JSC-13064
80-10180
NASA CR-160623

DETAILED DESIGN SPECIFICATION
FOR THE
YIELD ESTIMATION SUBSYSTEM
DATA MANAGEMENT SYSTEM
(YESDAMS)

Job Order 74-963

(E80-10180) DETAILED DESIGN SPECIFICATION FOR THE YIELD ESTIMATION SUBSYSTEM DATA MANAGEMENT SYSTEM (YESDAMS) (Lockheed Electronics Co.) 107 p HC A06/MF A01 CSCL 05B G3/43
N80-28781
Unclas
00180

Prepared By

Lockheed Electronics Company, Inc.
Systems and Services Division
Houston, Texas

Contract NAS 9-15200

For

EARTH OBSERVATIONS DIVISION

SPACE AND LIFE SCIENCES DIRECTORATE

National Aeronautics and Space Administration
LYNDON B. JOHNSON SPACE CENTER
Houston, Texas
July 1977

LEC-11110

JSC-13064

DETAILED DESIGN SPECIFICATION
FOR THE
YIELD ESTIMATION SUBSYSTEM
DATA MANAGEMENT SYSTEM
(YESDAMS)

Job Order 74-963

Design By

T. G. Phillips, Computer Specialist
Center for Climatic and Environmental Assessment

PREPARED BY

R. L. Davenport
R. F. Hansen
K. F. Williams

APPROVED BY

LEC

Phillip L. Krumm

P. L. Krumm, Supervisor
Applications Software Section

NOAA/CCEA

N. D. Strommen, Director
CCEA

NASA

J. L. Dragg

J. L. Dragg, Chief
Applications Analysis Branch

Prepared By

Lockheed Electronics Company, Inc.

For

Earth Observations Division

Space and Life Sciences Directorate

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LYNDON B. JOHNSON SPACE CENTER
HOUSTON, TEXAS

July 1977

LEC-11110

CONTENTS

Section Page

Appendix Page

FIGURES

Figure Page

1. SCOPE

The recent proliferation of Yield Estimation Subsystem (YES) yield models (and their associated data files) dictated the immediate need for a specialized data management system to adequately support the data requirements of the operational YES. This document provides the detailed system design specification of the YES Data Management System (YESDAMS) prior to the commencement of system coding, debugging, and program integration. YESDAMS provides for the basic functions of definition, replacement, addition, deletion, and listing of all data on the YES files.

Backup and recovery procedures will be addressed under separate cover.

2. APPLICABLE DOCUMENTS

● AD 63-1347-4963-18	YES	Data Management System

3. SYSTEM DESCRIPTION

The Yield Estimation Subsystem Data Management System (YESDAMS) is a specialized data management system designed to solve the specific data handling problems of the YES. While the basic functions of definition, replacement, addition, deletion, and listing are common to most data management systems, these functions are designed to address the unique requirements of the YES. Consequently, no attempt has been made to extend any of the functions to the more generalized data management problem. While this somewhat reduces the flexibility of the system, it is felt that the gains in reduced program size and execution time and ease of use far outweigh the loss of flexibility.

Every attempt has been rade to follow the basic tenets of top-down, structured programming using HIPO (Hierarchical Input Processing Output) techniques. The design consists of a main program which initiates a calling sequence several levels deep in order to process input requests. Input transactions are in the form of cards, but the concepts could easily be extended to encompass interactive, time-sharing techniques.

3.1 HARDWARE DESCRIPTION

The programs and associated data files will be resident on the IBM 360/195 complex at Suitland, Maryland. They should be transferable to any IBM 360-370 series computer with sufficient disk space to handle the data files and main memory to support the Pl/I optimizing compiler.

3.2 SOFTWARE DESCRIPTION

The software design consists of a main program and several levels of subprograms (or subroutines), which are called to process the individual input requests. The input requests are interpreted (parsed) at each level in order to determine the subsequent subroutine to be called or action to be taken.

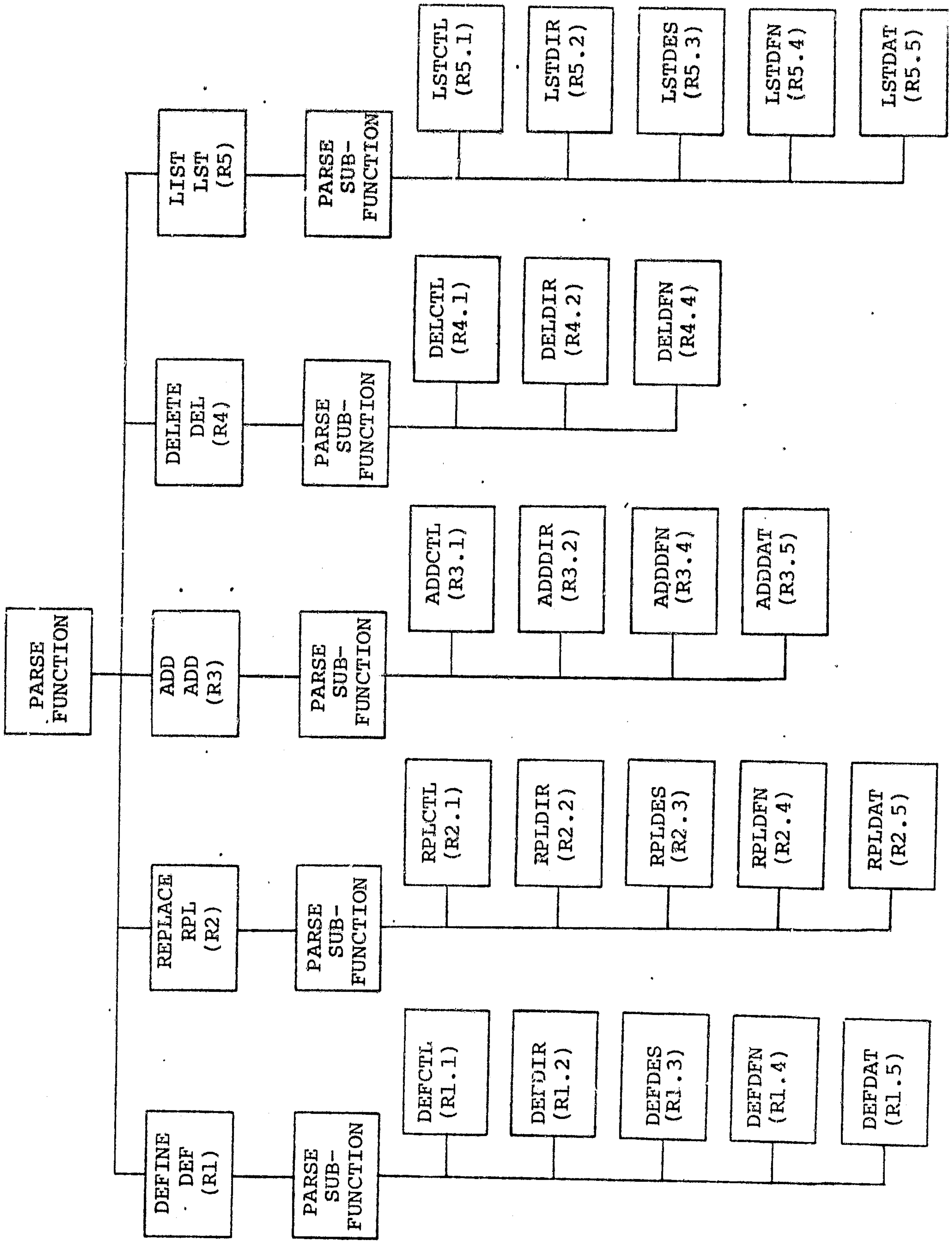


Figure 3.0.- Level 1 and Level 2 subroutines.

3.2.1 PROGRAM MAIN

MAIN reads the input cards and determines which one of the level 1 subroutines is to be called. The level 1 subroutines and their functions are:

ID	Name	Function
R1	DEF	Define
R2	RPL	Replace
R3	ADD	Add
R4	DEL	Delete
R5	LST	List

3.2.1.1 Linkages

MAIN calls subroutines DEF, RPL, ADD, DEL, and LST.

3.2.1.2 Interfaces

MAIN provides the interface between the input request and the proper interpretive subroutine.

3.2.1.3 Inputs

The input to MAIN is an action request in card format.

3.2.1.4 Outputs

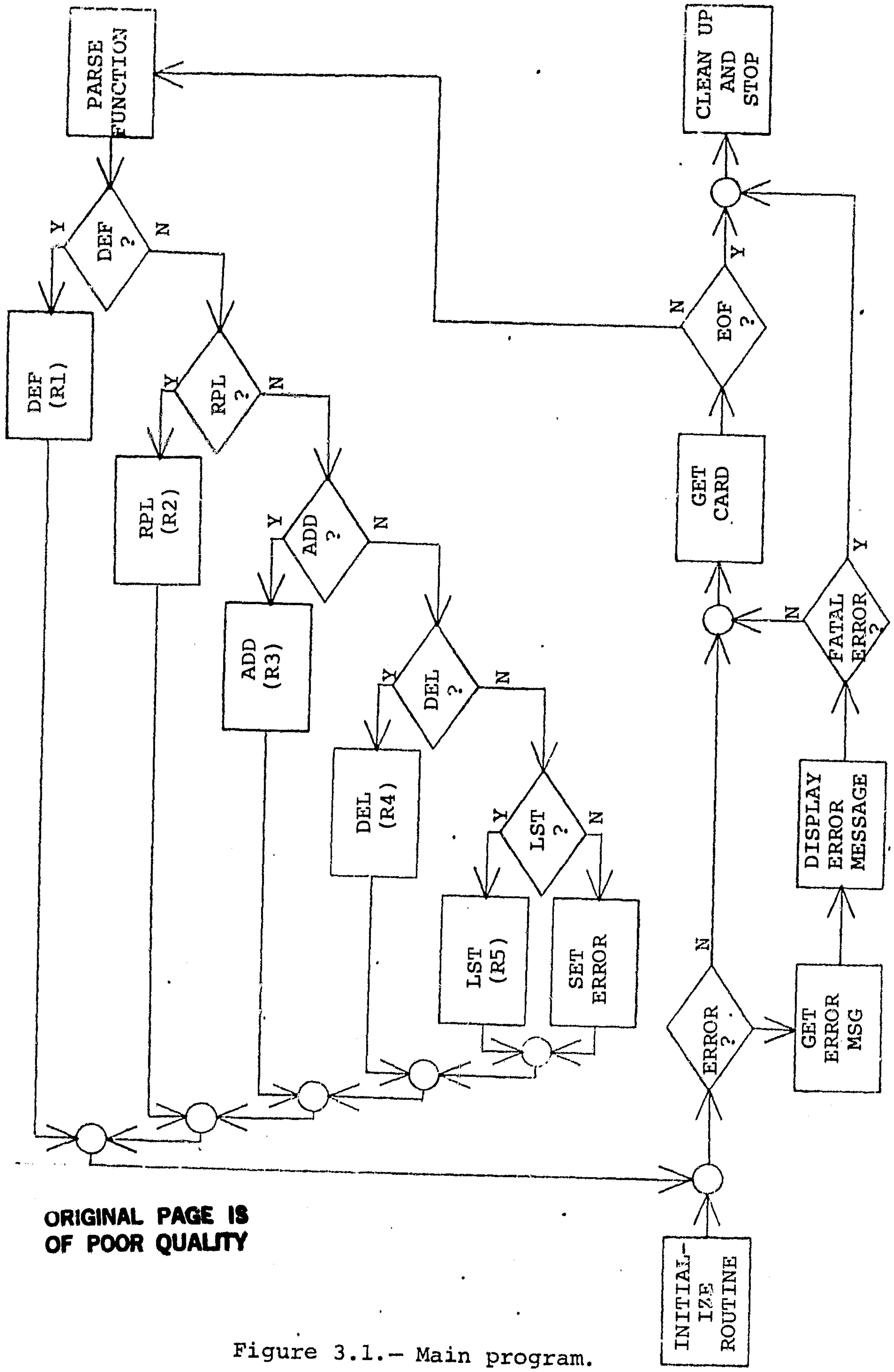
The output from MAIN is a partially parsed action request which is passed to the proper level 1 subroutine for further action and/or an error message which is passed to the system for display.

3.2.1.5 Description

MAIN reads the input action request cards one at a time and decides, by examining the first three characters of the request, which function is being requested. In the case of an error being generated somewhere during processing, MAIN will process the error number and send the proper error message to SYSOUT for display. In the case of a fatal error, MAIN will clean up and stop processing. After all the input cards have been read and processed, MAIN will perform all tasks necessary to clean up and stop processing.

3.2.1.6 Flowchart

Figure 3.1.



ORIGINAL PAGE IS OF POOR QUALITY

Figure 3.1.– Main program.

3.2.2 SUBROUTINE DEF (R1)

DEF (define) takes the input action request passed to it by MAIN and determines which one of the level 2 subroutines is to be called. The level 2 subroutines for DEF and their subfunctions are:

ID	Name	Subfunction
R1.1	DEFCTL	Control
R1.2	DEFDIR	Directory
R1.3	DEFDES	Descriptor
R1.4	DEFDFN	Definition
R1.5	DEFDAT	Data

3.2.2.1 Linkages

DEF calls subroutines DEFCTL, DEFDIR, DEFDES, DESDFN, and DEFDAT. It is called only by MAIN.

3.2.2.2 Interfaces

DEF provides the interface between MAIN and the proper interpretive subroutine.

3.2.2.3 Inputs

The input to DEF is a partially parsed action request.

3.2.2.4 Outputs

The output from DEF is a partially parsed action request which is passed to the proper level 2 subroutine for further action and/or an error message number which is returned to MAIN for action.

3.2.2.5 Description

DEF accepts the partially parsed action request from MAIN and decides, by examining characters 5 to 7 of the request, which subfunction is being requested, and calls that subroutine. If the subfunction being requested is not one of those listed, DEF will set an error condition and return to MAIN.

3.2.2.6 Flowchart

Figure 3.2.

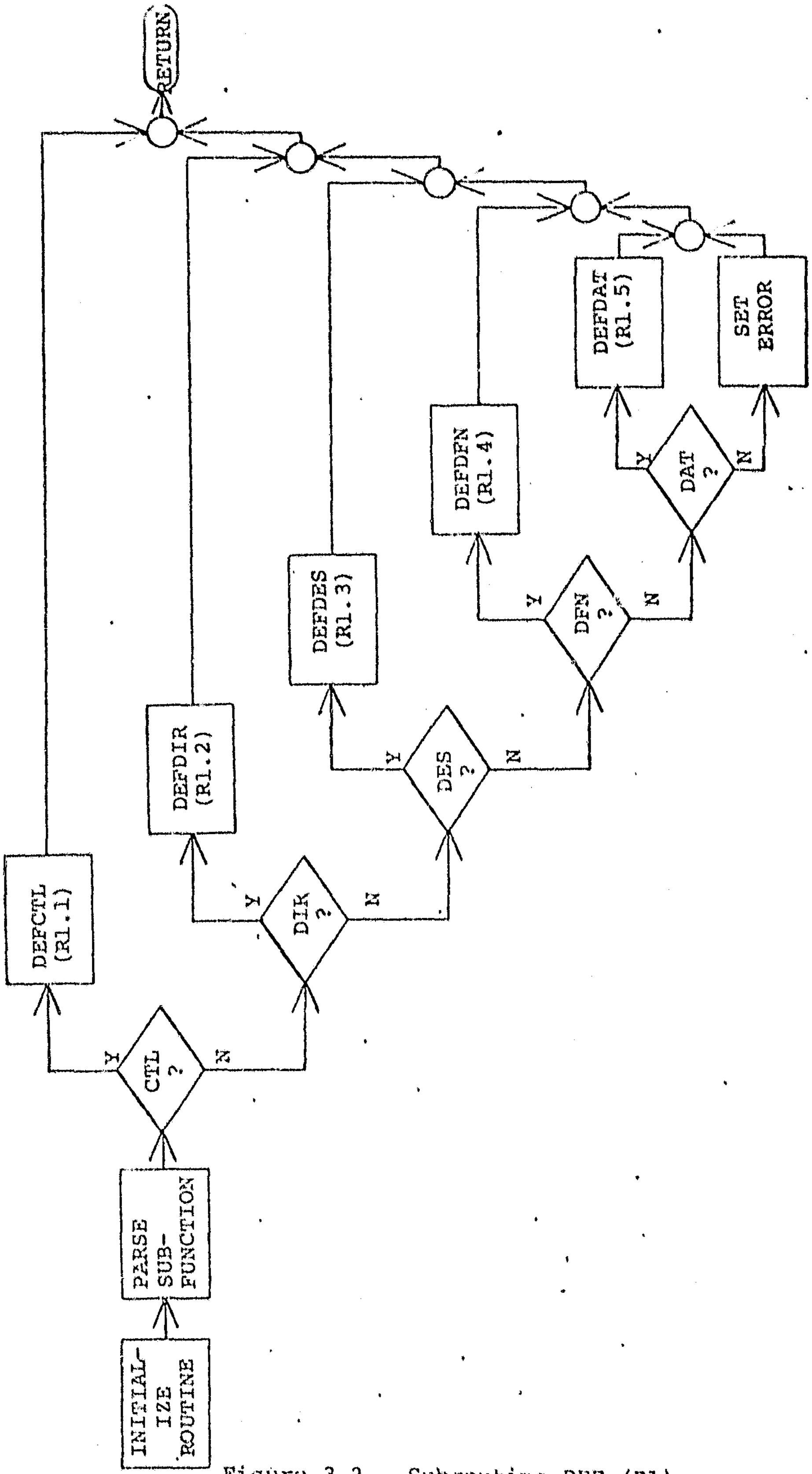


Figure 3.2.— Subroutine DEF (R1).

3.2.3 SUBROUTINE RPL (R2)

RPL (replace) takes the input action request passed to it by MAIN and determines which one of the level 2 subroutines is to be called. The level 2 subroutines for RPL and their subfunctions are:

ID	Name	Subfunction
R2.1	RPLCTL	· Control
R2.2	RPLDIR	Directory
R2.3	RPLDES	Descriptor
R2.4	RPLDFN	Definition
R2-5	RPLDAT	Data

3.2.3.1 Linkages

RPL calls subroutines RPLCTL, RPLDIR, RPLDES, RPLDFN, and RPLDAT. It is called only by MAIN.

3.2.3.2 Interfaces

RPL provides the interface between MAIN and the proper interpretive subroutine.

3.2.3.3 Inputs

The input to RPL is a partially parsed action request.

3.2.3.4 Outputs

The output from RPL is a partially parsed action request which is passed to the proper level 2 subroutine for further action and/or an error message number which is returned to MAIN for action.

3.2.3.5 Description

RPL accepts the partially parsed action request from MAIN and decides, by examining characters 5 to 7 of the request, which subfunction is being requested and calls that subroutine. If the subfunction being requested is not one of those listed, RPL will set an error and return to MAIN.

3.2.3.6 Flowchart

Figure 3.3.

