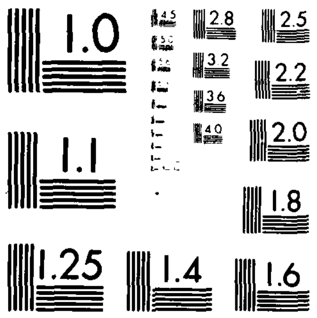
AD-A110 120   GEORGIA INST OF TECH ATLANTA   F/G 5/5
COMPUTER AIDED SPECIFICATION TESTING SYSTEM (CASTS). VOLUME I. --ETC(U)
MAR 81   D E HUMPHREY, D P MILLARD   DAAK70-79-D-0087

UNCLASSIFIED   NL

1 OF 2
AD
A110 120



MICROCOPY RESOLUTION TEST CHART

NATIONAL BUREAU OF STANDARDS 1963 A

22

AD A110120

AIRMICS

LEVEL I

US ARMY INSTITUTE FOR RESEARCH
IN MANAGEMENT INFORMATION AND COMPUTER SCIENC

COMPUTER SYSTEMS COMMAND
INFORMATION FOR DECISION
UNITED STATES ARMY

DTIC FILE COPY

CASTS
COMPUTER AIDED SPECIFICATION
TESTING SYSTEM

VOLUME I - SYSTEM DOCUMENTATION

MARCH 1981

01 26 82

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A[illegible]	3. RECIPIENT'S CATALOG NUMBER [illegible]

4. TITLE (and Subtitle)

CASTS
(SYSTEM DOCUMENTATION)

5. TYPE OF REPORT & PERIOD COVERED

R&D Technical Report Final

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)

D. E. Humphrey, D. P. Millard
EES, Georgia Institute of Technology

8. CONTRACT OR GRANT NUMBER(s)

DAAK70-79-D-0087
Task Order Number 0007

9. PERFORMING ORGANIZATION NAME AND ADDRESS

Georgia Institute of Technology
Atlanta, Georgia 30332

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS

US Army Institute for Research in Management
Information and Computer Science
115 O'Keefe Bldg., GIT
Atlanta, Georgia 30332

12. REPORT DATE

15 March 1981

13. NUMBER OF PAGES

183

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report)

Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

None

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release, distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer-Aided Design, Man-Machine Interface, Human Factors, Specification Testing, Interactive, Data Entry.

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

The Computer Aided Specifications Testing System (CASTS) is a tool to aid in the design of interactive systems. It is concerned with human factors and the man-machine interface during the initial capture of data and the editing of that data. CASTS is divided into two processes. The first process is that used by the designer to construct simulations of interactive video display terminal dialogue; the second process gathers the performance data of an actual user who runs the simulation designed in the first process.

COMPUTER AIDED SPECIFICATION TESTING SYSTEM

(CASTS)

SYSTEM DOCUMENTATION

EES/GIT Project A-2560

Prepared By

Computer Science and Technology Laboratory
Engineering Experiment Station
Georgia Institute of Technology
Atlanta, Georgia 30332

D. E. Humphrey
D. P. Millard

AIRMICS

Georgia Institute of Technology
Room 325, Hinman Research Building
Atlanta, Georgia 30332

Under

DAAK70-79-D-0087
Task Order #0007

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

1.0 INTRODUCTION

1.0 INTRODUCTION

The U.S. Army Institute for Research and Management Information and Computer Sciences (AIRMICS) has conducted research in the area of human factors and the man-machine interface. As a result of this research, a pilot system for testing the adequacy of functional specifications has been developed. The concepts demonstrated in this pilot system received a favorable review by several elements within the U.S. Army Computer Systems Command (USACSC). The Computer Science and Technology Laboratory of the Engineering Experiment Station, Georgia Institute of Technology, contracted with AIRMICS under Contract DAAK70-79-D-0087, Task Order #0007, to extend the research effort. The tasks contained in the Statement of Work have been designed to expand the concepts of the pilot system to create a tool for use by USACSC and the Army.

