAY "(By Date Due Follow-up)"
    @  4, 0  SAY "Report"
    @  4, 11 SAY "Requisition"
    @  4, 46 SAY "Dscp"
    @  4, 53 SAY "Total"
    @  4, 60 SAY "Dscp"
    @  4, 65 SAY "Act"
    @  4, 70 SAY "Date Due"
    @  5, 0  SAY "Number"
    @  5, 13 SAY "Number"
    @  5, 28 SAY "NSN/Part nbr"
    @  5, 47 SAY "Qty"
    @  5, 54 SAY "Cost"
    @  5, 60 SAY "Code"
    @  5, 65 SAY "Code"
    @  5, 70 SAY "Follow-up"

    * Draw lines
    @  6,  0 SAY "========================================"
    @  6, 39 SAY "========================================"
    linenum = 7
    pagenum = pagenum + 1
  ENDIF

  * Print information
  @ linenum,  0 SAY TEMP->rpt_number
  @ linenum, 11 SAY TEMP->req_number
  @ linenum, 28 SAY TEMP->nsn_or_pn
  @ linenum, 45 SAY TEMP->bad_qty
  @ linenum, 52 SAY TEMP->total_cost
  @ linenum, 61 SAY TEMP->dscp_code
  @ linenum, 66 SAY TEMP->actioncode
  @ linenum, 70 SAY TEMP->due_follup
```

```
  * Go to next line and read next record
  linenum = linenum + 1
  IF linenum > 20
    IF m_choice = "N"
      WAIT SPACE(25) + "Press any key to continue..."
      CLEAR
      linenum = 1
    ENDIF
  ENDIF
  SKIP
ENDDO

IF UPPER(m_choice) = "Y"
  EJECT
  SET DEVICE TO SCREEN
ENDIF

@ 19, 1 SAY SPACE(1)
WAIT
CLEAR
CLOSE DATABASES
ERASE TEMP.DBF
RETURN
```

```
*************************************************************
* Program...: COMPROD.PRG                                   *
* Author....: Capt James L. Johnson                         *
* Date......: 6/30/89                                       *
* Purpose...: Program produces a listing of closed ROD      *
*    files in report number sequence.                       *
*************************************************************
CLEAR

USE inactive

@ 10, 25 SAY "Counting records, please wait..."
COUNT TO m_count
CLEAR
IF m_count = 0
  @ 18, 20 SAY "Cannot produce report, no files exist"
  WAIT SPACE(20) + "Press any key to continue..."
  CLEAR
  CLOSE DATABASES
  RETURN
ENDIF

IF m_count = 1
  COPY TO TEMP
 ELSE
  @ 10, 25 SAY "Sorting data, please wait"
  SORT ON rpt_number TO TEMP
ENDIF

USE TEMP

* Initialize page number to 1.
pagenum = 1
linenum = 70

* Initialize variable
STORE SPACE(1) TO m_choice

* Prompt user for choice of screen or printer.
CLEAR
@  2, 27 SAY "PRINTER OPTION"
@  4, 15 TO 8, 52 DOUBLE
@  6, 17 SAY "Send output to printer (Y or N)"
@  6, 50 SAY m_choice
@  6, 50 GET m_choice PICTURE "A"
READ
@  6, 50 SAY m_choice

IF UPPER(m_choice) = "Y"
  * Direct output to printer.
  SET DEVICE TO PRINT
 ELSE
```