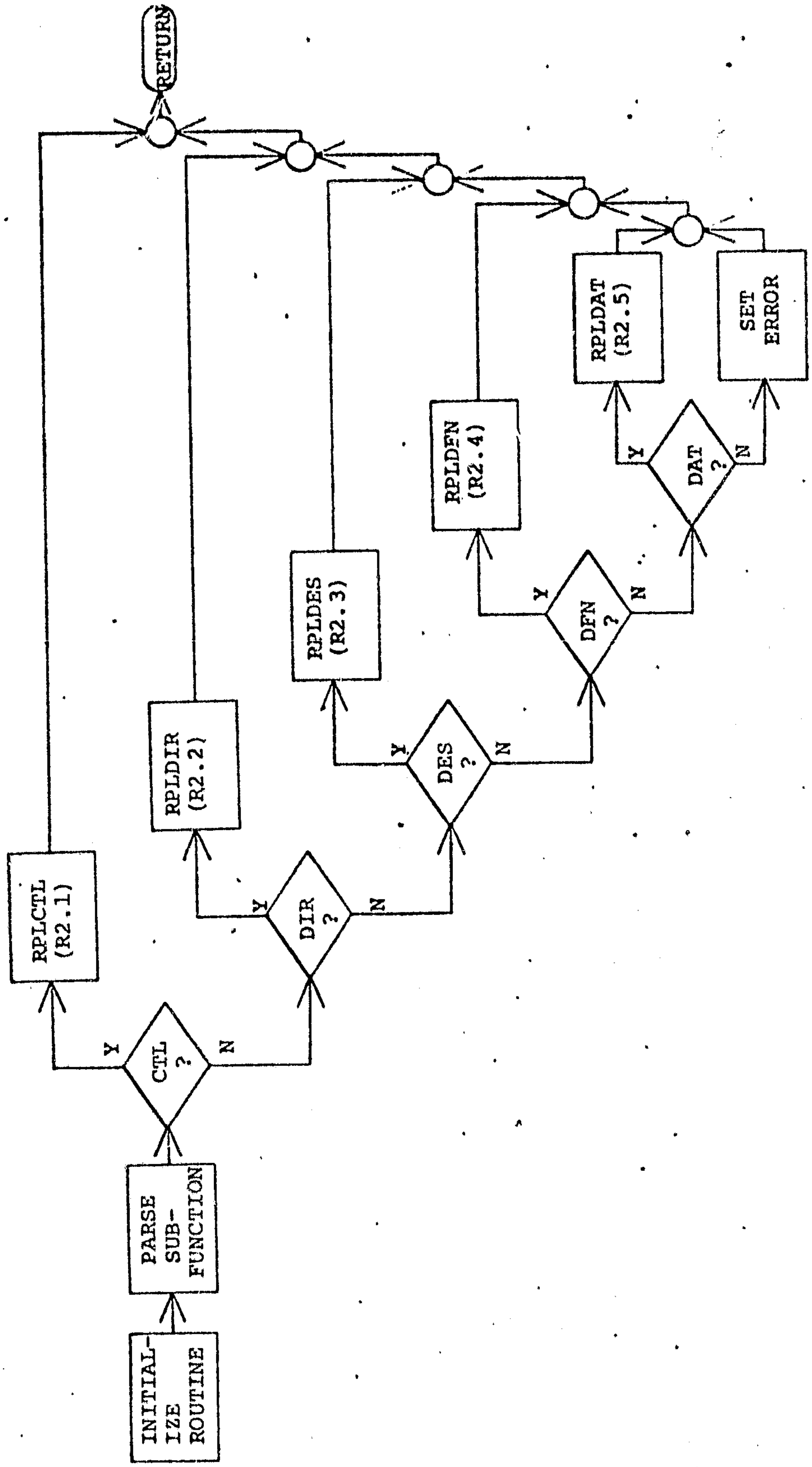


Figure 3.3.– Subroutine RPL (R2).

3.2.4 SUBROUTINE ADD (R3)

ADD (add) takes the input action request passed to it by MAIN and determines which one of the level 2 subroutines is to be called. The level 2 subroutines for ADD and their subfunctions are:

ID	Name	Subfunction
R3.1	ADDCTL	Control
R3.2	ADDDIR	Directory
R3.4	ADDDFN	Definition
R3.5	ADDDAT	Data

3.2.4.1 Linkages

ADD calls subroutines ADDCTL, ADDDIR, ADDDFN, and ADDDAT. It is called only by MAIN.

3.2.4.2 Interfaces

ADD provides the interface between MAIN and the proper interpretive subroutine.

3.2.4.3 Inputs

The input to ADD is a partially parsed action request.

3.2.4.4 Outputs

The output from ADD is a partially parsed action request which is passed to the proper level 2 subroutine for further action and/or an error number which is returned to MAIN for action.

3.2.4.5 Description

ADD accepts the partially parsed action request from MAIN and decides, by examining characters 5 to 7 of the request, which subfunction is being requested and calls that subroutine. If the subfunction being requested is not one of those listed above, ADD will set an error and return to MAIN.

3.2.4.6 Flowchart

Figure 3.4.

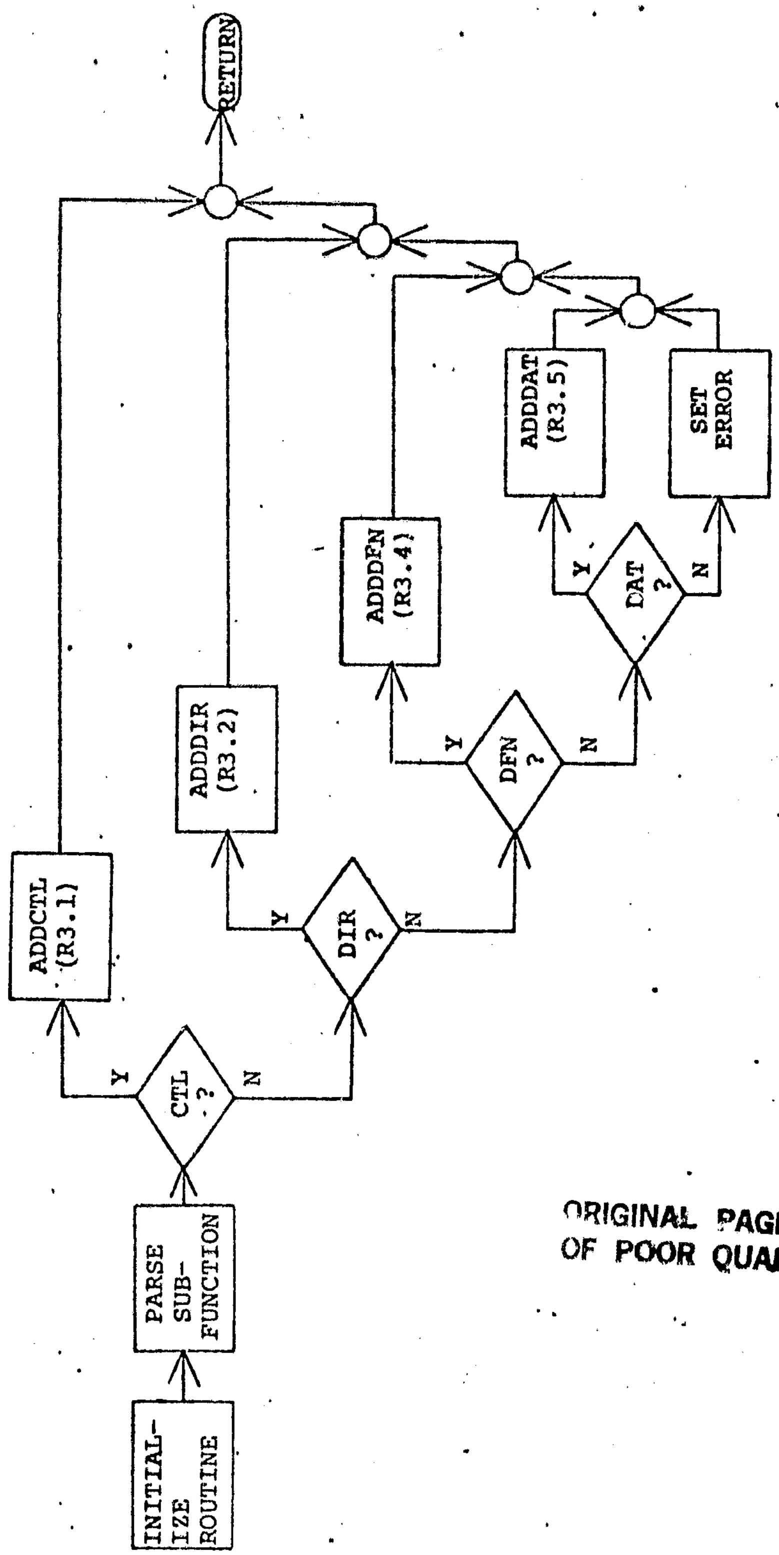


Figure 3.4.— Subroutine ADD (R3).

3.2.5 SUBROUTINE DEL (R4)

DEL (delete) takes the input action request passed to it by MAIN and determines which one of the level 2 subroutines is to be called. The level 2 subroutines for DEL and their subfunctions are:

ID	Name	Subfunctions
R4.1	DELCTL	Control
R4.2	DELDIR	Directory
R4.4	DELDFN	Definition

3.2.5.1 Linkages

DEL calls subroutines DELCTL, DELDIR and DELDFN. It is called only by MAIN.

3.2.5.2 Interfaces

DEL provides the interface between MAIN and the proper interpretive subroutine.

3.2.5.3 Inputs

The input to DEL is a partially parsed action request.

3.2.5.4 Outputs

The output from DEL is a partially parsed action request which is passed to the proper level 2 subroutine for further action and/or an error number which is returned to MAIN for action.

3.2.5.5 Description

DEL accepts the partially parsed action request from MAIN and decides, by examining characters 5 to 7 of the request, which subfunction is being requested and calls that subroutine. If the subfunction being requested is not one of those listed above, DEL will set an error and return to MAIN.

3.2.5.6 Flowchart

Figure 3.5.

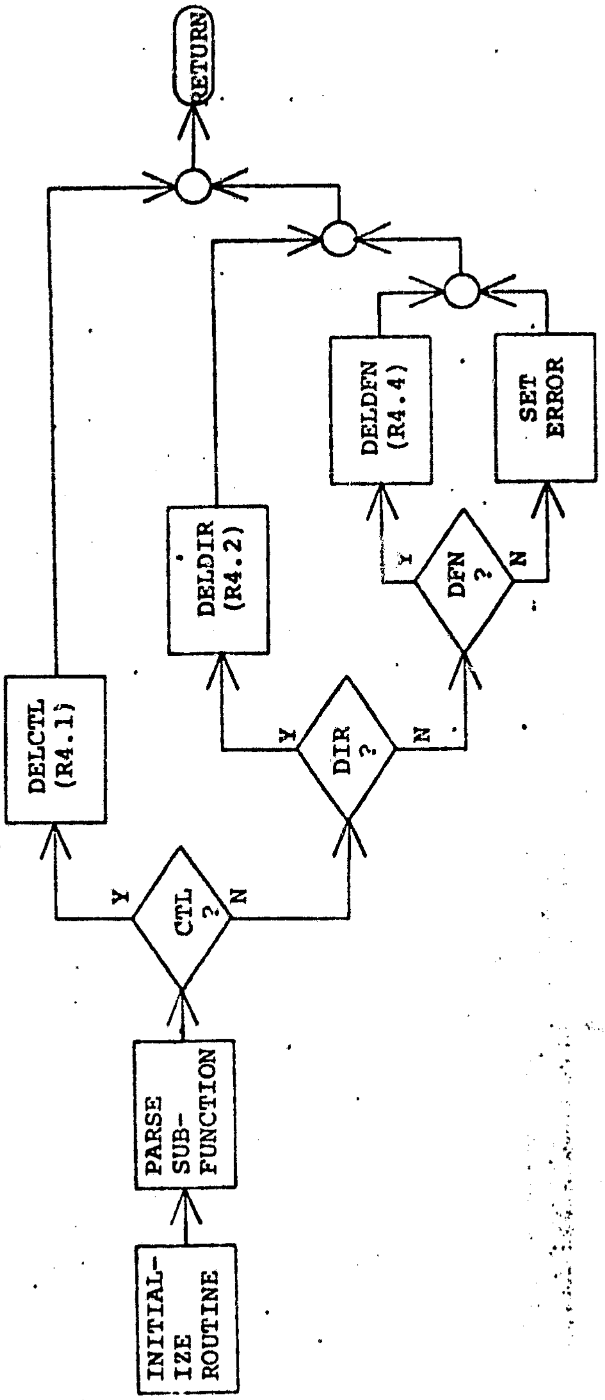


Figure 3.5.— Subroutine DEL (R4).

3.2.6 SUBROUTINE LST (R5)

LST (list) takes the input action request passed to it by MAIN and determines which one of the level 2 subroutines is to be called. The level 2 subroutines for LST and their subfunctions are:

ID	Name	Subfunction
R5.1	LSTCTL	Control
R5.2	LSTDIR	Directory
R5.3	LSTDES	Descriptor
R5.4	LSTDFN	Definition
R5.5	LSTDAT	Data

3.2.6.1 Linkages

LST calls subroutines LSTCTL, LSTDIR, LSTDES, LSTDFN, and LSTDAT. It is called only by MAIN.

3.2.6.2 Interfaces

LST provides the interface between MAIN and the proper interpretive subroutine.

3.2.6.3 Inputs

The input to LST is a partially parsed action request.

3.2.6.4 Outputs

The output from LST is a partially parsed action request which is passed to the proper level 2 subroutine for further action and/or an error number which is returned to MAIN for action.

3.2.6.5 Description

LST accepts the partially parsed action request from MAIN and decides, by examining characters 5 to 7 of the request, which subfunction is being requested and calls that subroutine. If the subfunction being requested is not one of those listed above, LST will set an error and return to MAIN.

3.2.6.6 Flowchart

Figure 3.6.

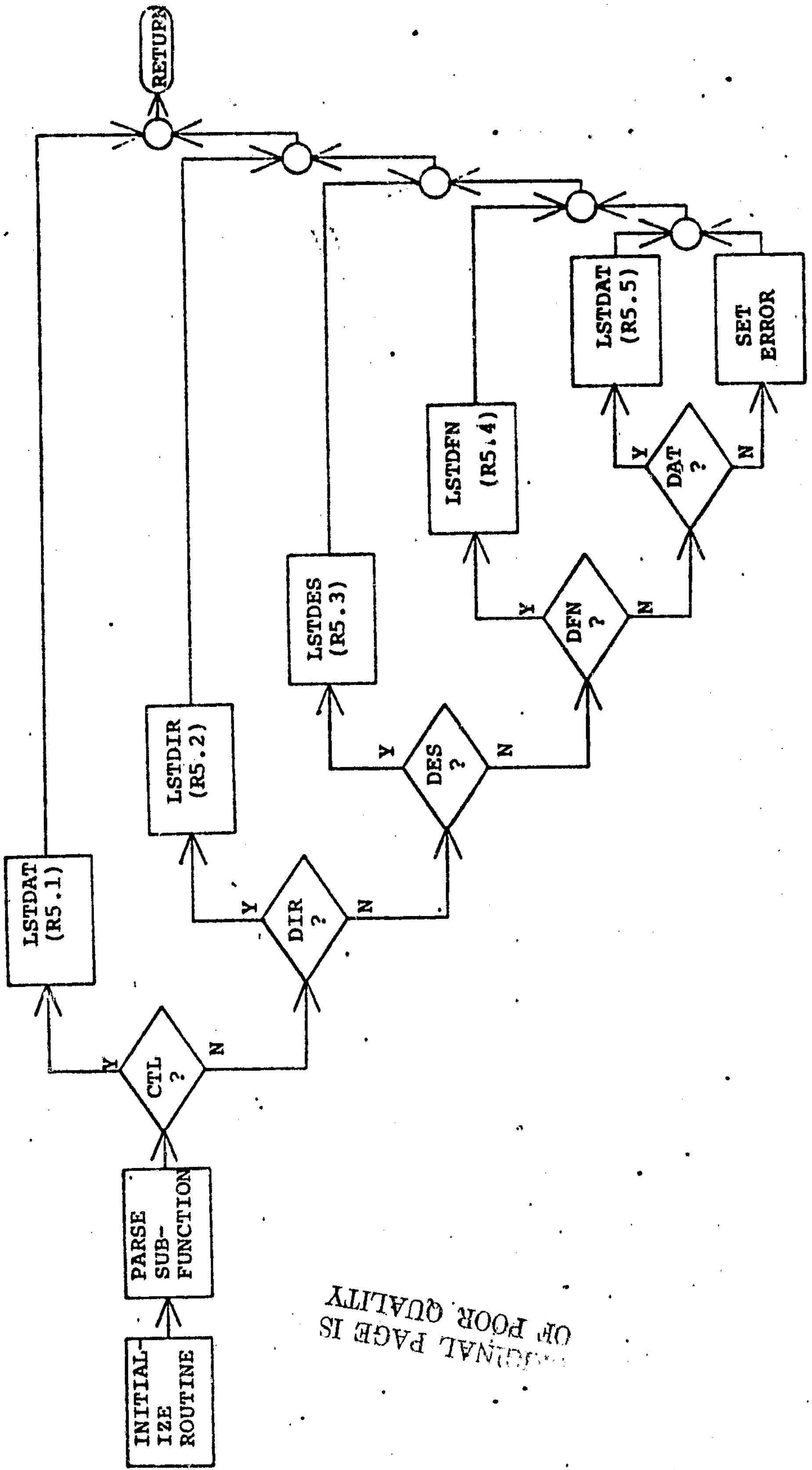


Figure 3.6.– Subroutine LST (R5).

3.2.7 SUBROUTINE DEFCTL (R1.1)

DEFCTL (define control) takes the input action request and parses it for parameter (PARM) and section (SECT) data to build the control record.

3.2.7.1 Linkages

DEFCTL calls subroutines PARPRM (X1), PARSEC (X2), and WRITE (C1). It is called only by DEF.

3.2.7.2 Interfaces

None.

3.2.7.3 Inputs

The input to DEFCTL is a partially parsed action request.

3.2.7.4 Outputs

The output of DEFCTL is a fully defined control record which is written to the data base or an error number.

3.2.7.5 Description

DEFCTL accepts the action request from DEF and interprets it for parameter and section data needed to build the control record. It then writes the newly created control record to the data base and initializes it with blanks. Any attempt to build a control record with incomplete or erroneous data will result in an error condition being returned to DEF.

3.2.7.6 Flowchart

Figure 3.7.

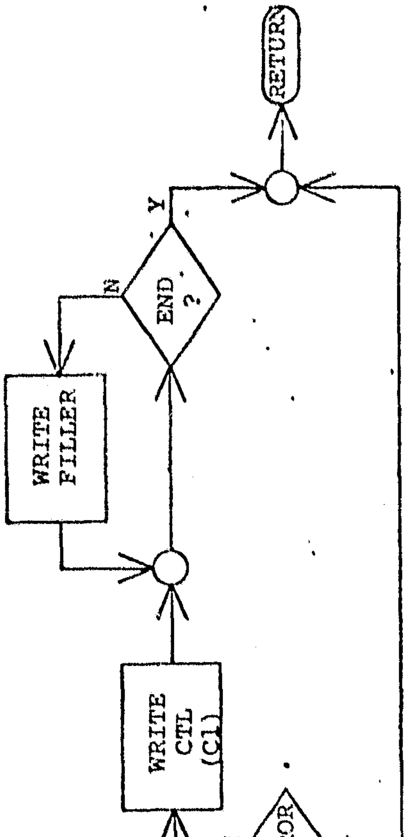


Figure 3.7.- Subroutine DEFCTL (R1.1).

3.2.8 SUBROUTINE DEFDIR (R1.2)

DEFDIR (define directory) takes the input action request and parses it for parameter and section data to build a directory entry.

3.2.8.1 Linkages

DEFDIR calls subroutines PARPRM, PARSEC, and WRITE. It is called only by DEF.

3.2.8.2 Interfaces

DEFDIR updates the proper control record to insure that the newly created directory entry is properly referenced.

3.2.8.3 Inputs

The input to DEFDIR is a partially parsed action request.

3.2.8.4 Outputs

The outputs of DEFDIR are a fully defined directory entry which is written to the data base and an updated directory pointer in the control record or an error number.

3.2.8.5 Description

DEFDIR accepts the action request from DEF and interprets it for parameter and section data needed to build the directory entry. It then writes the newly created directory entry to the data base and updates the directory pointer in the control record and rewrites the record. Any attempt to build a directory entry with incomplete or erroneous data will result in an error condition being returned to DEF.

3.2.8.6 Flowchart

Figure 3.8.