The Computer Aided Specification Testing System (CASTS) is a tool for the design of interactive data entry systems. The CASTS project is concerned specifically with the development of this tool. The work is divided into two processes within the CASTS system: The first process is that used by the designer to construct simulations of interactive video display terminal dialogue and the second process gathers the performance data of an actual user who runs the simulation designed in Process I of the system.

2.0 SYSTEM OVERVIEW AND DATA BASE DESCRIPTION

2.0 SYSTEM OVERVIEW AND DATA BASE DESCRIPTION

2.1 IMPLEMENTATION

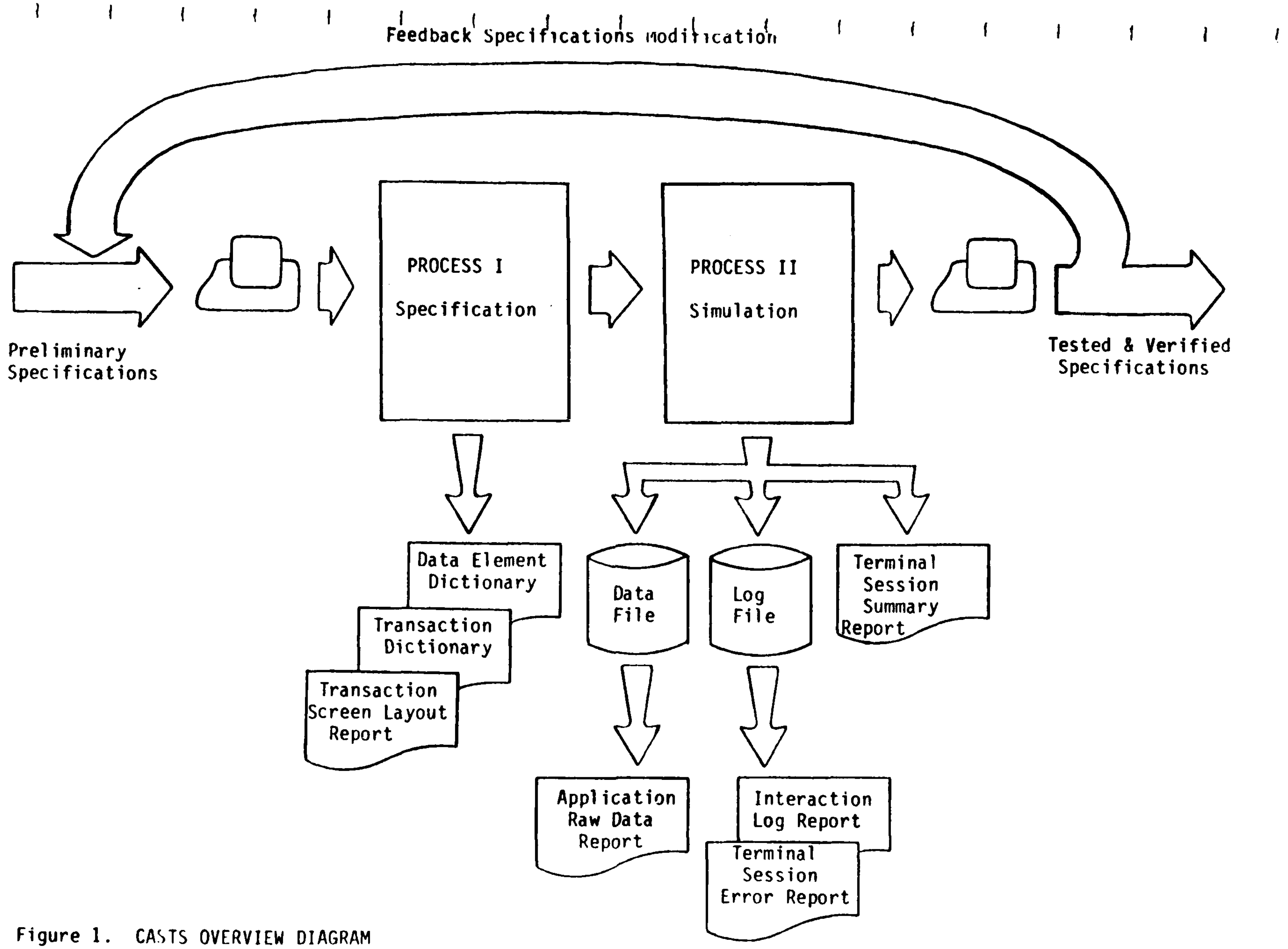
The CASTS system has been designed and implemented on a DEC PDP-11/70 running under the IAS operating system, on site at the AIRMICS installation at the Georgia Institute of Technology. The data base management system utilized for CASTS is DBMS-11. Most of the software for the system is written in ANSI X3.23 1974 COBOL. A few programs are written in FORTRAN IV, and are so designated in the documentation.