```
  * Direct output to screen.
  m_choice = "N"
  SET DEVICE TO SCREEN
ENDIF

CLEAR

* Loop until end of file.
DO WHILE .NOT. EOF()
  * If more than 55 lines have been printed, print page
  *    heading, set line number to 7, and increase page
  *    number.
  IF linenum > 55
    @  1,  2 SAY DATE()
    @  1, 25 SAY "Report of Discrepancy"
    @  1, 72 SAY "Page"
    @  1, 77 SAY pagenum PICTURE "99"
    @  2, 30 SAY "Master List"
    @  3, 27 SAY "(By Report Number)"
    @  4, 0  SAY "Report"
    @  4, 11 SAY "Requisition"
    @  4, 46 SAY "Dscp"
    @  4, 53 SAY "Total"
    @  4, 60 SAY "Dscp"
    @  4, 65 SAY "Act"
    @  4, 70 SAY "Date"
    @  5, 0  SAY "Number"
    @  5, 13 SAY "Number"
    @  5, 28 SAY "NSN/Part nbr"
    @  5, 47 SAY "Qty"
    @  5, 54 SAY "Cost"
    @  5, 60 SAY "Code"
    @  5, 65 SAY "Code"
    @  5, 70 SAY "Completed"

    * Draw lines
    @  6,  0 SAY "========================================"
    @  6, 39 SAY "========================================"
    linenum = 7
    pagenum = pagenum + 1
  ENDIF

  * Print information
  @ linenum,  0 SAY TEMP->rpt_number
  @ linenum, 11 SAY TEMP->req_number
  @ linenum, 28 SAY TEMP->nsn_or_pn
  @ linenum, 45 SAY TEMP->bad_qty
  @ linenum, 52 SAY TEMP->total_cost
  @ linenum, 61 SAY TEMP->dscp_code
  @ linenum, 66 SAY TEMP->actioncode
  @ linenum, 70 SAY TEMP->date_comp
```

```
  * Go to next line and read next record
  linenum = linenum + 1
  IF linenum > 20
    IF m_choice = "N"
      WAIT SPACE(25) + "Press any key to continue..."
      CLEAR
      linenum = 1
    ENDIF
  ENDIF
  SKIP
ENDDO