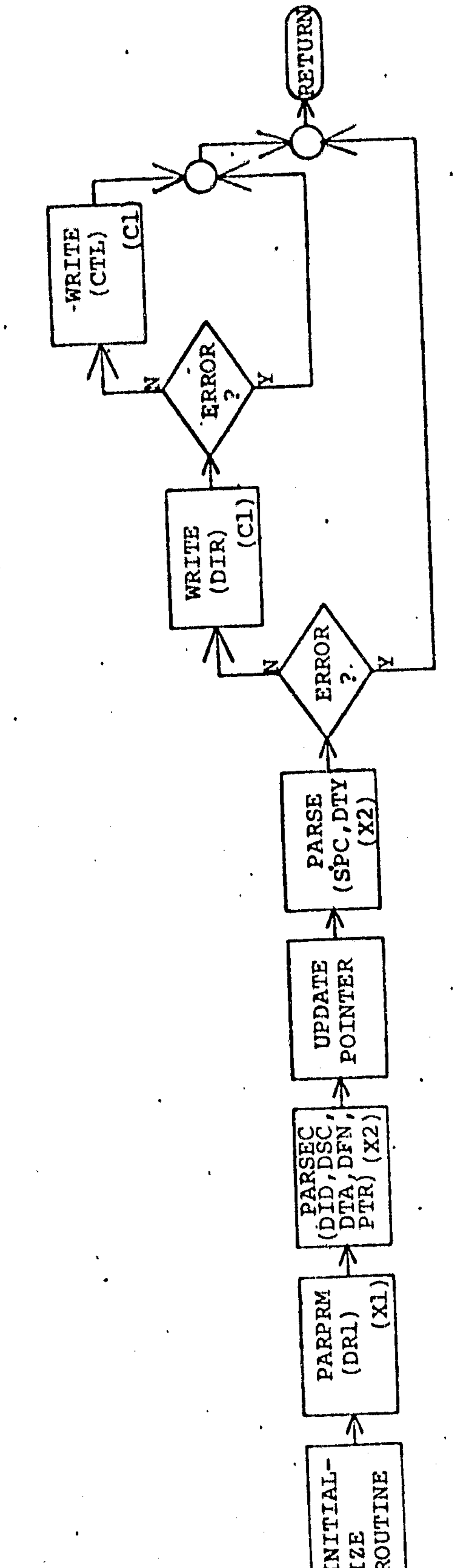


Figure 3.8.- Subroutine DEFDIR (R1.2).

3.2.9 SUBROUTINE DEFDES (R1.3)

DEFDES (define descriptor) takes the input action request and parses it for parameter and section data to build a descriptor entry.

3.2.9.1 Linkages

DEFDES calls subroutines PARPRM, PARSEC, and WRITE. It is called only by DEF.

3.2.9.2 Interfaces

DEFDES updates the proper control record and correct directory entry to insure that the newly created descriptor entry has the proper pointers.

3.2.9.3 Inputs

The input to DEFDES is a partially parsed action request.

3.2.9.4 Outputs

The outputs of DEFDES are a fully defined descriptor entry which is written to the data base and an updated control record and directory entry or an error number.

3.2.9.5 Description

DEFDES accepts the action request from DEF and interprets it for parameter and section data needed to build the descriptor entry. It then writes the newly created descriptor entry to the data base and updates the control record and correct directory entry for proper descriptor pointers, and rewrites them to the data base. Any attempt to build a descriptor entry with incomplete or erroneous data will result in an error condition being returned to DEF.

3.2.9.6 Flowchart

Figure 3.9.

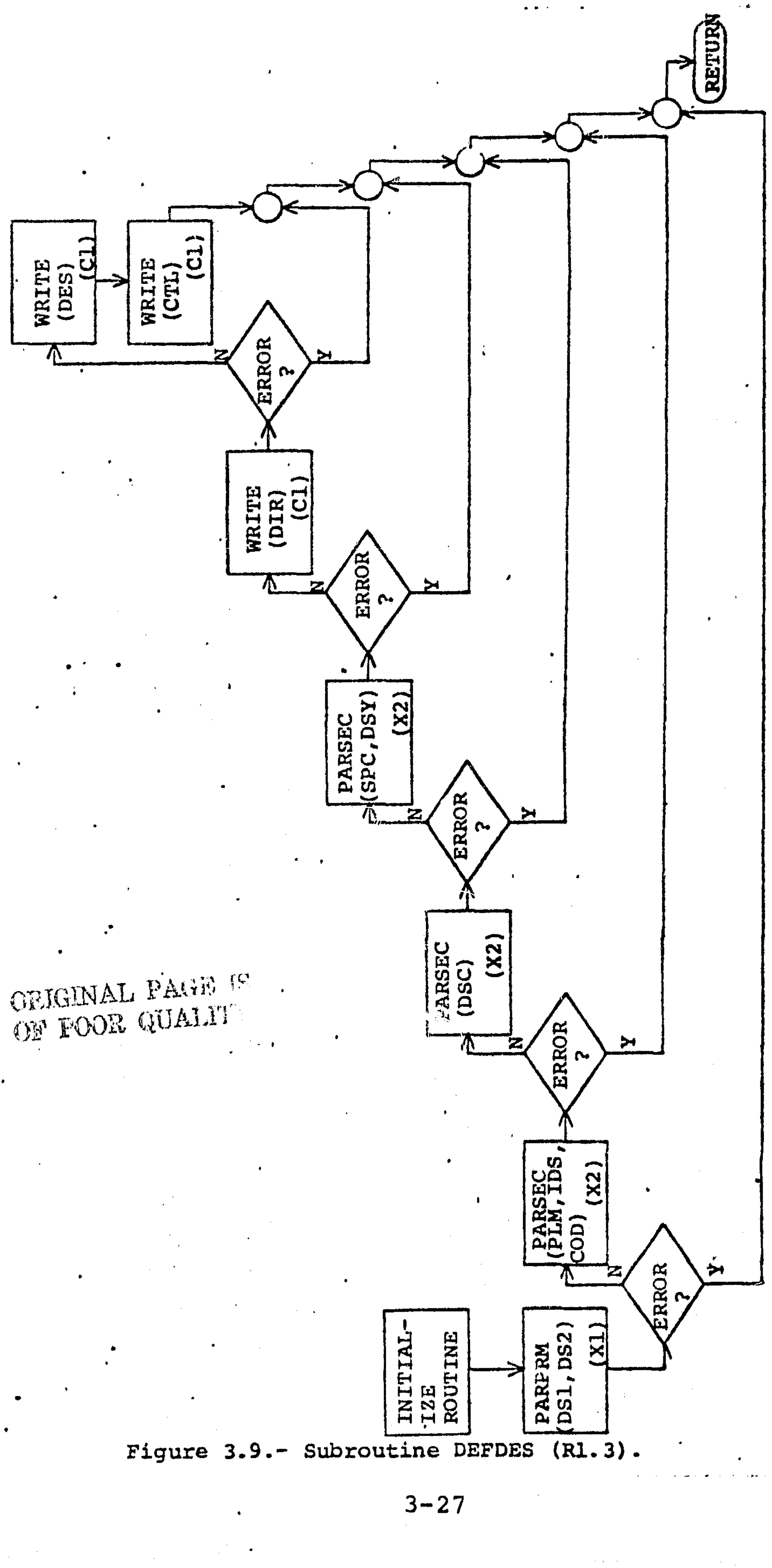


Figure 3.9.- Subroutine DEFDES (R1.3).

3.2.10 SUBROUTINE DEFDFN (R1.4)

DEFDFN (define definition) takes the input action request and parses it for parameter and section data to build a definition record.

3.2.10.1 Linkages

DEFDFN calls subroutines PARPRM, PARSEC, and WRITE. It is called only by DEF.

3.2.10.2 Interfaces

DEFDEN updates the proper control record and correct directory entry to insure that the newly created definition record has the proper pointers.

3.2.10.3 Inputs

The input to DEFDFN is a partially parsed action request.

3.2.10.4 Outputs

The outputs of DEFDFN are a fully defined definition record which is written to the data base and an updated control record and directory entry or an error number.

3.2.10.5 Description

DEFDFN accepts the action request from DEF and interprets it for parameter and section data needed to build the model definition record. It then writes the newly created definition record to the data base and updates the control record and correct directory entry for proper definition pointers and rewrites them to the data base. Any attempt to build model definition record with incomplete or erroneous data will result in an error condition being returned to DEF.

3.2.10.6 Flowchart

Figure 3.10.

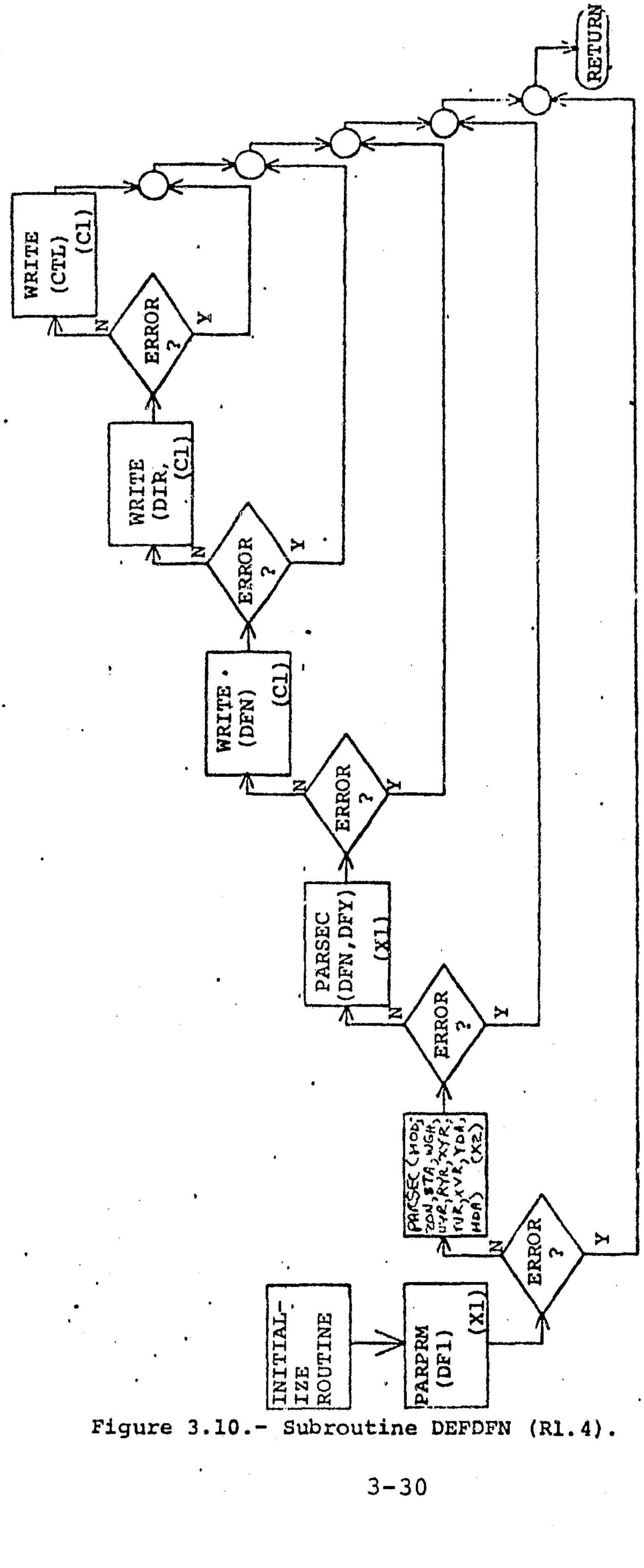


Figure 3.10.- Subroutine DEFDFN (R1.4).

3.2.11 SUBROUTINE DEFDAT (R1.5)

DEFDAT (define data) takes the input action request and parses it for parameter and section data to build the data records.

3.2.11.1 Linkages

DEFDAT calls subroutines PARPRM, PARSEC, and WRITE. It is called only by DEF.

3.2.11.2 Interfaces

DEFDAT updates the proper control record and correct directory entry to insure that the newly created data records have the proper pointers.

3.2.11.3 Inputs

The input to DEFDAT is a partially parsed action request.

3.2.11.4 Outputs

The outputs of DEFDAT are fully defined data records which are written to the data base and an updated control record and directory entry or an error number.

3.2.11.5 Description

DEFDAT accepts the action request from DEF and interprets it for parameter and section data needed to build the data records. It then writes the newly created data records to the data base and updates the control record and correct directory entry for proper data pointers and rewrites them to the data base. Any attempt to build data records with incomplete or erroneous data will result in an error condition being returned to DEF.

3.2.11.6 Flowchart

Figure 3.11.

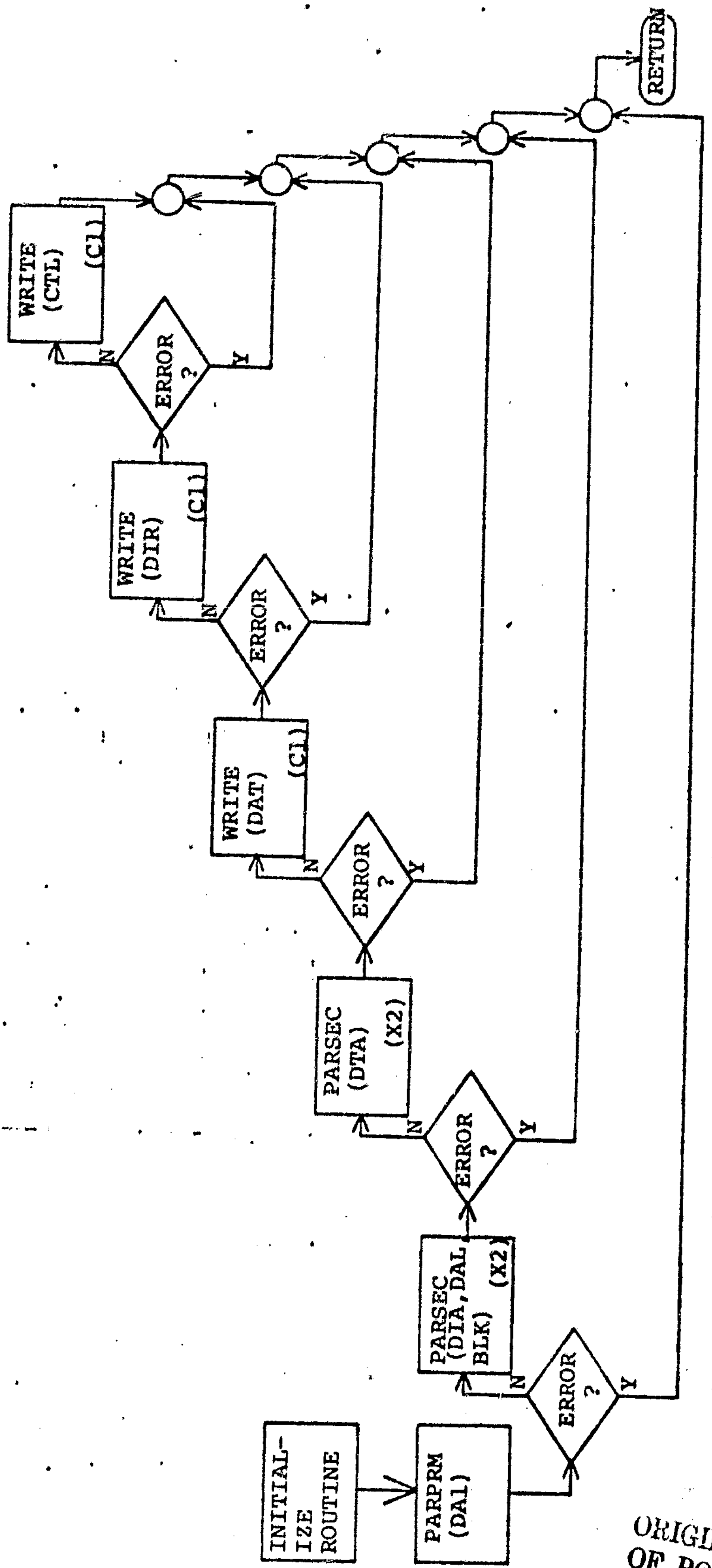


Figure 3.11.- Subroutine DEFDAT(R1.5).

3.2.12 SUBROUTINE RPLCTL (R2.1)

RPLCTL (replace control) takes the input action request and parses it for data to update the proper control record.

3.2.12.1 Linkages

RPLCTL calls subroutines PARPRM, PARSEC, and WRITE. It is called only by RPL.

3.2.12.2 Interfaces

None

3.2.12.3 Inputs

The input to RPLCTL is a partially parsed action request.

3.2.12.4 Outputs

The output of RPLCTL is an updated control record or an error number.

3.2.12.5 Description

RPLCTL accepts the action request from RPL and interprets it for the data necessary to update the proper control record. It then rewrites the updated record to the data base. Any attempt to update the control record with incomplete or erroneous data will result in an error condition being returned to RPL.

3.2.12.6 Flowchart

Figure 3.12.

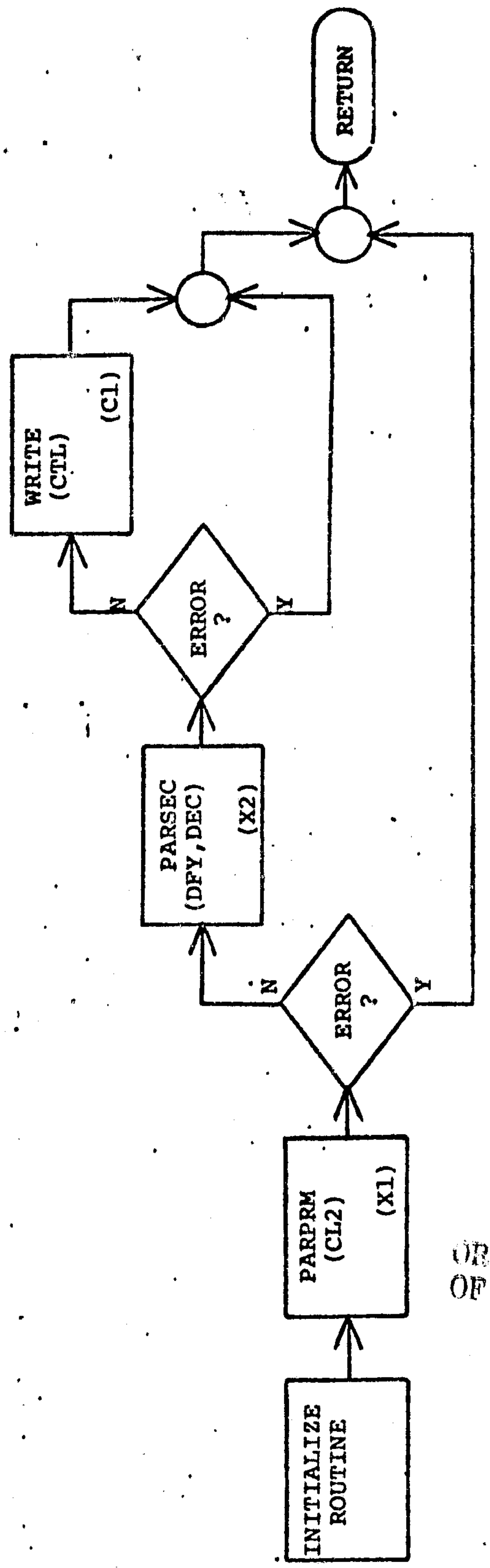


Figure 3.12.- Subroutine RPLCTL (R2.1).

3.2.13 SUBROUTINE RPLDIR (R2.2)

RPLDIR (replace directory) takes the input action request and parses it for data to update the proper directory entry.

3.2.13.1 Linkages

RPLDIR calls subroutines PARPRM, PARSEC, and WRITE. It is called only by RPL.

3.2.13.2 Interfaces

None.

3.2.13.3 Inputs

The input to RPLDIR is a partially parsed action request.

3.2.13.4 Outputs

The output of RPLDIR is an updated directory entry or an error number.

3.2.13.5 Description

RPLDIR accepts the action request from RPL and interprets it for the data necessary to update the proper directory entry. It then rewrites the updated directory entry to the data base. Any attempt to update a directory entry with incomplete or erroneous data will result in an error condition being returned to RPL.

3.2.13.6 Flowchart

Figure 3.13.