2.2 CASTS OVERVIEW

CASTS is a software system for use as a tool in the development of interactive data entry systems. It is composed of two major processes:

(1) Specification of a system (Process I)
(2) Simulation (Process II).

The system designer uses the Process I Specification interactively to model the system to be developed. This specification may be a rough approximation or it may be complete to minute detail. The Process II Simulation can then be run by the system designer or an end user to test the specifications under consideration. Process I may be reentered to add more detail, change or fine tune the specifications in response to problems encountered in the simulation process. This cycle can continue until the system designer is satisfied with the system specifications (see Figure 1).

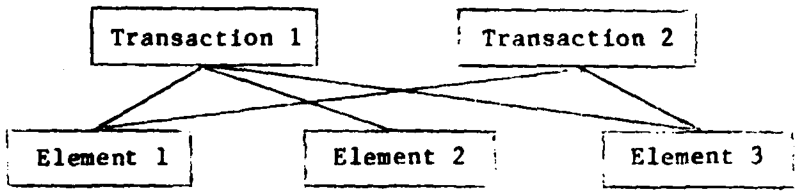


Feedback Specifications modification
Preliminary Specifications
PROCESS I
Specification
PROCESS II
Simulation
Tested & Verified Specifications
Data Element Dictionary
Transaction Dictionary
Transaction Screen Layout Report
Data File
Log File
Terminal Session Summary Report
Application Raw Data Report
Interaction Log Report
Terminal Session Error Report

Figure 1. CASTS OVERVIEW DIAGRAM

The basic element of the interactive system to be modeled is the transaction. A transaction may represent an employee record in a payroll application or a part record in an inventory application. A transaction consists of elements. Elements contain the actual data, such as employee identification number, employee name or part number.

The elements are related to the transactions in a network fashion. Each transaction is composed of several elements. However, several transactions may share the same elements. These relationships are maintained within a data base management system.



During the Process I Specification the system designer interactively describes the application system transactions, element prompts, element edit criteria and error messages. All or portions of this specification can be entered during the initial session. At any point in this dialogue, the system designer can terminate the session temporarily and receive reports detailing the current specifications or proceed to the Process II Simulation for system testing.

In the simulation process the specifications can be tested by entering sample transaction data in response to the element prompts. The transaction data entered will be checked in accordance with the edit criteria previously described. If no criteria has been specified for a particular element the simulation will accept any input, so as to allow incomplete specifications to be partially tested. The user of Process II may select one of several input modes for entering data, such as element by element prompting or a complete transaction at a time. A log of all user interaction with the system may be accumulated during the simulation, by time stamping all prompts, inputs and errors encountered. This log can be used to provide useful feedback information to the development process for the system under consideration.

A significant concept in CASTS is the capability for the system designer to return to the model specification phase of design to change specifications as often as necessary without major reentry of the same information. Iterative refinements allow testing and modification of specifications without investing considerable coding effort in the initial system design process. In addition, the system designer utilizes an interactive tool which can provide valuable experience in designing other interactive systems. Other benefits of CASTS include ease of use, a help/query capability, and the ability to build and test a sample data base. Ease of use is built into the interaction between CASTS and the user, and incorporated into the user documentation. The help capability allows the user to query CASTS for helpful information at any time it is needed.