IF UPPER(m_choice) = "Y"
  EJECT
  SET DEVICE TO SCREEN
ENDIF

@ 19, 1 SAY SPACE(1)
WAIT
CLEAR
CLOSE DATABASES
ERASE TEMP.DBF
RETURN
```

```
*************************************************************
* Program..: TOTALS.PRG                                     *
* Author...: Capt James L. Johnson                          *
* Date.....: 6/30/89                                        *
* Notes....: This program provides a total count and        *
*   dollar value of all open ROD files.                     *
*************************************************************
CLEAR
USE rod_file
@ 10, 25 SAY "Counting records and computing totals"
COUNT TO m_count
SUM total_cost TO m_total
CLEAR
@ 5, 15 SAY "TOTALS FOR OPEN REPORT OF DISCREPANCIES"
@ 6, 15 TO 6, 53 DOUBLE
@ 8, 15 SAY "Total Number of Records.."
@ 8, 41 SAY m_count
@ 9, 15 SAY "Total Dollar Value......."
@ 9, 41 SAY m_total
@ 19, 1 SAY SPACE(1)
WAIT SPACE(25) + "Press any key to continue..."
CLEAR
CLOSE DATABASES
RETURN
```

Bibliography

Bailey, Capt Jeffery. Automating the Air Force Retail-Level Equipment Management Process: An Application of Microcomputer-Based Information System Techniques. MS thesis, AFIT/GLM/LSM/88S-1. School of Systems and Logistics. Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1988 (AD-A202624).

Bates, Airman First Class Vanessa., Report of Discrepancy Monitor. Personal Interview. 2750th Logistics Squadron/DMSDR, Wright-Patterson AFB OH, 12 October 1988 through 8 December 1988.

Department of The Air Force. Basic Air Force Supply Procedures. AFM 67-1, Vol I, Part One. Washington: HQ USAF, 1 October 1987.

Department of The Air Force. Reporting of Item and Packaging Discrepancies. AFR 400-54. Washington: HQ USAF, 1 October 1980.

Department of The Air Force. USAF Standard Base Supply System. AFM 67-1, Vol II, Part Two. Washington: HQ USAF, 1 March 1988.

Hodge, Bartow and others. Management Information Systems. Reston, Virginia: Reston Publishing Company, Inc., 1984.

Jackson, William T., Report of Discrepancy (ROD) Monitor. Personal interviews. 2750th Logistics Squadron/DMSDR, Wright-Patterson AFB OH, 22 March through 30 May 1989.

Jones, Edward. Using dBASE III Plus. Berkeley CA: Osborne McGraw-Hill, 1987.

Kendall, Sally. Supply Analyst. Telephone interviews. AFLMC/LGS, Gunter AFB AL, 17 October 1988 through 30 May 1989.

Pratt, Philip J. Microcomputer Database Management Using dBase III Plus. Boston: Boyd & Fraser Publishing Company, 1988.

Smith, Forrest E. and Monrak Saengaram. An Analysis of Reports Item Discrepancies Submitted and Processed By Selected Department of Defense Agencies. MS thesis, AFIT/LSSR/80S-55. School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, June 1980 (AD-A089327).

Thompson, Carolyn. Supply Analyst. Personal interviews. 2750th Logistics Squadron/DMSPA, Wright-Patterson AFB OH, 12 December 1988 through 4 May 1989.

Wright, Sandy. Supply Analyst. Personal interview. 2750th Logistics Squadron/DMSPA, Wright-Patterson AFB OH, 22 March 1989.

VITA

Captain James L. Johnson [redacted]

1979 [redacted] enlisted in the United States Air Force. During his enlistment, he served tours at RAF Upper Heyford, England, Columbus AFB, Mississippi, and Maxwell AFB, Alabama. He received a Bachelor of Science degree in Business Administration from Troy State University in June 1984. He was commissioned from OTS in April 1985.

After commissioning, he was assigned to RAF Greenham Common, England, where he served as the Stock Control Section Chief, Materiel Management Branch Chief, and Materiel Storage and Distribution Branch Chief. He received a Master of Science Degree in Business Management from Troy State University, European Division, in February 1988. He entered into the School of Systems and Logistics, Air Force Institute of Technology, in June 1988.

[redacted]

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AFIT/GLM/LSM/89S-33	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION School of Systems and Logistics 6b. OFFICE SYMBOL (If applicable) AFIT/LSM	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) Air Force Institute of Technology Wright-Patterson AFB OH 45433-6583	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
AUTOMATING THE AIR FORCE BASE-LEVEL REPORT OF DISCREPANCY PROGRAM: AN APPLICATION OF DATABASE MANAGEMENT TECHNIQUES

12. PERSONAL AUTHOR(S)
James L. Johnson, B.S., M.S., Captain, USAF

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
MS Thesis	FROM _____ TO _____	1989 September	109

16. SUPPLEMENTARY NOTATION

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP
05	02	
12	05	

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Report of Discrepancy Databases
Shipment Discrepancy Information Systems

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

Thesis Advisor: John E. Sullivan III, Captain, USAF
Instructor
Department of Logistics Management

Approved for public release: IAW AFR 190-1.

Larry W. Emmelhainz

LARRY W. EMMELHAINZ, Lt Col, USAF 14 Oct 89
Director of Research and Consultation
Air Force Institute of Technology (AU)
Wright-Patterson AFB OH 45433-6583

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL John E. Sullivan III	22b. TELEPHONE (Include Area Code) (513) 255-4042 / 22c. OFFICE SYMBOL LSMA

DD Form 1473, JUN 86 *Previous editions are obsolete.* SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

The purpose of this study was to apply principles of database management to the Air Force Report of Discrepancy (ROD) Program. The overall goal was to reduce the number of manhours required to manage the program and to improve the reliability of information. In conducting this study, the researcher asked and answered four research questions. Each question constituted one of four steps in the development of this study. The four steps were to (1) evaluate the current system, (2) identify aspects of the current system that could be improved through automation, (3) choose a database management software (DBMS) package to automate those aspects identified in step 2, and (4) structure and implement a database application.

The study resulted in the development of a database application designed to automate aspects of the ROD program. This program could have universal application to all Air Force base supply organizations. Validation and/or distribution of the programs or findings will be managed by the Logistics Management Center at Gunter AFB, Alabama.