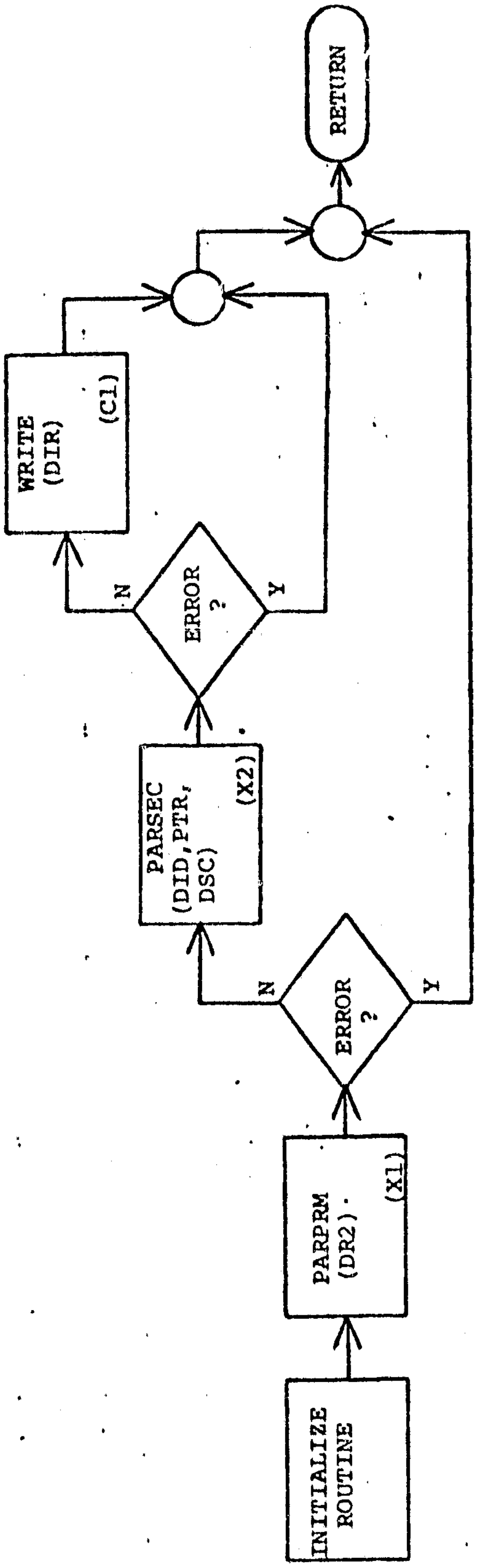


Figure 3.13.- Subroutine RPLDIR (R2.2).

3.2.14 SUBROUTINE RPLDES (R2.3)

RPLDES (replace descriptor) takes the input action request and parses it for data to update the proper descriptor entry.

3.2.14.1 Linkages

RPLDES calls subroutines PARPRM, PARSEC, and WRITE. It is called only by RPL.

3.2.14.2 Interfaces

None.

3.2.14.3 Inputs

The input to RPLDES is a partially parsed action request.

3.2.14.4 Outputs

The output of RPLDES is an updated descriptor entry or an error number.

3.2.14.5 Description

RPLDES accepts the action request from RPL and interprets it for the data necessary to update the proper descriptor entry. It then rewrites the updated descriptor entry to the data base. Any attempt to update a descriptor entry with incomplete or erroneous data will result in an error condition being returned to RPL.

3.2.14.6 Flowchart

Figure 3.14.

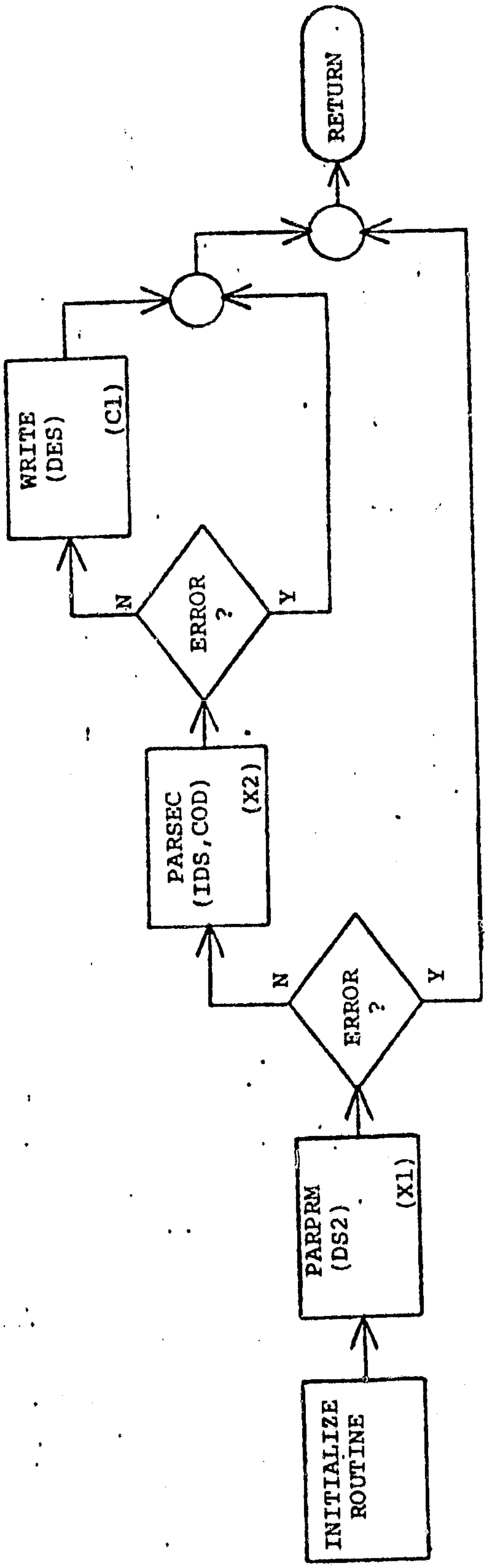


Figure 3.14.- Subroutine RPLDES (R2.3).

3.2.15 SUBROUTINE RPLDFN (R2.4)

RPLDFN (replace definition) takes the input action request and parses it for data to update the proper model definition record.

3.2.15.1 Linkages

RPLDFN calls subroutines PARPRM, PARSEC, and WRITE. It is called only by RPL.

3.2.15.2 Interfaces

None.

3.2.15.3 Inputs

The input to RPLDFN is a partially parsed action request.

3.2.15.4 Outputs

The output of RPLDFN is an updated model definition record or an error number.

3.2.15.5 Description

RPLDFN accepts the action request from RPL and interprets it for the data necessary to update the proper model definition record. It then rewrites the updated record to the data base. Any attempt to update a model definition with incomplete or erroneous data will result in an error condition being returned to RPL.

3.2.15.6 Flowchart

Figure 3.15.

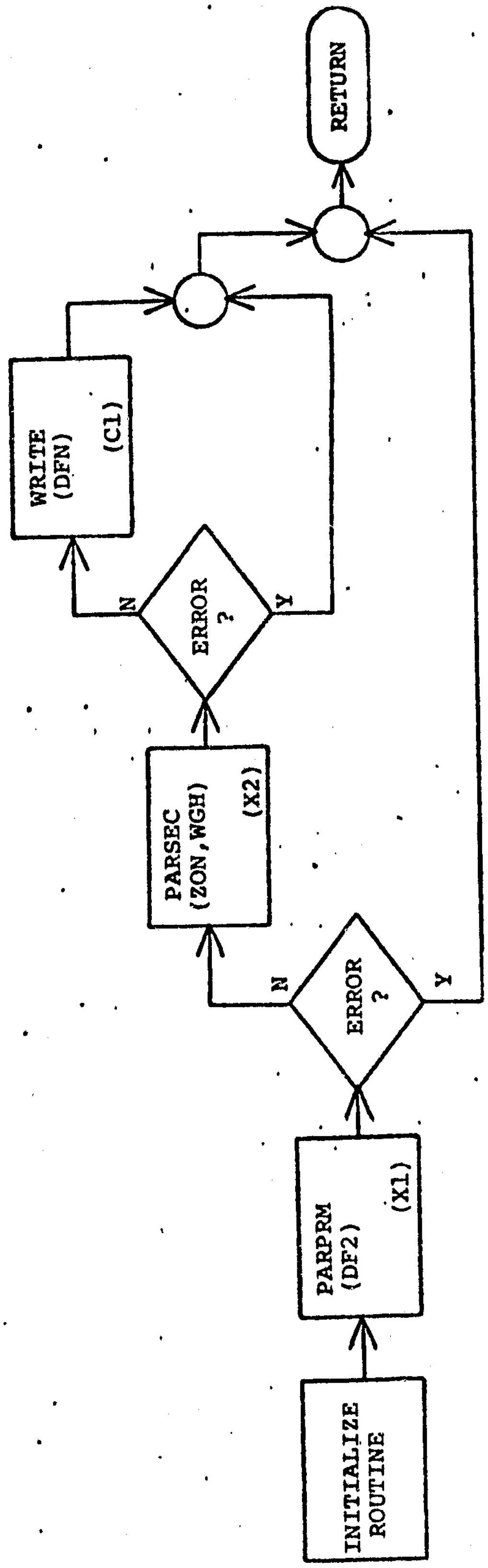


Figure 3.15.- Subroutine RPLDFN (R2.4).

3.2.16 SUBROUTINE RPLDAT (R2.5)

RPLDAT (replace data) takes the input action request and parses it for data to update the proper data record(s).

3.2.16.1 Linkages

RPLDAT calls subroutines PARPRM, PARSEC, and WRITE. It is called only by RPL.

3.2.16.2 Interfaces

None.

3.2.16.3 Inputs

The input to RPLDAT is a partially parsed action request.

3.2.16.4 Outputs

The output of RPLDAT is an updated data record(s) or an error number.

3.2.16.5 Description

RPLDAT accepts the action request from RPL and interprets it for the data necessary to update the proper data record(s). It then rewrites the updated record(s) to the data base. Any attempt to update a data record with incomplete or erroneous data will result in an error condition being returned to RPL.

3.2.16.6 Flowchart

Figure 3.16.

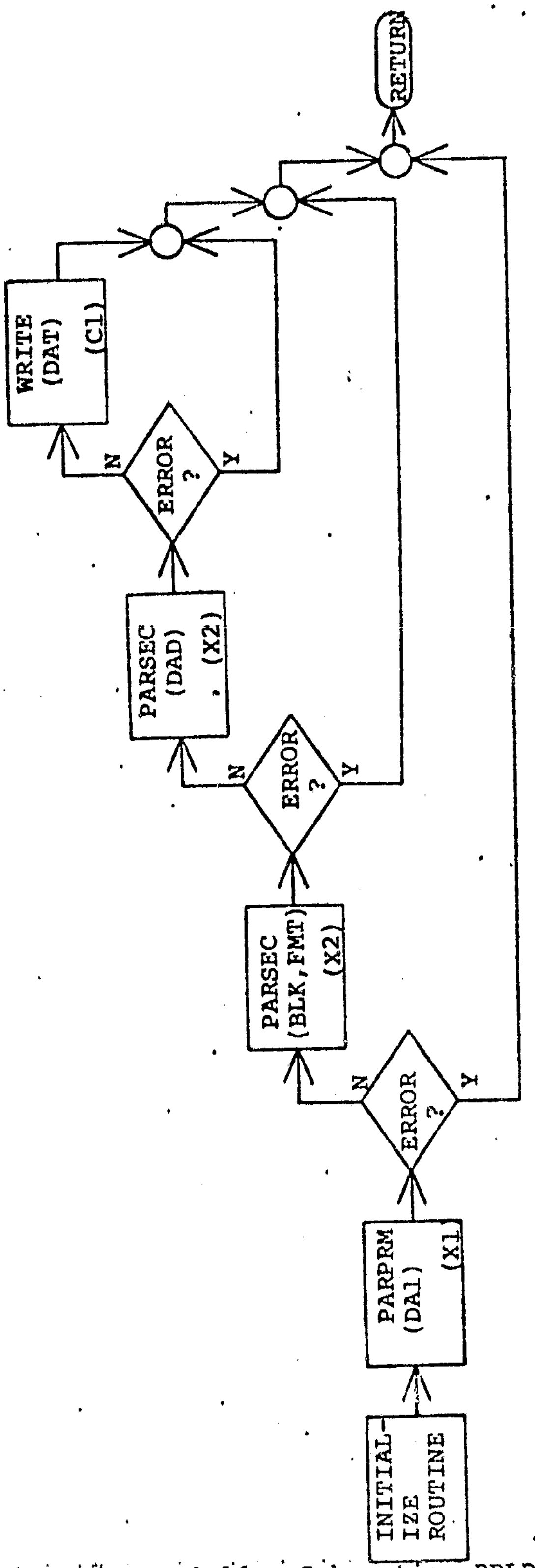


Figure 3.16.- Subroutine RPLDAT (R2.5).

3.2.17 SUBROUTINE ADDCTL (R3.1)

ADDCTL (add control) takes the input action request and parses it for data to add to the proper control record.

3.2.17.1 Linkages

ADDCTL calls the subroutines PARPRM, PARSEC, and WRITE. It is called only by ADD.

3.2.17.2 Interfaces

None.

3.2.17.3 Inputs

The input to ADDCTL is a partially parsed action request.

3.2.17.4 Outputs

The output of ADDCTL is an updated control record or an error number.

3.2.17.5 Description

ADDCTL accepts the action request from ADD and interprets it for the password to be added to the proper control record. It then rewrites the updated record to the data base. Any attempt to add to the control record with incomplete or erroneous data will result in an error condition being returned to ADD.

3.2.17.6 Flowchart

Figure 3.17.

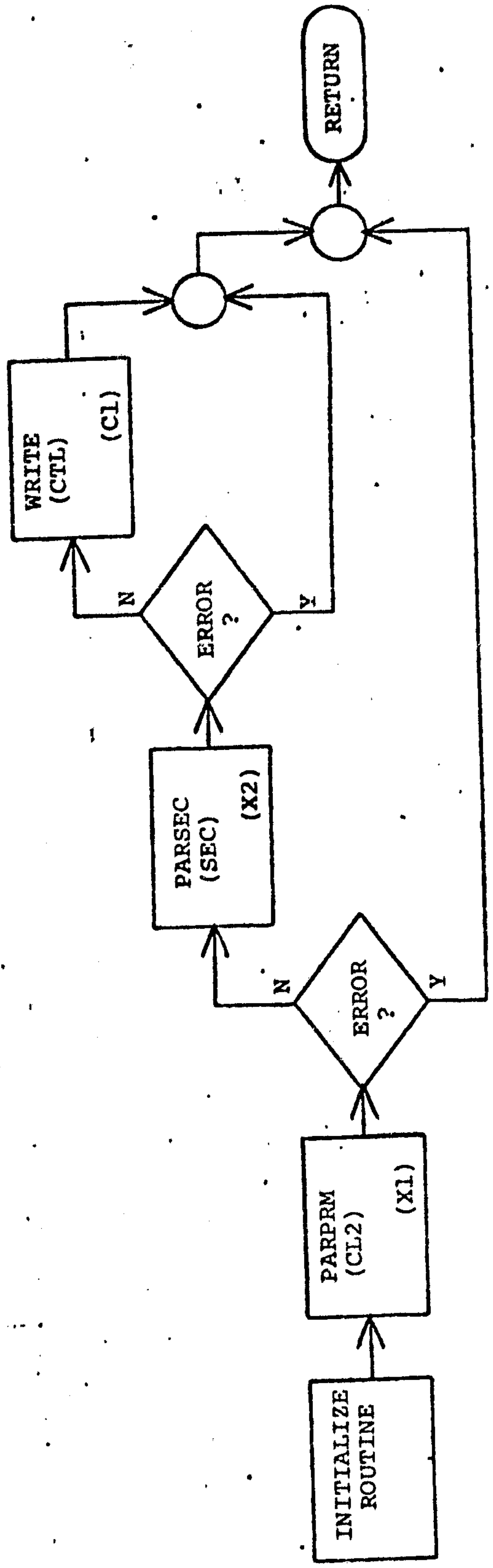


Figure 3.17.- Subroutine ADDCTL (R3.1).

3.2.18 SUBROUTINE ADDDIR (R3.2)

ADDDIR (add directory) takes the input action request and parses it for data to be added to the proper directory entry.

3.2.18.1 Linkages

ADDDIR updates the proper control record to insure that the added directory element is properly referenced.

3.2.18.3 Inputs

The input to ADDDIR is a partially parsed action request.

3.2.18.4 Outputs

The output of ADDDIR is an updated directory entry and an updated control record or an error number.

3.2.18.5 Description

ADDDIR accepts the action request from ADD and interprets it for the data to be added to the correct directory entry. It then updates the proper control record and rewrites both to the data base. Any attempt to add to a directory entry with incomplete or erroneous data will result in an error condition being returned to ADD.

3.2.18.6 Flowchart

Figure 3.18.

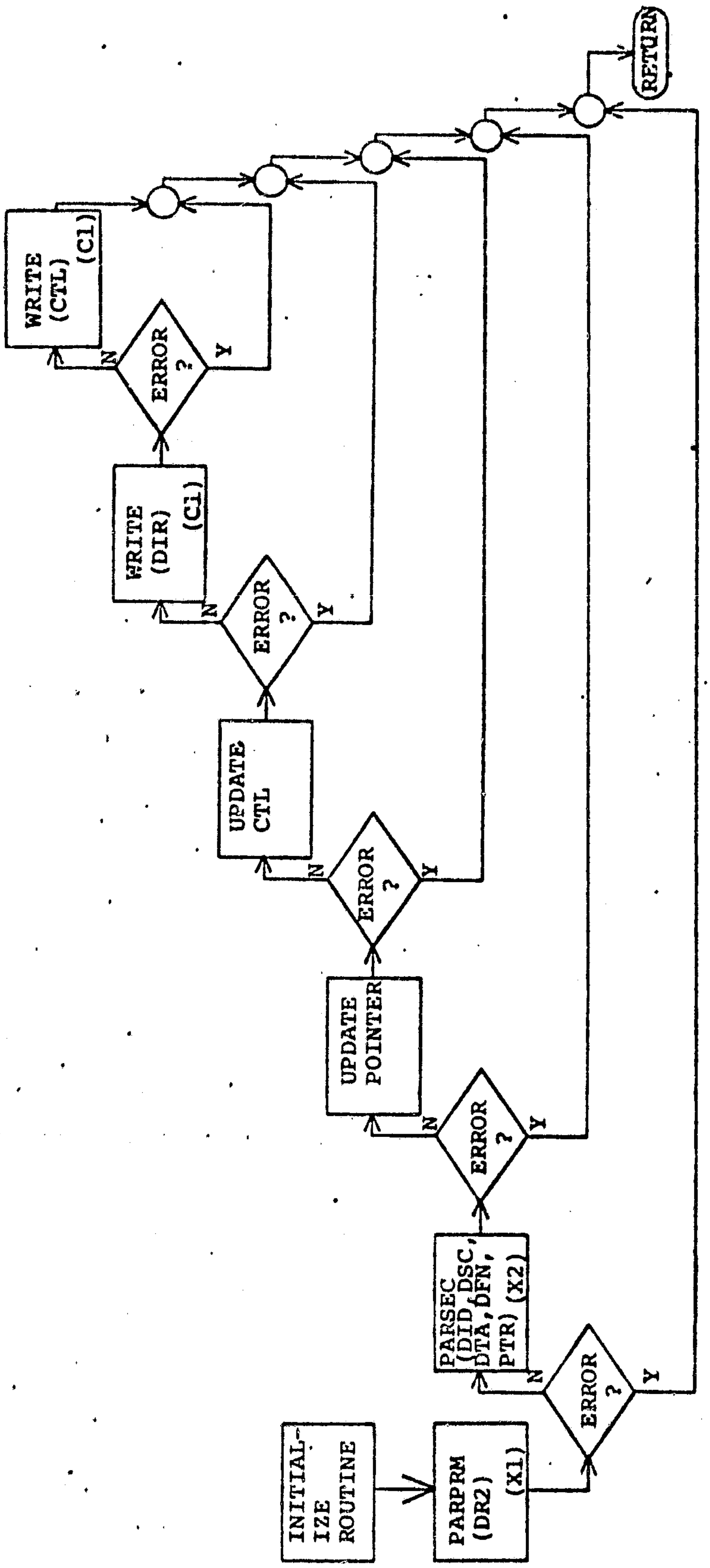


Figure 3.18.- Subroutine ADDDIR (R3.2).

3.2.19 SUBROUTINE ADDDFN (R3.4)

ADDDFN (add definition) takes the input action request and parses it for new data to be added to the correct definition record.

3.2.19.1 Linkages

ADDDFN calls the subroutines PARPRM, PARSEC, and WRITE. It is called only by ADD.

3.2.19.2 Interfaces

None.

3.2.19.3 Inputs

The input to ADDDFN is a partially parsed action request.

3.2.19.4 Outputs

The output of ADDDFN is an updated definition record or an error number.

3.2.19.5 Description

ADDDFN accepts the action request from ADD and interprets it for the data to be added to the correct definition record. It then rewrites the definition record to the data base. Any attempt to add to a definition with incomplete or erroneous data will result in an error condition being returned to ADD.

3.2.19.6 Flowchart

Figure 3.19.