2.3 DATA BASE DESCRIPTION

The function of the data base is to organize and store information entered in the Process I Specification of an interactive data entry system. The data base is interpreted to produce an interactive screen dialogue during the Process II Simulation. The screen dialogue is intended to be part of the input procedures of a more complete system. The information handled by the data base is organized into two general classifications:

(1) transactions
(2) data elements.

Transactions are built from data elements. Each element in the system has a unique identifier, a field edit criteria, a prompt that is used in a CRT dialogue, an error message, and a short narrative "HELP" message to provide information in addition to the prompt.

2.3.1 Schema Description

The data base schema, or structure, reflects the orientation of the system towards transactions (see Figure 2). An application may deal with several types of transactions, which are in turn composed of elements. Information dealing specifically with one transaction is stored in the TRANS-REC and includes the:

(1) transaction name
(2) application to which it belongs
(3) length
(4) screen partitioning data.

These records are accessed using the CALC method, i.e., directly using the transaction name as the key to the record. For more information on data base organization and access, see the DBMS-11 COBOL Data Manipulation Language Reference Manual.

Each element is associated with a TRANS-ELE-DEFN, which describes the positioning of the element prompt and data entry field on the screen. This screen information is used whenever the Process II end-user is simulating data entry in the screen input mode, as opposed to the direct or prompt modes. The association of the TRANS-ELE-DEFN with the related element is attained by its