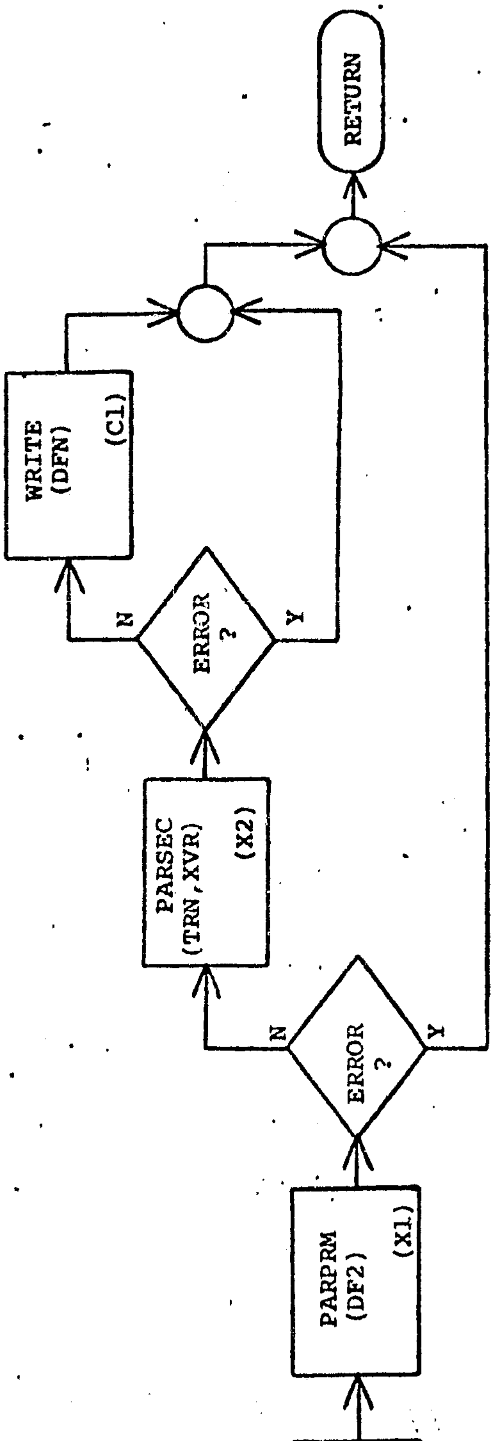


Figure 3.19.- Subroutine ADDDFN (R3.4).

3.2.20 SUBROUTINE ADDDAT (R3.5)

ADDDAT (add data) takes the input action request and parses it for new data to be added to the correct data record(s).

3.2.20.1 Linkages

ADDDAT calls the subroutines PARPRM, PARSEC, and WRITE. It is called only by ADD.

3.2.20.2 Interfaces

None.

3.2.20.3 Inputs

The input to ADDDAT is a partially parsed action request.

3.2.20.4 Outputs

The output of ADDDAT is an updated data record(s) or an error number.

3.2.20.5 Description

ADDDAT accepts the action request from ADD and interprets it for the data to be added to the correct data record(s) It then rewrites the data record(s) to the data base. Any attempt to add incomplete or erroneous data will result in an error condition being returned to ADD.

3.2.20.6 Flowchart

Figure 3.20.

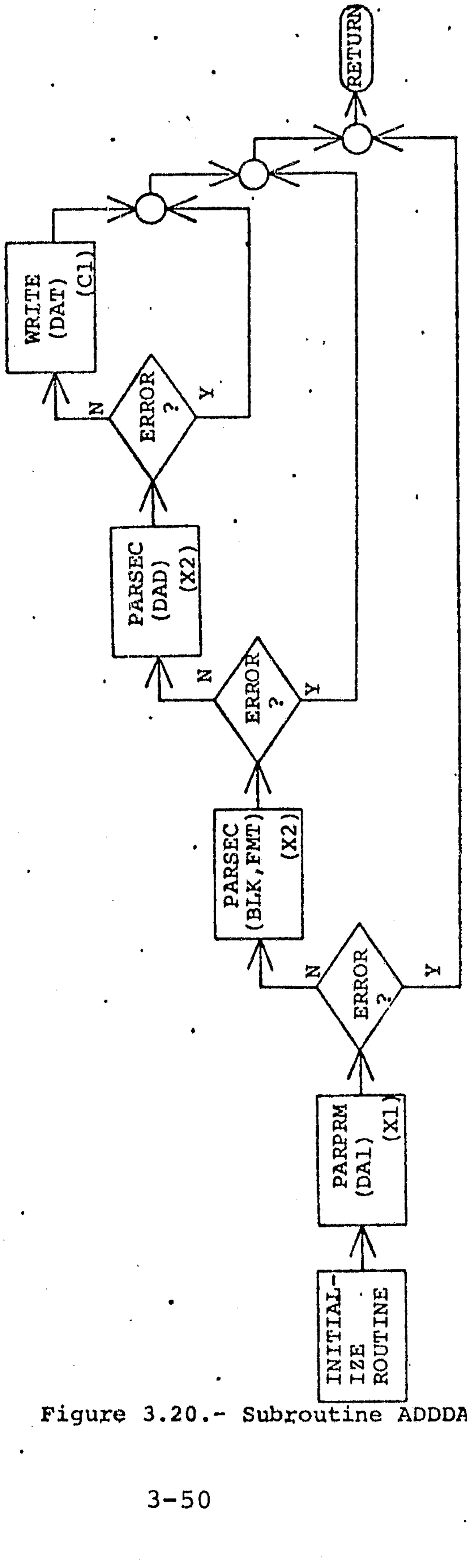


Figure 3.20.- Subroutine ADDDAT (R3.5)

3.2.21 SUBROUTINE DELCTL (R4.1)

DELCTL (delete control) takes the input action request and parses it to determine what is to be deleted from the control record.

3.2.21.1 Linkages

DELCTL calls the subroutines PARPRM, PARSEC, and WRITE. It is called only by DEL.

3.2.21.2 Interfaces

None.

3.2.21.3 Inputs

The input to DELCTL is a partially parsed action request.

3.2.21.4 Outputs

The output of DELCTL is an updated control record or an error number.

3.2.21.5 Description

DELCTL accepts the action request from DEL and interprets it for the password to be deleted from the proper control record. It then rewrites the updated record to the data base. Any attempt to delete a password with incomplete or erroneous data will result in an error condition being returned to DEL.

3.2.21.6 Flowchart

Figure 3.21.

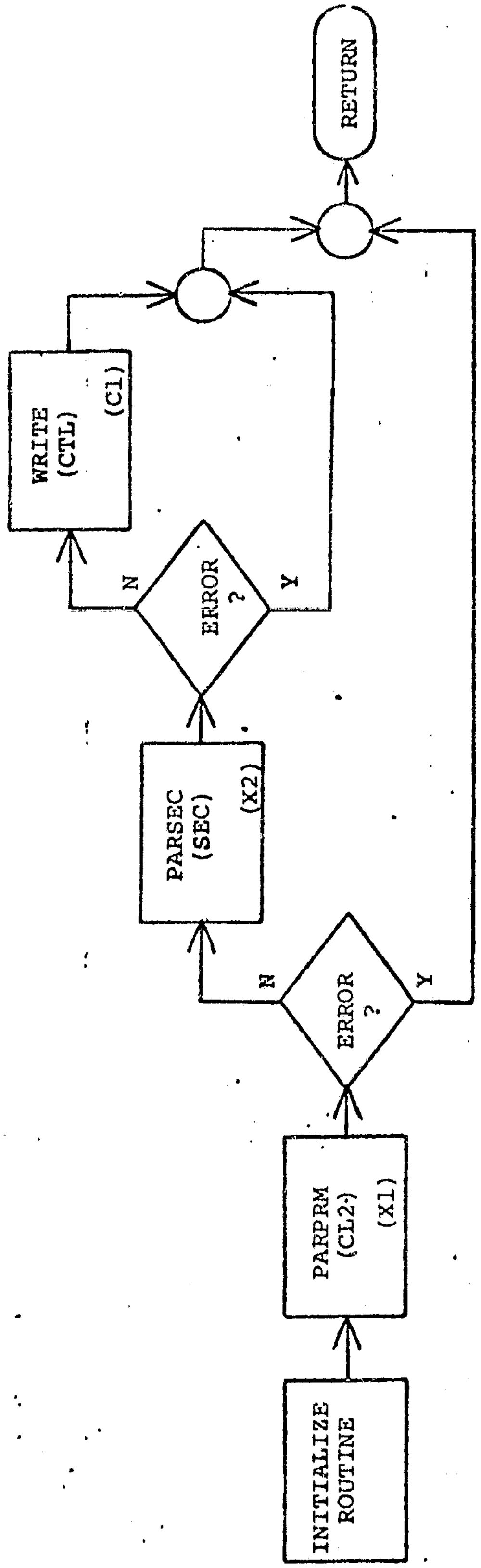


Figure 3/21.- Subroutine DELCTL (R4.1)

3.2.22 SUBROUTINE DELDIR (R4.2)

DELDIR (delete directory) takes the input action request and parses it to determine what is to be deleted from the correct directory entry.

3.2.22.1 Linkages

DELDIR calls the subroutines PARPRM, PARSEC, and WRITE. It is called only by DEL.

3.2.22.2 Interfaces

DELDIR updates the proper control record to insure that the deleted directory element references are deleted.

3.2.22.3 Inputs

The input to DELDIR is a partially parsed action request.

3.2.22.4 Outputs

The output of DELDIR is an updated directory entry and an updated control record or an error number.

3.2.22.5 Description

DELDIR accepts the action request from DEL and interprets it to determine which directory elements of the correct directory entry are to be deleted. It then updates the proper control record and rewrites both to the data base. Any attempt to delete a directory element with incomplete or erroneous data will result in an error condition being returned to DEL.

3.2.22.6 Flowchart

Figure 3.22.

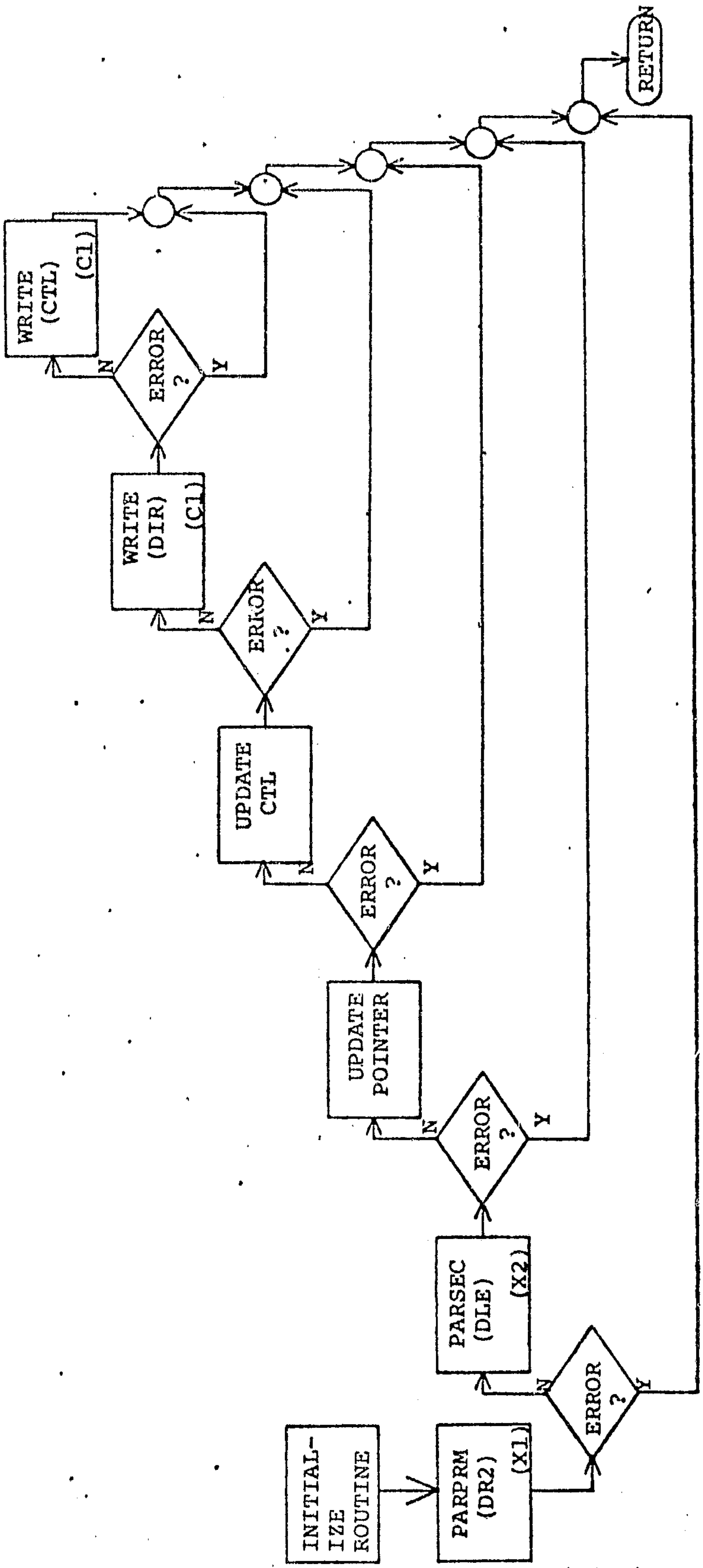


Figure 3.22.- Subroutine DELDIR (R4.2).

3.2.23 SUBROUTINE DELDFN (R4.4)

DELDFN (delete definition) takes the input action request and parses it to determine what part of the correct definition is to be deleted.

3.2.23.1 Linkages

DELDFN calls the subroutines PARPRM, PARSEC, and WRITE. It is called only by DEL.

3.2.23.2 Interfaces

None.

3.2.23.3 Inputs

The input to DELDFN is a partially parsed action request.

3.2.23.4 Output

The output of DELDFN is an updated definition record or an error number.

3.2.23.5 Description

DELDFN accepts the action request from DEL and interprets it to determine what part of the model definition record is to be deleted. It then rewrites the updated record to the data base. Any attempt to delete any portion of a model definition record with incomplete or erroneous data will result in an error condition being returned to DEL.

3.2.23.6 Flowchart

Figure 3.23.

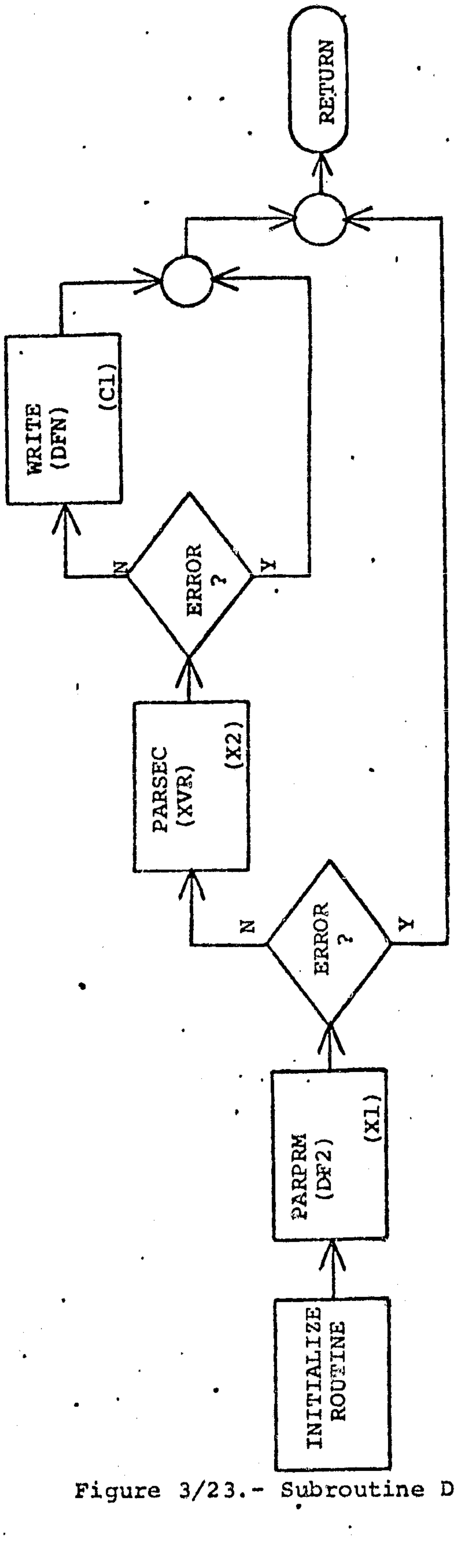


Figure 3/23.- Subroutine DELDFN (R4.4).

3.2.24 SUBROUTINE LSTCTL (R5.1)

LSTCTL (list control) takes the input action request and parses it to determine how much of the proper control record is to be listed and prints that portion.

3.2.24.1 Linkages

LSTCTL calls subroutines PARPRM and PARSEC.

3.2.24.2 Interfaces

None.

3.2.24.3 Inputs

The input to LSTCTL is a partially parsed action request.

3.2.24.4 Outputs

The output of LSTCTL is a formatted listing of the requested portions of the selected control record or an error number.

3.2.24.5 Description

LSTCTL accepts the action request from LST and interprets it to determine what control record is to be listed and what parts are to be selected for printing. It then reformats the selected portions for listing on the printer. Any attempt to list a control record with incomplete or erroneous data will result in an error condition being returned to LST.

3.2.24.6 Flowchart

Figure 3.24.

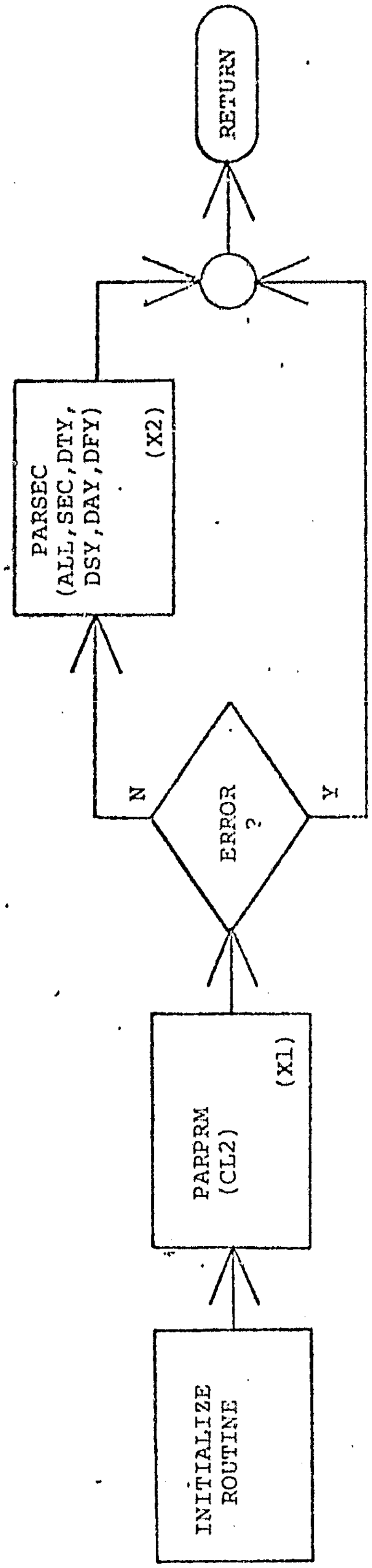


Figure 3.24.- Subroutine LSTCTL (R5.1).

3.2.25 SUBROUTINE LSTDIR (R5.2)

LSTDIR (list directory) takes the input action request and parses it to determine how much of the selected directory entry is to be listed and prints that portion.

3.2.25.1 Linkages

LSTDIR calls subroutines PARPRM and PARSEC.

3.2.25.2 Interfaces

None.

3.2.25.3 Inputs

The input to LSTDIR is a partially parsed action request.

3.2.25.4 Outputs

The output of LSTDIR is a formatted listing of the requested portions of the selected directory entry or an error number.

3.2.25.5 Description

LSTDIR accepts the action request from LST and interprets it to determine what directory entry is to be listed and what parts are to be selected for printing. It then reformats the selected portions for listing on the printer. Any attempt to list a directory entry with incomplete or erroneous data will result in an error condition being returned to LST.

3.2.25.6 Flowchart

Figure 3.25.

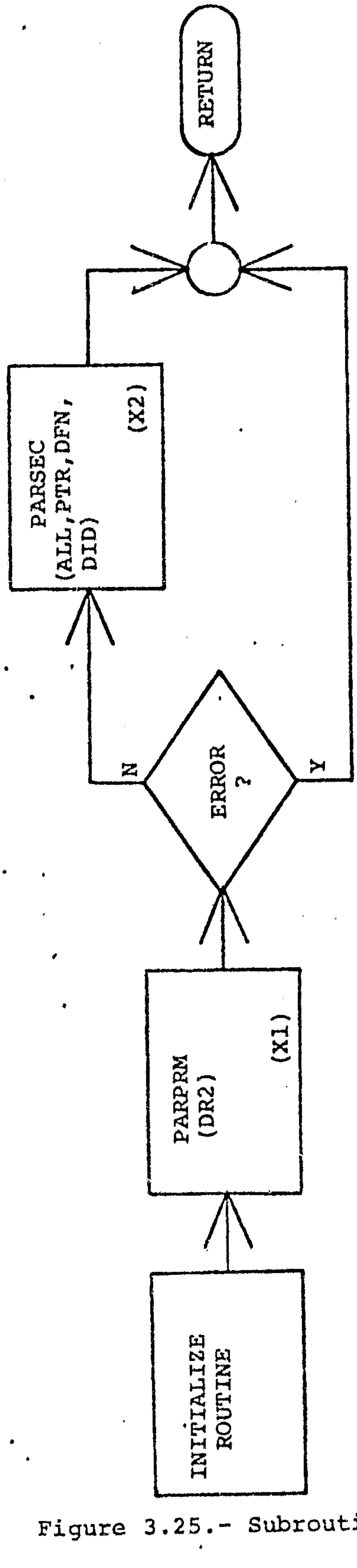


Figure 3.25.- Subroutine LSTDIR (R5.2).

3.2.26 SUBROUTINE LSTDES (R5.3)

LSTDES (list descriptor) takes the input action request and parses it to determine how much of the selected descriptor entry is to be listed and prints that portion.

3.2.26.1 Linkages

LSTDES calls the subroutines PARPRM and PARSEC.

3.2.26.2 Interfaces

None.

3.2.26.3 Inputs

The input to LSTDES is a partially parsed action request.

3.2.26.4 Outputs

The output of LSTDES is a formatted listing of the requested portions of the selected descriptor entry or an error number.

3.2.26.5 Description

LSTDES accepts the action request from LST and interprets it to determine what descriptor entry is to be listed and what parts are to be selected for printing. It then reformats the selected portions for listing on the printer. Any attempt to list a descriptor entry with incomplete or erroneous data will result in an error condition being returned to LST.

3.2.26.6 Flowchart

Figure 3.26.

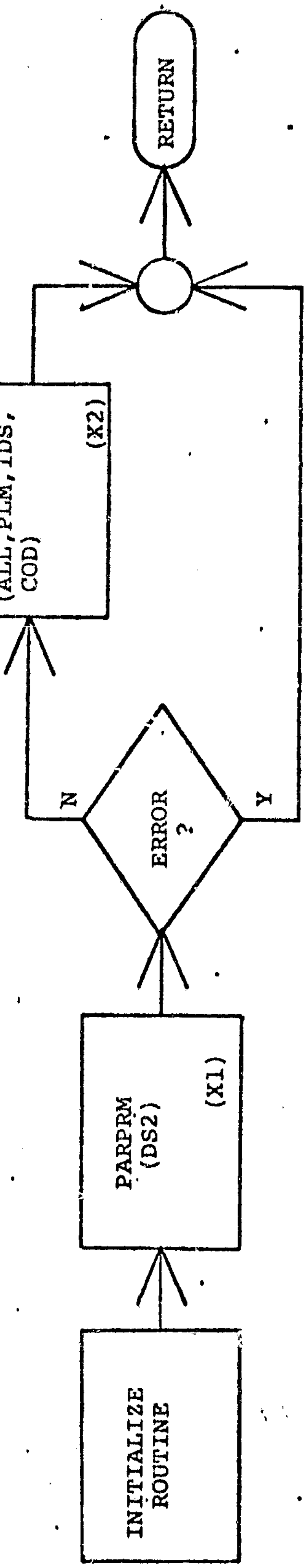


Figure 3.26.- Subroutine LSTDES (R5.3).

3.2.27 SUBROUTINE LSTDFN (R5.4)

LSTDFN (list definition) takes the input action request and parses it to determine how much of the selected definition record is to be listed and prints that portion.

3.2.27.1 Linkages

LSTDFN calls the subroutines PARPRM and PARSEC.

3.2.27.2 Interfaces

None.

3.2.27.3 Inputs

The input to LSTDFN is a partially parsed action request.

3.2.27.4 Outputs

The output of LSTDFN is a formatted listing of the requested portions of the selected model definition or an error number.

3.2.27.5 Description

LSTDFN accepts the action request from LST and interprets it to determine what model definition is to be listed and what parts are to be selected for printing. It then reformats the selected portions for listing on the printer. Any attempt to list a model definition record with incomplete or erroneous data will result in an error condition being returned to LST.

3.2.27.6 Flowchart

Figure 3.27.

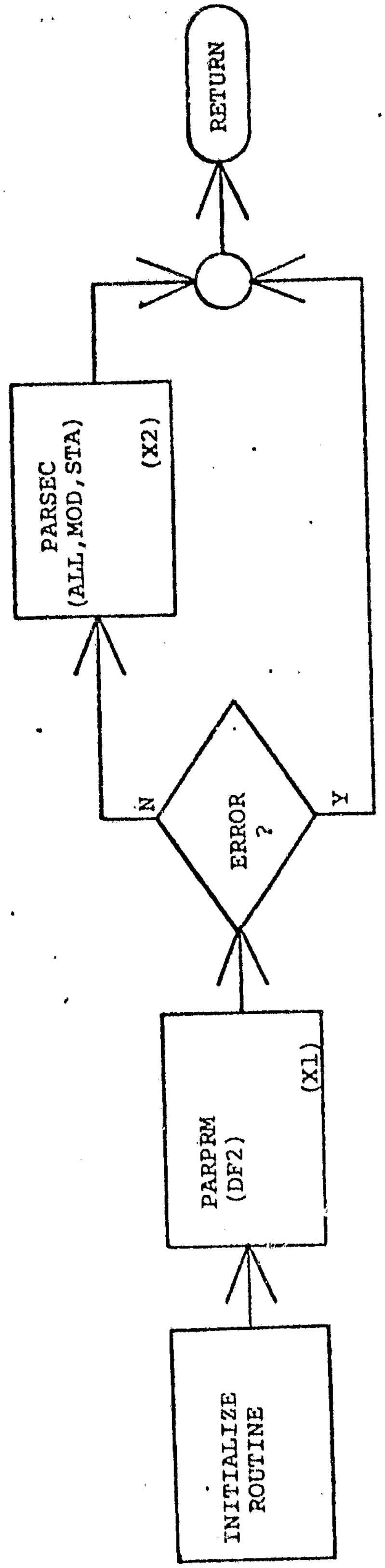


Figure 3.27.- Subroutine LSTDFN (R5.4).

3.2.28 SUBROUTINE LSTDAT (R5.5)

LSTDAT (list data) takes the input action request and parses it to determine how much of the selected data record(s) is to be listed and prints that portion.

3.2.28.1 Linkages

LSTDAT calls the subroutines PARPRM and PARSEC.

3.2.28.2 Interfaces

None.

3.2.28.3 Inputs

The input to LSTDAT is a partially parsed action request.

3.2.28.4 Outputs

The output of LSTDAT is a formatted listing of the requested portions of the selected data record(s) or an error number.

3.2.28.5 Description

LSTDAT accepts the action request from LST and interprets it to determine what data record(s) is to be listed and what parts are to be selected for printing. It then reformats the selected portions for listing on the printer. Any attempt to list data using an incomplete or erroneous request will result in an error condition being returned to LST.

3.2.28.6 Flowchart

Figure 3.28.

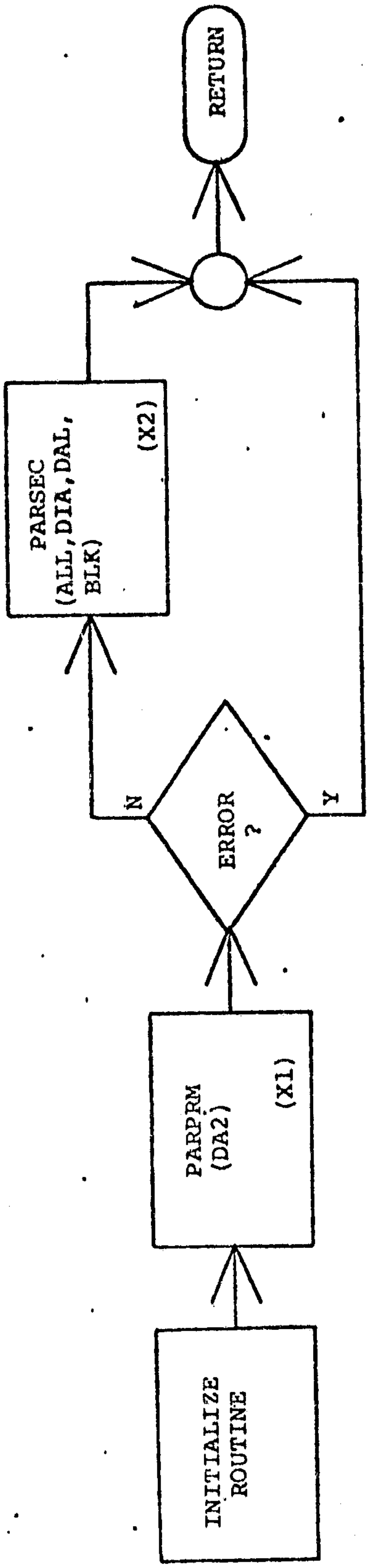


Figure 3.28.- Subroutine LSTDAT (R5.5).

3.2.29 SUBROUTINE PARPRM (X1)

PARPRM (parse parameter) takes the input action request and parses it to determine the sections (see 3.2.32) to be subsequently processed.

3.2.29.1 Linkages

To be determined.

3.2.29.2 Interfaces

To be determined.

3.2.29.3 Inputs

The input to PARPRM is a partially parsed action request.

3.2.29.4 Outputs

The output of PARPRM is a partially parsed action request, interpreted section name(s), and the applicable data base record(s)/entry(ies).

3.2.29.5 Description

PARPRM accepts the action request from its calling routine, parses the section name from the request, and determines whether the section name is legitimate. It then opens the applicable file, checks security, and reads the record(s)/entry(ies) of interest into core. If an error is detected in the input request, an error condition is set and control is returned to the calling routine.

3.2.29.6 Flowchart

Figure 3.29.

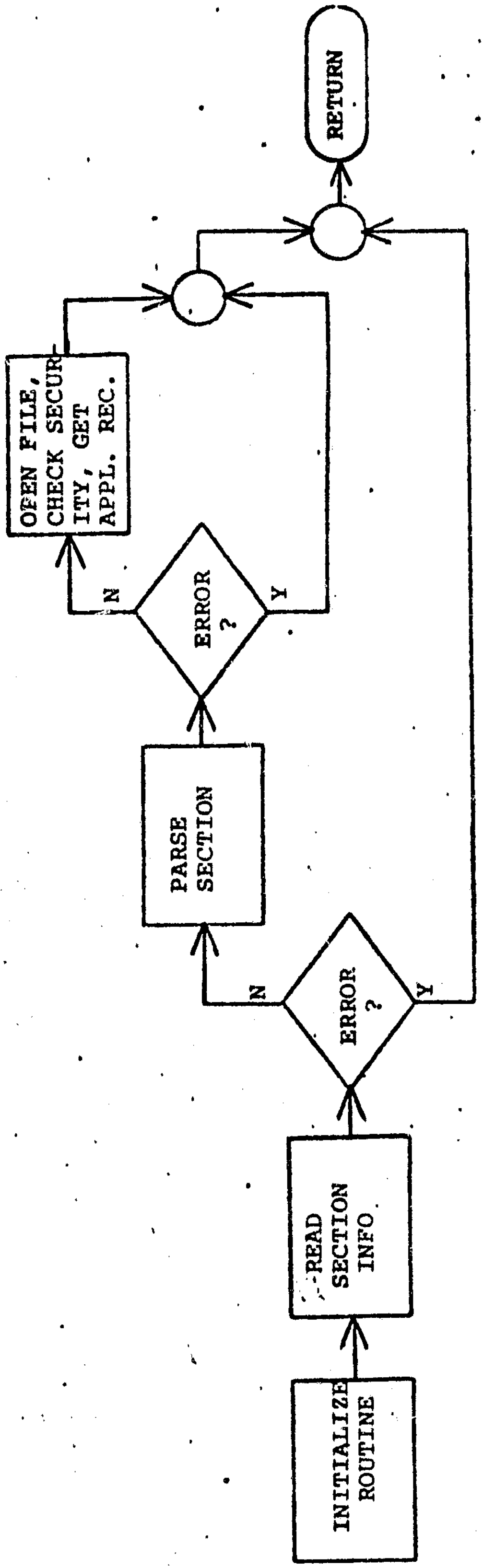


Figure 3.29.- Subroutine PARPRM (X1).

3.2.30 SUBROUTINE PARSEC (X2)

PARSEC (parse sections) takes the input action request and parses it for the subparameters (see Appendix A) and data necessary to accomplish the task(s) called for by the section name(s).

3.2.30.1 Linkages

To be determined.

3.2.30.2 Interfaces

To be determined.

3.2.30.3 Inputs

The inputs to PARSEC are a partially parsed action request, interpreted section name(s), and the applicable data base record(s)/entry(ies).

3.2.30.4 Outputs

The output of PARSEC is an updated record(s)/entry(ies) which is ready for writing to the data base or (in the case of LST) a completely formatted listing to the printer or (in the case of an error) an error number.

3.2.30.5 Description

PARSEC accepts the action request from its calling routine, parses the rest of the request for section data, determines whether the section data (subparameters) are legitimate, and applies the parsed data to the record(s)/entry(ies) in core, in accordance with the tasking implied by the section name. It then either moves the updated record(s)/entry(ies) to the output area for rewriting to the data base or (in the case of LST) reformats the record(s)/entry(ies) for listing and moves it to SYSOUT for printing. If an error is detected during processing an error condition is set and control is returned to the calling routine.

3.2.30.6 Flowchart

Figure 3.30.

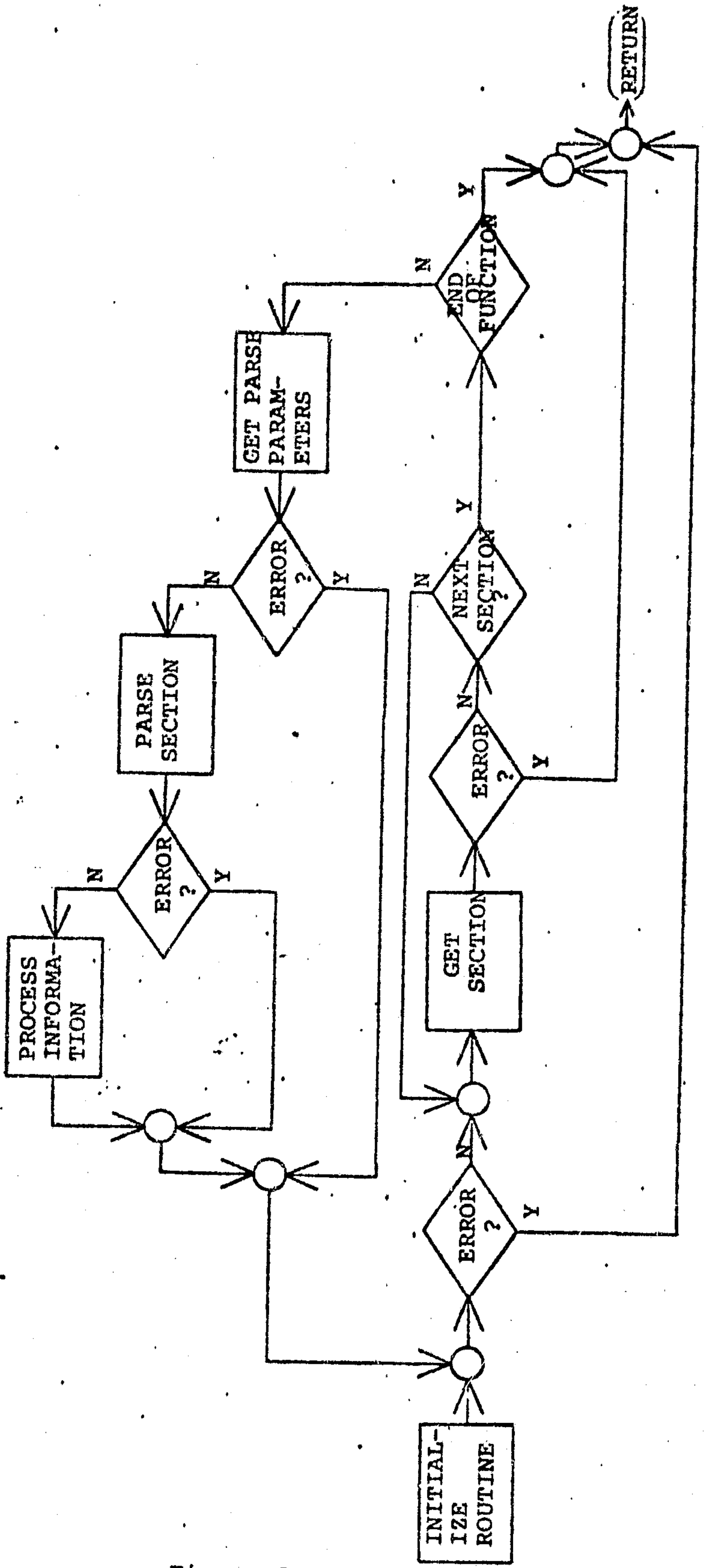


Figure 3.30.- Subroutine PARSEC (X2).

3.2.31 SUBROUTINE WRITE (C1)

WRITE takes the updated record from the output area and writes it to the data base.

3.2.31.1 Linkages

To be determined.

3.2.31.2 Interfaces

To be determined.

3.2.31.3 Inputs

The input to WRITE is an updated/newly defined record in the output area.

3.2.31.4 Output

The output from WRITE is an updated/newly defined record which is written to the data base.

3.2.31.5 Description

Using its calling argument as the decision parameter, WRITE writes the record from the output area to the proper area in the data base. In the case of an error, WRITE will set an error condition and return control to its calling routine.

3.2.31.6 Flowchart

Figure 3.31.