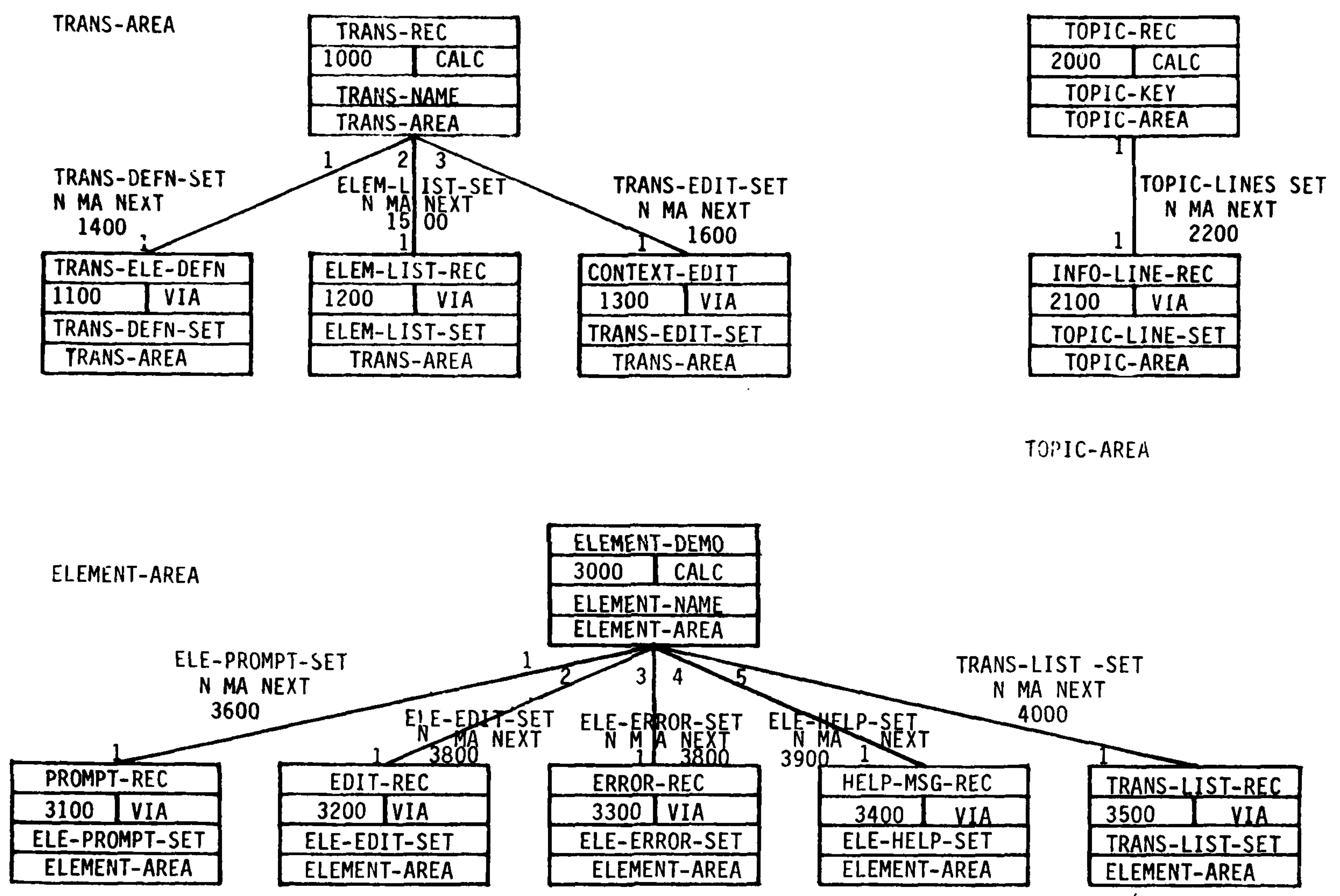


Figure 2. CASTS Schema Diagram

being in the corresponding position to the ELEM-LIST-REC within the respective sets. For example, the third TRANS-ELE-DEFN within the TRANS-DEFN-SET describes the screen position of the element named by the third ELEM-LIST-REC within the ELEM-LIST-SET.

ELEM-LIST-REC's are records containing the names of elements. These element names are used to locate the ELEMENT-DEMO (further discussion in later paragraph) records in CALC mode and other element information is obtained from there by using VIA <set-name>. The VIA data base access method allows access to records according to their membership and placement within sets.

The schema includes CONTEXT-EDIT records within a TRANS-EDIT-SET. This structure is unused in the final design of CASTS. It was originally intended to provide compatibility checks among elements, but was eliminated during project redesign.

ELEMENT-DEMO records contain the demographic information pertaining to an element. Fields of the record contain:

(1) the element name
(2) description
(3) proponent
(4) creation date
(5) creator identification.

The ELEMENT-DEMO record is retrieved using CALC with the element name as the key. It has five sets associated with it that describe various types of information relating to an element:

(1) prompts
(2) edit criteria
(3) error messages
(4) help messages
(5) a list of the transactions which include the element.

The PROMPT-REC contains at least one line that is described as a seventy-two (72) character string, for use as a prompt in the Process II simulation. The prompt may be more than one line long if it is specified by more than one PROMPT-REC. Its purpose is to inform the end-user as to which data element he is expected to enter.

The edit criteria (EDIT-REC) for an element specifies the element format, default value, and range or value checking The format is given by the system designer in Process I using format characters similar to the COBOL picture format without the keyword PICTURE (or PIC). There is a twenty character field reserved for format specification. Format characters are described in Section 5.1.1 of this document. The EDIT-REC also contains a seventy-two character string default value for the element, which may be used as data input if none is supplied by the user. The first EDIT-REC is required and Process I will not create an element without a format and default specification. Any succeeding EDIT-RECs provide either an upper and lower bound for range checking or a list of possible correct values for value checking. In these EDIT-RECs the format field indicates what type of value will be found in the default field i.e. upper bound, lower bound or valid value.

The error message, printed when an input error has been made by a Process II end-user, is specified during Process I in the ERROR-REC. This is a seventy-two character string. There may be more than one per element, allowing a three or four line error message, or no message at all.

The HELP-MSG-REC is similiar to the ERROR-REC in form and specification. Its purpose is to provide additional information upon request to the end-user concerning the to be entered.