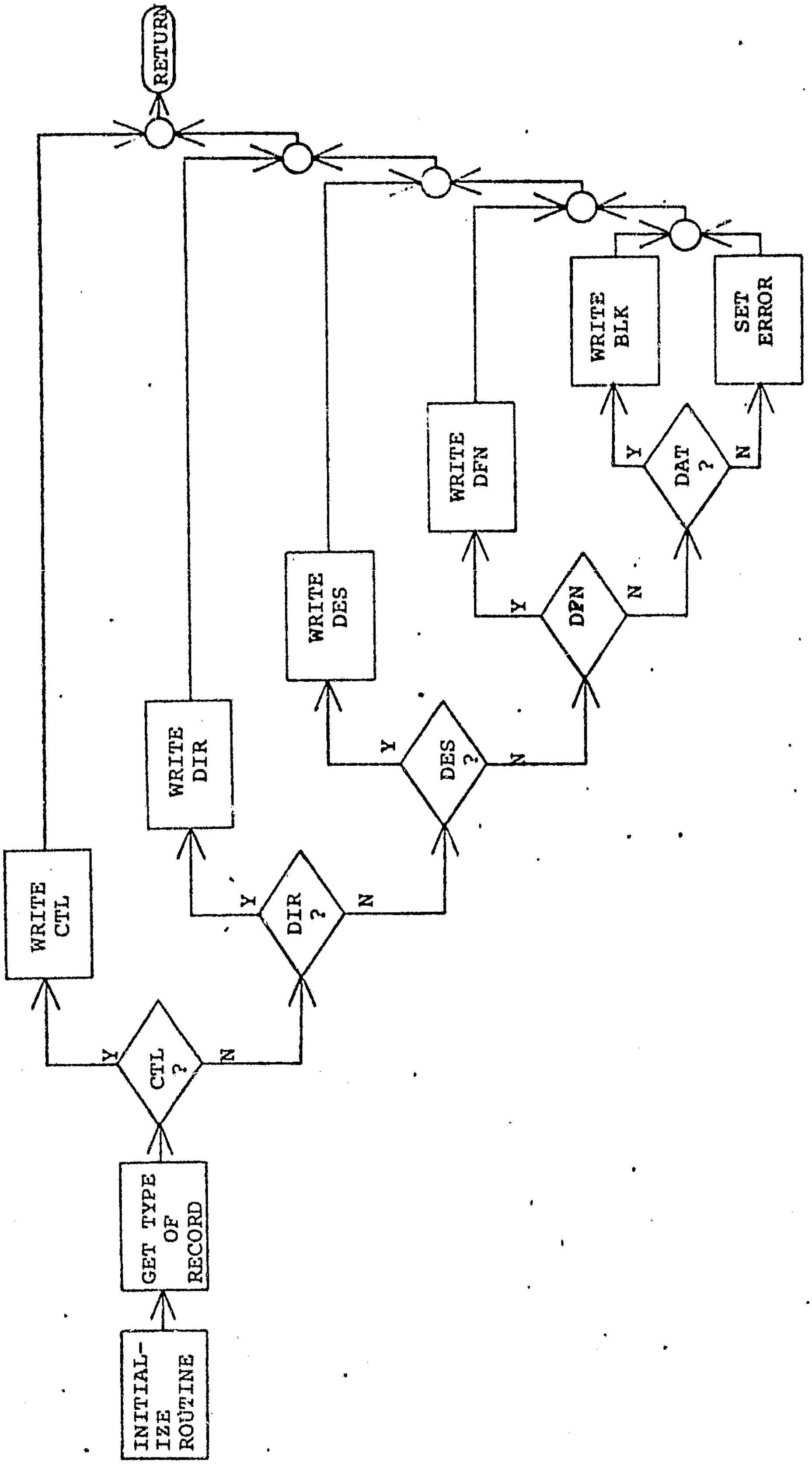


Figure 3.31.- Subroutine WRITE (C1).

3.2.32 SUBFUNCTION STRUCTURES AND FORMATS

3.2.32.1 Control Record Structure

```
DCL 1 CTL EXTERNAL,
      2 SEC,
        3 FILEID             CHAR(8),
        3 NUMPASS            FIXED BIN(15,0),
        3 PASSWORD(8)        CHAR(8),
        3 RESERVED           CHAR(28),
        3 NUMDIR             FIXED BIN(15,0),
        3 NUMREC             FIXED BIN(15,0),
        3 NUMDES             FIXED BIN(15,0),
        3 NUMDEF             FIXED BIN(15,0),
        3 NUMDAT             FIXED BIN(15,0),
      2 DTY,
        3 DIRIDNUM(8)        FIXED BIN(15,0),
        3 DIRIDCHR(8)        CHAR(4),
        3 DIRNOREC(8)        FIXED BIN(15,0),
        3 DIRTOTDE(8)        FIXED BIN(15,0),
        3 DIRLOCAT(8,4)      FIXED BIN(15,0),
        3 DIRNUMDE(8,4)      FIXED BIN(15,0),
      2 DSY,
        3 DESID(16)          FIXED BIN(15,0),
        3 DESLOCAT(16)       FIXED BIN(15,0),
        3 DESDISPL(16)       FIXED BIN(15,0),
      2 DFY,
        3 DEFID(200)         FIXED BIN(15,0),
        3 DEFLOCAT(200)      FIXED BIN(15,0),
        3 DEFDISPL(200)      FIXED BIN(15,0),
      2 DAY,
        3 DATID(200)         FIXED BIN(31,0),
        3 DATDESID(200)      FIXED BIN(15,0),
        3 DATBLKSZ(200)      FIXED BIN(15,0),
        3 DATBKALC(200)      FIXED BIN(15,0),
        3 DATBKUSE(200)      FIXED BIN(15,0),
        3 DATNOREC(200)      FIXED BIN(15,0),
        3 DATPOINT(200)      FIXED BIN(15,0),
        3 DATLOCAT(456)      FIXED BIN(15,0),
        3 DATDISPL(456)      FIXED BIN(15,0),
        3 DATRBALC(456)      FIXED BIN(15,0),
        3 DATRBUSE(456)      FIXED BIN(15,0),
        3 DATFBKID(456)      FIXED BIN(15,0),
        3 DATNXARR(456)      FIXED BIN(15,0),
      2 REC,
        3 RECTYPE(456)       FIXED BIN(15,0),
        3 RECSPACE(456)      FIXED BIN(15,0),
        3 RECLOCAT(456)      FIXED BIN(15,0);
```

3.2.32.2 Subfunction Control Format

functions	applicable sections
DEF	CL1, SEC, DTY, DSY, DFY, DAY, REC
RPL	CL2, SEC, DSY, DFY, DAY,
ADD	CL2, SEC
DEL	CL2, SEC, DSY, DFY
LST	CL2, SEC, DTY, DSY, DFY, DAY, REC, ALL

section	parm/subparm	abv.	description
CL1	DDNAME=(c)	DD=(	JCL DD name
	NUMSECS=(n)	NC=(	number of sections used in this run (max=3)
	SECTIONS=(c)	SC=(	section names
CL2	DDNAME=(c)	DD=(	JCL DD name
	FILEID=(c)	ID=(	file identification name
	PASSWORD=(c)	PW=(	password that can access the file
	NUMSECS =(n)	NC=(	number of sections used in this run (max=7)
	SECTIONS=(c)	SC=(	section names
SEC	FILEID=(c)	ID=(	file identification name (1-8 characters)
	NUMPASS=(n)	NP=(	# passwords that can access the file (max=8)
	PASSWORD=(c)	PW=(	passwords that can access the file
	RESERVED=(c)	RV=(	extra space or filler
	NUMDIR=(n)	ND=(	# of countries on the file (max= 8)
	NUMREC =(n)	FR=(	# of records on file, excluding control rec.
	NUMDES=(n)	NS=(	# of descriptors found on the file
	NUMDEF=(n)	NF=(	# of model definitions found on the file
	NUMDAT=(n)	NT=(	# of directory elements with data on the file
	SUBRANGE=(n)	SR=(	range of values in subscript array to be considered
REC	RECTYPE =(n)	RT=(	record type
	RECSPACE =(n)	RS=(	freespace on record(bytes)
	RELOCAT=(n)	RC=(	record #
	SUBRANGE =(n)	SR=(	

Subfunction Control Format, continued

DTY	DIRIDNUM=(n)	DI=(	numeric country codes
	DIRIDCHR=(c)	DC=(	alphabetic country codes
	DIRNOREC=(n)	DR=(	# of directory records per country
	DIRTOTDE=(n)	DT=(	total # of directory elements per country
	DIRLOCAT=(n)	DL=(	record number of each directory record
	DIRNUMDE=(n)	DE=(	# of directory elements per directory record
	SUBRANGE=(n)	SR=(	range of values in subscript array to be considered
DSY	DESID=(n)	SI=(	identification #'s of descriptors on file
	DESLOCAT=(n)	SL=(	record number of each descriptor
	DESDISPL=(n)	SP=(	byte # of beginning of each descriptor
	SUBRANGE=(n)	SR=(	
DFY	DEFID=(n)	FI=(	identification #'s of model definition
	DEFLOCAT=(n)	FL=(	record number of each model definition
	DEFDISPL=(n)	FP=(	byte # of beginning of each definition
	SUBRANGE=(n)	SR=(	
DAY	DATID=(n)	TI=(	directory ID # of each area with data on the file
	DADESID=(n)	TD=(	ID # of descriptor for data in this block
	DATBLKS=(n)	BS=(	#bytes in data block
	DATBKALC=(n)	TB=(	total # of blocks allocated
	DATBKUSE=(n)	TU=(	total # blocks used
	DATNOREC=(n)	TN=(	# records used for this directory element's data
	DATPOINT=(n)	TP=(	subscript of data location array
	DATLOCAT=(n)	TL=(	record # for data
	DATDISPL=(n)	TH=(	byte # of beginning of data
	DATRBALC=(n)	TO=(	#blocks allocated for this data on record
	DATRBUSE=(n)	TZ=(	# blocks used for this data
	DATFBKID=(n)	TF=(	first block ID
	DATNXARR =(n)	TX=(	pointer to next subscript of data location array
	SUBRANGE =(n)	SR=(	
ALL	SUBRANGE=(n)	SR=(	

3.2.32.3 Directory Structure

```
DCL 1 DIR(268),
        2 DID,
          3 NUMLEVEL              FIXED BIN(15,0),
          3 DIRID                 FIXED BIN(15,0),
          3 DIRNAME               CHAR(16),
          3 NUMBDES               FIXED BIN(15,0),
          3 NUMBDEF               FIXED BIN(15,0),
        2 PTR,
          3 PARENT                FIXED BIN(15,0),
          3 BROTHER               FIXED BIN(15,0),
          3 CHILD                 FIXED BIN(15,0),
        2 DSC,
          3 DESID(3)              FIXED BIN(15,0),
          3 DESLOCAT(3)           FIXED BIN(15,0),
          3 DESDISPL(3)           FIXED BIN(15,0),
        2 DTA,
          3 DATLOCAT(3)           FIXED BIN(15,0),
          3 DATDISPL(3)           FIXED BIN(15,0),

        2 DFN,
          3 DEFID(6)              FIXED BIN(15,0),
          3 DEFLOCAT(6)           FIXED BIN(15,0),
          3 DEFDISPL(6)           FIXED BIN(15,0);
```

3.2.32.4 Directory Subfunction Format

functions	applicable sections
DEF	DR1, DID, DSC, DTA, DFN, PTR, DTY, REC
RPL	DR2, DID, PTR, DSC, DTA, DFN
ADD	DR2, DID, DSC, DTA, DFN, PTR, DTY, REC
DEL	DR2, DLE, PTR, DTY, REC,
LST	DR2, DID, PTR, DSC, DTA, DFN, ALL

section	parm/subparm	abv.	description
DR1	DDNAME=(c)	DD=(	JCL DD name
	FILEID=(c)	ID=(	file identification name
	PASSWORD =(c)	PW=(	password that can access the file
	NUMSECS=(n)	NC=(	number of sections used in this run (max=7)
	SECTIONS =(c)	SC=(	section names
DR2	DDNAME =(c)	DD=(	JCL DD name
	FILEID = (c)	ID=(	file identification name
	PASSWORD = (c)	PS=(	password that can access the file
	COUNTRY = (n)	CO=(	number of country in question
	NUMSECS = (n)	NC=(	number of sections used in this run (max=7)
	SECTIONS = (c)	SC=(	
DID	DIRID=(n)	RI=(	id # of lowest level of directory in question
	DIRNAME = (c)	DN=(	name of directory level in question
	NUMBDES =(n)	NS=(	descriptor number used by this directory elem
	NUMBDEF =(n)	NF=(	model defn # used by this directory element
	NUMLEVEL=(n)	NL=(	number of levels
	LEVELS =(n)	LV=(	level numbers
	SUBRANGE =(n)	SR=(	range of values in subscript
PTR	LEVELS =(n)	LV=(	
	PARENT= (n)	PA=(	parent of directory element in question
	BROTHER=(n)	BR=(	brother of directory elem in question
	CHILD= (n)	CH=(	child of directory elem in question
	SUBRANGE =(n)	SR=(	
	NUMLEVEL =(n)	NL=(	

Directory Subfunction Format, continued

DSC	DESID=(n)	SI=(	# of descriptor for this directory element
	DESLOCAT=(n)	SL=(	record # of descriptor for this directory elem
	DESDISPL=(n)	SP=(	byte # of beginning of descriptor
	NUMLEVEL=(n)	NL=(	
	LEVELS=(n)	LV=(	
	SUBRANGE=(n)	SR=(	
DTA	DATLOCAT=(n)	TL=(	record number of data for this directory
	DATDISPL=(n)	TH=(	byte # of beginning of data
	NUMLEVEL=(n)	NL=(	
	LEVELS=(n)	LV=(	
	SUBRANGE=(n)	SR=(	
DFN	DEFID=(n)	FI=(	id # of model defn for this directory elem
	DEFLOCAT=(n)	FL=(	record number of model def for this directory
	DEFDISPL=(n)	FP=(	byte # of beginning of model definition
	NUMLEVEL=(n)	NL=(	
	LEVELS=(n)	LV=(	
	SUBRANGE=(n)	SR=(	
DLE	NUMLEVEL=(n)	NL=(	
	LEVELS=(n)	LV=(	
REC	RECTYPE=(n)	RT=(	record type
	RECSPACE=(n)	RS=(	freespace on record (bytes)
	RELOCAT=(n)	RC=(	record #
DTY	DIRIDNUM=(n)	DI=(	numeric country codes
	DIRIDCHR=(c)	DC=(	alphabetic country codes
	DIRNOREC=(n)	DR=(	# directory records per country
	DIRTOTDE=(n)	DT=(	total # directory elements per country
	DIRLOCAT=(n)	DL=(	record # of each directory record
	DIRNUMDE=(n)	DE=(	# directory elements per directory record
	SUBRANGE=(n)	SR=(	
ALL	SUBRANGE=(n)	SR=(	
	NUMLEVEL=(n)	NL=(	
	LEVELS=(n)	LV=(	

3.2.32.5 Descriptor Structure

```
DCL 1 DES EXTERNAL,
      2 PLM,
        3 DESID                 FIXED BIN(15,0),
        3 BLKSIZE               FIXED BIN(15,0),
        3 NUMBLKS               FIXED BIN(15,0),
        3 NUMVARBL              FIXED BIN(15,0),
        3 TOTCODES              FIXED BIN(15,0),
        3 TOTUNITS              FIXED BIN(15,0),
        3 RESERVED              CHAR(16),
      2 IDS,
        3 CODEID(56)            FIXED BIN(15,0),
        3 CODENAME(56)          CHAR(16),
        3 CODEABV(56)           CHAR(4),
        3 UNITID(20)            FIXED BIN(15,0),
        3 UNITNAME(20)          CHAR(16),
        3 UNITABV(20)           CHAR(4),
      2 COD,
        3 CODLOCAT(20)          FIXED BIN(15,0),
        3 UNTLOCAT(20)          FIXED BIN(15,0),
        3 BASE(20)              FIXED BIN(15,0),
        3 SCALE(20)             FIXED BIN(15,0),
        3 OUTFIELD(20)          FIXED BIN(15,0),
        3 BLKLOCAT(20)          FIXED BIN(15,0),
        3 NUMELEM(20)           FIXED BIN(15,0),
        3 ELMLOCAT(20,31)       FIXED BIN(15,0);
```

3.2.32.6 Subfunction Descriptor Format

functions	applicable sections
DEF	DS1,PLM,IDS,COD,DSC,DSY,REC
RPL	DS2,PLM,IDS,COD
LST	DS2,PLM,IDS,COD,ALL

section	parm/subparm	abv.	description
DS1	DDNAME=(c)	DD=(	JCL DD name
	FILEID=(c)	ID=(	file identification name
	PASSWORD=(c)	PW=(	password that can access the file
	NUMSECS=(n)	NC=(	number of sections used in this run (max=6)
	SECTIONS=(c)	SC=(	section names
	COUNTRY=(n)	CO=(	country ID
DS2	DDNAME=(c)	DD=(	JCL DD name
	FILEID=(c)	ID=(	file identification name
	PASSWORD=(c)	PW=(	password that can access the file
	DESID=(n)	SI=(	identification # of this descriptor
	NUMSECS=(n)	NC=(	number of sections used in this run (max=6)
	SECTIONS=(c)	SC=(	section names
PLM	DESID=(n)	SI=(	identification # of this descriptor
	BLKSIZE=(n)	BS=(	# bytes in data block
	NUMBLKS=(n)	NB=(	# blocks in this descriptor's data blocks
	NUMVARBL=(n)	NV=(	# variables in this descriptor
	TOTUNITS=(n)	TS=(	total # of units in this descriptor
	TOTCODES=(n)	TC=(	total # of codes
	RESERVED=(c)	RV=(	free space or filler
IDS	CODEID=(n)	CI=(	code # for variable
	CODENAME=(c)	CN=(	variable name
	CODEABV=(c)	CA=(	variable abbreviation
	UNITID=(n)	UI=(	variable unit code #
	UNITNAME=(c)	UN=(	variable unit name
	UNITABV=(c)	UA=(	variable unit abbreviation
	SUBRANGE=(n)	SR=(	range of values in subscript array to be considered

Subfunction Descriptor Format, continued

COD	CODLOCAT=(n)	CL=(	position in code array
	UNTLOCAT=(n)	UL=(	position in unit array
	BASE =(n)	BA=(	type of variable
	SCALE=(n)	SE=(	location of decimal point, if any
	OUTFIELD=(n)	OF=(	output print field size
	BLKLOCAT=(n)	BL=(	where variable is located in data block
	NUMELEM=(n)	NE=(	# subelements (#months, #crops, etc.)
	ELMLOCAT=(n)	EL=(	positions in subelement array to be considered
	SUBRANGE=(n)	SR=(	range of values in subscript array to be considered
DSC	NUMLEVEL=(n)	NL=(	
	LEVELS=(n)	LV=(	
	DESID=(n)	SI=(	ID # of descriptor
	DESLOCAT=(n)	SL=(	record # of descriptor
	DESDISPL=(n)	SP=(	byte # of beginning of descriptor
DSY	DESID=(n)	SI=(	
	DESLOCAT=(n)	SL=(	
	DESDISPL=(n)	SP=(	
REC	RECTYPE=(n)	RT=(	record type
	RECSPACE =(n)	RS=(	freespace on record (bytes)
	RELOCAT=(n)	RC=(	record #
ALL	SUBRANGE=(n)	SR=(	

3.2.32.7 Model Definition Structure

```
DCL 1 DEF EXTERNAL,
     2 MOD,
       3 DEFID                 FIXED BIN(15,0),
       3 NOZONE                FIXED BIN(15,0),
       3 NOSTRATA              FIXED BIN(15,0),
       3 NOZONSTA              FIXED BIN(15,0),
       3 NWGHCODE              FIXED BIN(15,0),
       3 NOHYEARS              FIXED BIN(15,0),
       3 NORYEARS              FIXED BIN(15,0),
       3 NOXYEARS              FIXED BIN(15,0),
       3 NOTRUNCS              FIXED BIN(15,0),
       3 NOYVARBL              FIXED BIN(15,0),
       3 NOXVARBL              FIXED BIN(15,0),
       3 NOYLDDAT              FIXED BIN(15,0),
       3 NOMETDAT              FIXED BIN(15,0),
       3 NOZINDEX              FIXED BIN(15,0),
       3 NOINVARB              FIXED BIN(15,0),
       3 CROPNUMB              FIXED BIN(15,0),
       3 MODLNAME              CHAR(24),
       3 RESERVED              CHAR(244),
     2 INV,
       3 NUMVARBL              FIXED BIN(15,0)
       3 RESERVED              CHAR(2),
       3 VARBCODE (8)          FIXED BIN(15,0);
```

Model Definition Structure, continued

```
      2 ZON,
        3 LATITUDE((26)      FLOAT BIN(21),
        3 NUMLEVEL  (26)     FIXED BIN(15,0),
        3 NUMVARBL(26)       FIXED BIN(15,0),
        3 LEVELS(26,8)       FIXED BIN(15,0),
        3 VARBCODE(26,8)     FIXED BIN(15,0),
      2 WGH,
        3 WGHVARBL(8)        FIXED BIN(15,0),
        3 NOWEIGHT(8)        FIXED BIN(15,0),
        3 WEIGHTS(8,20)      FLOAT BIN(21),
      2 HYR,
        3 NUMLEVEL(6)        FIXED BIN(15,0),
        3 NUMYRPRS(8)        FIXED BIN(15,0),
        3 LEVELS(6,8)        FIXED BIN(15,0),
        3 BEGYEAR(6,8)       FIXED BIN(15,0),
        3 ENDYEAR(6,8)       FIXED BIN(15,0),
      2 RYR,
        3 NUMLEVEL(6)        FIXED BIN(15,0),
        3 NUMYRPRS(8)        FIXED BIN(15,0),
        3 LEVELS(6,8)        FIXED BIN(15,0),
        3 BEGYEAR(6,8)       FIXED BIN(15,0),
        3 ENDYEAR(6,8)       FIXED BIN(15,0),
      2 XYR,
        3 NUMLEVEL(6)        FIXED BIN(15,0),
        3 NUMYRPRS(8)        FIXED BIN(15,0)
        3 LEVELS(6,8)        FIXED BIN(15,0),
        3 BEGYEAR(6,8)       FIXED BIN(15,0),
        3 ENDYEAR(6,8)       FIXED BIN(15,0),
      2 ZDX
        3 NUMLEVEL  (6)      FIXED BIN(15,0),
        3 LEVELS  (6,8)      FIXED BIN(15,0),
        3 UPPERAWC  (6)      FLOAT BIN(21),
        3 LOWERAWC  (6)      FLOAT BIN(21),
        3 UPPERPCP(6)        FLOAT BIN(21),
        3 LOWERPCP  (6)      FLOAT BIN(21),
```

Model Definition Structure, continued

```
     2 TRN,
       3 TRUNCID(16)        FIXED BIN(15,0),
       3 TRUNCABV(16)       CHAR(4),
       3 TRUNNAME(16)       CHAR(16),
       3 NUMVARBL(16)       FIXED BIN(15,0),
       3 VARBLNUM(16,24)    FIXED BIN(15,0),
     2 YVR,
       3 VARBLID            FIXED BIN(15,0),
       3 VAROPER            FIXED BIN(15,0),
       3 VARVCODE(3)        FIXED BIN(15,0),
       3 VARSCODE(3)        FIXED BIN(15,0),
       3 VARIKON(3)         FIXED BIN(15,0),
       3 VARLOCAT(3)        FIXED BIN(15,0),
       3 VARFKON(3)         FLOAT BIN(21),
     2 XVR,
       3 VARBLID(24)        FIXED BIN(15,0),
       3 VAROPER(24)        FIXED BIN(15,0),
       3 VARDEV(24)         FIXED BIN(15,0),
       3 VARVCODE(24,3)     FIXED BIN(15,0),
       3 VARSCODE(24,3)     FIXED BIN(15,0),
       3 VARIKON(24,3)      FIXED BIN(15,0),
       3 VARLOCAT(24,3)     FIXED BIN(15,0),
       3 VARFKON(24,3)      FLOAT BIN(21),
     2 YDA,
       3 NUMLEVEL(6)        FIXED BIN(15,0),
       3 WHEREGET(6)        FIXED BIN(15,0),
       3 USEDYET(6)         FIXED BIN(15,0),
       3 NUMVARBL(6)        FIXED BIN(15,0),
       3 LEVELS(6,8)        FIXED BIN(15,0),
       3 VARBCODE(6,8)      FIXED BIN(15,0),
       3 NUMSCODE(6,8)      FIXED BIN(15,0),
       3 FIELDSIZ(6,8)      FIXED BIN(15,0),
       3 DIGIT(6,8)         FIXED BIN(15,0),
     2 MDA,
       3 NUMLEVEL(6)        FIXED BIN(15,0),
       3 WHEREGET(6)        FIXED BIN(15,0),
       3 USEDYET(6)         FIXED BIN(15,0),
       3 NUMVARBL(6)        FIXED BIN(15,0),
       3 LEVELS(6,8)        FIXED BIN(15,0),
       3 VARBCODE(6,8)      FIXED BIN(15,0),
       3 NUMSCODE(6,8)      FIXED BIN(15,0),
       3 FIELDSIZ(6,8)      FIXED BIN(15,0),
       3 DIGIT(6,8)         FIXED BIN(15,0);
```

3.2.32.8 Subfunction MOdel Definition Format

function	applicable section
DEF	DF1,MOD,INV,ZON,WGH,HYR,RYR,XYR,ZDX,TRN,YVR,XVR, YDA,MDA,DFY,DFN,REC
RPL	DF2,MOD,INV,ZON,WGH,HYR,RYR,XYR,ZDX,TRN,YVR,XVR, YDA,MDA
ADD	DF2,ZON,INV,WGH,HYR,RYR,XYR,ZDX,TRN,YVR,XVR,YDA, MDA
DEL	DF2,INV,ZON,WGH,HYR,RYR,XYR,ZDX,TRN,YVR,XVR, YDA,MDA
LST	DF2,MOD,INV,ZON,WGH,HYR,RYR,XYR,ZDX,TRN,YVK,XVR, YDA,MDA,ALL

section	parm/subparm	abv.	description
DF1	DDNAME=(c)	DD=)	JCL DD name
	FILEID=(c)	ID=(	file identification name
	PASSWORD=(c)	PW=(	password that can access the file
	NUMSECS=(n)	NC=(	number of sections used in this run (max=17)
	SECTIONS=(c)	SC=(	section names
	COUNTRY=(n)	CO=(	country ID number
DF2	FILEID=(c)	ID=(	
	PASSWORD=(c)	PW=(	
	NUMSECS=(n)	NC=(	
	SECTIONS=(c)	SC=(	
	DEFID=(n)	FI=(	model definition ID number
	DDNAME=(c)	DD=(	
MOD	NOZONE=(n)	NZ=(	# zones
	NOSTRATA =(n)	NR=(	#strata
	NOZONSTA=(n)	ZS=(	total # zones plus strata
	NWGHCODE=(n)	NW=(	# weight codes
	NOHYEARS=(n)	HY=(	# historical years
	NORYEARS=(n)	RY=(	# run years
	NOXYEARS=(n)	XY=(	# normal years
	NOTRUNCS =(n)	NU=(	# truncations
	NOYVARBL=(n)	NY=(	# Y-variables
	NOXVARBL=(n)	NX=(	# X-variables
	NOYLDDAT=(n)	DY=(	# yield data cards
	NOZINDEX=(n)	ZX=(	# 2-index cards
	NOINVARB=(n)	IB=(	# raw variable cards

Subfunction Model Definition Format, continued

MOD (cont)	NOMETDAT=(n)	DM=(	# met data cards
	CROPNUMB=(n)	CR=(	crop ID number
	MODLNAME=(c)	MN=(	model name
	RESERVED=(c)	RV=(	extra space or filler
	DEFID=(n)	FI=(	model definition number
INV	NVMVARBL=(n)	NV=(	# of variables
	RESERVED =(c)	RV=C	reserved space or filler
	VARBCODE=(n)	VC=(	variable code number
	SUBRANGE=(n)	SR=(	subscript range
ZON/STA	SUBRANGE=(n)	SR=(	subscript range
	NUMLEVEL=(n)	NL=(	number of levels
	LEVELS=(n)	LV=(	level numbers
	LATITUDE=(f)	LA=(	latitude of zone/strata
	NUMVARBL=(n)	NV=(	number of variables
	VARBCODE=(n)	VC=(	variable code numbers
WGH	WGHVARBL=(n)	WV=(	code # of variable to be weighted
	NOWEIGHT=(n)	WN=(	# weights
	WEIGHTS=(n)	WH=(	weights
	SUBRANGE=(n)	SR=(	
HYR/RYR/ XYR	SUBRANGE=(n)	SR=(	
	NUMLEVEL=(n)	NL=(	
	LEVELS=(n)	LV=(	
	NUMYRPRS=(n)	NM=(	number of year pairs (year ranges)
	BEGYEAR=(n)	BY=(	beginning year of year range
	ENDYEAR=(n)	EY=(	ending year of year range
TRN	SUBRANGE=(n)	SR=(	
	TRUNCID=(n)	TR=(	ID number of truncation
	TRUNCABV=(c)	TA=(	truncation abbreviation
	TRUNNAME=(c)	TM=(	truncation name
	NUMVARBL=(n)	NV=(	# of variables
	VARBLNUM=(n)	VN=(	variable code numbers

Subfunction Model Definition Format, continued

YVR	SUBRANGE=(n)	SR=(	
	VARBLID=(n)	VI=(	ID # of variable
	VAROPER=(n)	VO=(	operand of variable
	VARVCODE=(n)	VV=(	code(s) of variables used to calculate variable
	VARSCODE=(n)	VS=(	variable subcode
	VARIKON=(n)	IK=(	integer constant
	VARLOCAT=(n)	VL=(	where variable is located
	VARFKON=(f)	FK=(	floating constant
XVR			
	SUBRANGE=(n)	SR=(	
	VARBLID=(n)	VI=(	
	VAROPER=(n)	VO=(	variable operand
	VARDEV=(n)	DV=(	variable deviation
	VARVCODE=(n)	VV=(	codes of variables used to calculate the variable
	VARSCODE=(n)	VS=(	variable subcode
	VARIKON=(n)	IK=(	integer constant
	VARLOCAT=(n)	VL=(	variable location
	VARFKON=(f)	FK=(	floating constant
YDA/MDA			
	SUBRANGE=(n)	SR=(	
	NUMLEVEL=(n)	NL=(	
	LEVELS=(n)	LV=(	
	WHEREGET=(n)	WG=(	where to find data
	USEDYET=(n)	UY=(	variable used yet?
	NUMVARBL=(n)	NV=(	# variables
	VARBCODE=(n)	VC=(	variable codes
	NUMSCODE=(n)	NO=(	# subcodes
	FIELDSIZ=(n)	FS=(	size of each element
	DIGIT=(n)	DG=(	location of decimal point
DFY			
	DEFID=(n)	FI=(	ID # of this model definition
	DEFLOCAT=(n)	FL=(	record # of this model definition
DFN			
	DEFID=(n)	FI=(	ID # of model definition
	DEFLOCAT=(n)	FL=(	record # of model definition
	NUMLEVEL=(n)	NL=(	
	LEVELS=(n)	LV=(	

Subfunction Model Definition Format, continued

ZDX	NUMLEVEL=(n)	NL=(	# of levels
	LEVELS=(n)	LV=(	level numbers
	UPPERAWC=(n)	UW=(	upper available water capacity
	LOWERAWC=(n)	LA=(	lower available water capacity
	UPPERPCP=(n)	UP=(	upper precipitation
	LOWERPCP=(n)	LP=(	lower precipitation
	SUBRANGE=(n)	SR=(	
REC	RECTYPE=(n)	RT=(	record type
	RECSPACE=(n)	RS=(	freespace on record (bytes)
	RELOCAT=(n)	RC=(	record #
ALL	SUBRANGE=(n)	SR=(	

3.2.32.9 Data Structure

NOT APPLICABLE

C-2

3.2.32.10 Subfunction Data Format

function	applicable section
DEF	DA1,BLK,DTA,DAY,REC
RPL	DA1,BLK,FMT,DAD
ADD	DA1,BLK,FMT,DAD,DTA,DAY,REC
LST	DA2,ALL,BLK

section	parm/subparm	abv.	description
DA1	DDNAME=(c)	DD=(	JCL DD name
	FILEID=(c)	ID=(	file identification name
	PASSWORD=(c)	PW=(	password to access file
	COUNTRY=(n)	CO=(	# of country
	DESID=(n)	SI=(	descriptor identification
	NUMSECS=(n)	NC=(	# of sections used
	SECTIONS=(c)	SC=(	section names
DA2	DDNAME=(c)	DD=(	JCL DD name
	FILEID=(c)	ID=(	file identification name
	PASSWORD=(c)	PW=(	password to access file
	COUNTRY=(n)	CO=(	# of country
	DESID=(n)	SI=(	descriptor identification
	NUMSECS=(n)	NC=(	# of sections used
	SECTIONS=(c)	SC=(	section names
	LEVELS=(n)	LV=(	level numbers
	NUMLEVEL=(n)	NL=(	
BLK	VRBLCODE=(n)	BC=(	variable code
	NUMELEM=(n)	NE=(	# subelements
	SUBECODE=(n)	EC=(	subelement code
	FORMAT=(n)	FM=(	format number
	INTLVAL=(n)	IV=(	input value
	SUBRANGE=(n)	SR=(	

Subfunction Data Format, continued

FMT	NUMENTRY=(n)	NA=(	# of fields on card
	LEADCOLM=(n)	LC=(	starting card column of field
	SIZE=(n)	SZ=(	size of field
	DATATYP=(C)	AT=(	data type
	SUBRANGE=(n)	SR=(	
DAD	INMED=(c)	IN=(	input medium
	DDNAME=(c)	DD=(	JCL DD name
	INRECLEN=(n)	RL=(	input record length
	ENDFILE=(c)	EF=(	end of file delimiter
	NUMLEVEL=(n)	NL=(	number of levels
	LEVELS=(n)	LV=(	level numbers
	LEVVARFY=(n)	VF=(	level varifier
DTA	SUBRANGE=(n)	SR=(	
	NUMLEVEL=(n)	NL=(	
	LEVELS=(n)	LV=(	
	DATLOCAT=(n)	TL=(	data record #
	DATDISPL=(n)	TP=(	data displacement
DAY	DATID=(n)	TI=(	directory ID # of each area with data on the file
	DATDESID=(n)	TD=(	ID # of descriptor for data in this block
	DATBLKSZ=(n)	BS=(	# bytes in data block
	DATBKALC=(n)	TB=(	total # blocks allocated
	DATBKUSE=(n)	TU=(	total # blocks used
	DATNOREC=(n)	TN=(	# records used for this directory element's data
	DATPOINT=(n)	TP=(	subscript of data location array
	DATLOCAT=(n)	TL=(	record # for data
	DATDISPL=(n)	TH=(	byte # of beginning of data
	DATRBALC=(n)	TO=(	# blocks allocated for this data on record
	DATRBUSE=(n)	TZ=(	# blocks used for this data on record
	DATFBKID=(n)	TF=(	first block ID
	DATNXARR=(n)	TX=(	pointer to next subscript of data location array
	SUBRANGE =(n)	SR=(	
REC	RECTYPE=(n)	RT=(	record type
	RECSPACE=(n)	RS=(	freespace in record (bytes)
	RECLOCAT=(n)	RC=(	record number
ALL	SUBRANGE=(n)	SR=(	

3.2.33 Parameter File Structures

3.2.33.1 Section File Record Structure

```
DCL 1     SECTFIL EXTERNAL,
          2 SUBFUN            CHAR(4),
          2 SECNAME(24)       CHAR(3),
          2 OPT(5,24)         FIXED BIN(15,0),
          2 ORDER(5,24)       FIXED BIN(15,0),
          2 RESERVED          CHAR(248),
          2 MAXTIME(5,24)     FIXED BIN(15,0);
```

3.2.33.2 Parameter File Record Structure

```
DCL 1     PARMFIL EXTERNAL,
          2 SFUNCTN          CHAR(4),
          2 SECTION          CHAR(4),
          2 PARM(24)         CHAR(10)VARYING,
          2 PABV(24)         CHAR(4),
          2 PLEN(24)         FIXED BIN(15,0),
          2 MAXNO(24)        FIXED BIN(15,0),
          2 MINNO(24)        FIXED BIN(15,0),
          2 OPTN(5,24)       FIXED BIN(15,0),
          2 TYPE FLAG(32)    BIT(1),
          2 RESERVED         CHAR(264);
```

3.2.34 Error Message File Structure

```
DCL 1    ERREC EXTERNAL
         2 ERROR          FIXED BIN(15,0),
         2 RESERVED       CHAR(2),
         2 MESSAGE        CHAR(116);
```

APPENDIX A

Parameter/Subparameter Names

Parameter/ Subparameter	Abv.	Description
BASE	BA	type of variable
BEGYEAR	BY	beginning year of year range
BLKLOCAT	BL	where variable is located in data block
BROTHER	BR	brother of directory element
CHILD	CH	child of directory element
CODEABV	CA	variable abbreviation
CODEID	CI	code # for variable
CODENAME	CN	variable name
CODLOCAT	CL	position in code array
COUNTRY	CO	country ID number
CROPNUMB	CR	crop ID number
DATATYP	AT	data type
DATBKALC	TB	total # blocks allocated
DATBKUSE	TU	total # blocks used
DATBLKSZ	BS	# bytes in data block
DATDESID	TD	ID # of descriptor for data in this block
DATDISPL	TH	byte # of beginning of data
DATFBKID	TF	first block ID
DATID	TI	directory ID # of each area with data
DATLOCAT	TL	record # for data
DATNOREC	TN	# records used for this directory element's data
DATNXARR	TX	pointer to next subscript of data location array
DATPOINT	TP	subscript of data location array
DATRBALC	TO	# blocks allocated for this data on record
DATRBUSE	TZ	# blocks used for this data on record
DDNAME	DD	JCL DD name
DEFDISPL	FP	byte # of beginning of model definition
DEFID	FI	ID # of each model definition
DEFLOCAT	FL	record # of each model definition
DESDISPL	SP	byte # of beginning of each descriptor
DESID	SI	IS # of descriptor on file
DESLOCAT	SL	record # of each descriptor
DIGIT	DG	location of decimal point
DIRID	RI	ID # of directory
DIRIDCHR	FC	alpha country codes
DIRIDNUM	DI	numeric country codes

Parameter/ Subparameter	Abv.	Description
DIRLOCAT	DL	directory record number
DIRNAME	DN	name of directory level
DIRNOREC	DR	# of directory records per country
DIRNUMDE	DE	# of directory elements per directory record
DIRTOTDE	DT	total # of directory elements per country
ELMLOCAT	EL	position in subelement array
ENDFILE	EF	end of file delimiter
ENDYEAR	EY	ending year of year range
FIELDSIZ	FS	size of each element
FILEID	ID	file ID name
FORMAT	FM	format number
INMED	IN	input medium
INRECLEN	RL	input record length
INTLVAL	IV	input value
LEADCOLM	LC	starting card column of field
LEVELS	LV	level numbers
LEVVARFY	VF	level verify
LOWERAWC	LA	lower available water capacity
LOWERPCP	LP	lower precipitation
MODLNAME	MN	model name
NOHYEARS	HY	# of history years
NOINVARB	IB	# raw variable cards
NOMETDAT	DM	# of met data cards
NORYEARS	RY	# of run years
NOSTRATA	NR	# of stratas
NOTRUNCS	NU	# of truncations
NOWEIGHT	WN	# of weights
NOXVARBL	NX	# of X-variables
NOXYEARS	XY	# of normal years
NOYLDDAT	DY	# of yield data cards
NOYVARBL	NY	# of Y-variables
NOZINDEX	ZX	# of Z-index cards
NOZONE	NZ	# of zones
NOZONSTA	ZS	total # of zones + strata
NUMBLKS	NB	# data blocks on record
NUMDAT	NT	# of directory elements with data
NUMDEF	NF	# of model definitions on file
NUMDES	NS	# of descriptors on file
NUMDIR	ND	# of countries on file
NUMELEM	NE	# of subelements
NUMENTRY	NA	# of fields on input card
NUMLEVEL	NL	# of levels
NUMPASS	NP	# of passwords
NUMREC	FR	# of records on file

Parameter/ Subparameter	Abv.	Description
NUMSCODE	NO	# of subcodes
NUMSECS	NC	# of sections
NUMVARBI	NV	# of variables
NUMYRPRS	NM	# of year pairs
NWGHCODE	NW	# of weight codes
NXTREC	NQ	address of next data record
OUTFIELD	OF	output print field size
PARENT	PA	parent of directory element
PASSWORD	PW	password
RELOCAT	RC	record #
RECSPACE	RS	freespace on record (bytes)
RECTYPE	RT	record type
RESERVED	RV	extra space or filler
SCALE	SE	location of decimal point
SECTIONS	SC	section name
SIZE	SZ	size of field
SUBECODE	EC	subelement code
SUBRANGE	SR	range of values in subscript
TOTCODES	TC	# of codes in descriptor
TOTUNITS	TS	# of units in descriptor
TRUNCABV	TA	truncation abbreviation
TRUNCID	TR	ID number of truncation
TRUNNAME	TM	truncation name
UNITABV	UA	variable unit abbreviation
UNITID	UI	variable unit code #
UNITNAME	UN	variable unit name
UNTLOCAT	UL	position in unit array
UPPERAWC	UW	upper available water capacity
UPPERPCP	UP	upper precipitation
USEDYET	UY	variable used yet?
VARBCODE	VC	variable code number
VARBLID	VI	ID# of variable
VARBLNUM	VN	variable code number
VARDEV	DV	variable deviation
VARFKON	FK	floating constant
VARIKON	IK	integer constant
VARLOCAT	VL	variable location
VAROPER	VO	variable operand
VARSCODE	VS	variable subcode
VARVCODE	VV	variable variable code
VRBLCODE	BC	variable code
WEIGHTS	WH	weights
WGHVARBL	WV	code # of variable to be weighted
WHEREGET	WG	where to find data