The list of transactions in which the element participates is composed of TRANS-LIST-REC records. Each record contains the transaction name.

The TOPIC-AREA of the schema provides the mechanism for storing CASTS help messages on various topics. The TOPIC-REC contains the TOPIC-KEY by which the help messages can be located, using the CALC method. Individual lines within the help message are stored in the record INFO-LINE-REC within the TOPIC-LINES-SET.

2.3.2 Record Layout

The following is a description of the record layouts for the data base system:

<u>Record Layout</u>

TRANS-REC		Transaction Record
TRANS-NAME	X(40)	Transaction Name
TRANS-LOG	X	Unused
TRANS-LENGTH	9999	Transaction Length
APPL-NAME	X(40)	Application Name
ERROR-LOG-TYPE	99	Unused
ENTRY-START-LINE	99	Interaction or data entry area, start line
ENTRY-END-LINE	99	Interaction area end line
ERRMSG-START-LINE	99	Error Msg, start line
ERRMSG-END-LINE	99	Error Msg, end line
CMD-START-LINE	99	Unused
CMD-END-LINE	99	Unused
T-CREATE-DATE	X(6)	Creation date of trans
T-CREATOR-ID	X(9)	Creator ID
TRANS-ELE-DEFN		Transaction Related Element Definitions
START-PSN-TRANS	9999	Start Position within Transaction
SCREEN-PROMPT-X	99	X Coordinate on Screen in char positions
SCREEN-PROMPT-Y	99	Y Coordinate on Screen
PROMPT-CONTENTS	X(72)	Screen Prompt Contents
FIELD-X	99	X Coordinate on Screen of field
FIELD-Y	99	Y Coordinate on Screen of field

ELEM-LIST-REC		List of elements in Trans
ELEM-NAME-L	X(30)	Element name
CONTEXT-EDIT		Transaction Context Edit Info
COND-VALUE	X	Unused
FIRST-ELE-NAME	X(30)	Unused
FIRST-RELN	X	Unused
FIRST-VALUE	X(72)	Unused
OPERATOR	X	Unused
SECOND-ELE-NAME	X(30)	Unused
SECOND-RELN	X	Unused
SECOND-VALUE	X(72)	Unused
ELEMENT-DEMO		Element Demographic Info
ELEMENT-NAME	X(30)	Name of Element
ELEMENT-DESC	X(288)	Element Description
PROPONENT	X(40)	Name of Proponent
ELEMENT-NUMBER	X(6)	Identification Number
E-CREATE-DATE	X(6)	Creation Date of Element
E-CREATOR-ID	X(90	User ID of Creator
CONSTANT-FIELD	X	Is this field a constant? Y or N
PROMPT-REC		Element Prompt Info
CAR-CNTL	X	Unused
PROMPT-LINE	X(72)	Prompt Line Contents
EDIT-REC		Element Edit Criteria
ELE-FORMAT	X(20)	Element Format i.e. X,9
DEFAULT-VALUE	X(72)	Default Value
ERROR-REC		Element Error Info
ERR-MSG	X(72)	Error Message
HELP-MSG-REC		Element Help Info
HELP-MSG	X(72)	Help Message
TRANS-LIST-REC		
TRANS-NAME-L	X(40)	Trans name (this elem is in)
MEMBER-NUM	9999	Unused
TOPIC-REC		Help topics available
TOPIC-KEY		Unique to a topic
APPL-NAME-T	X(40)	Application name
TOPIC-NAME	X(30)	Topic name (typed by user)
H-CREATE-DATE	X(6)	Date help topic created
H-CREATOR-ID	X(9)	ID of creator
INFO-LINE-REC		Topic Info record
INFO-LINE	X(72)	One line of info

3.0 REPORTS FROM THE DATA BASE

3.0 REPORTS FROM DATABASE (PROCESS I)

The following reports pertain to the database, and should serve as a tool for the designer. They provide printed information on the current specifications as they are stored in the data base. See sample reports at the end of this section.

3.1 DATA ELEMENT DICTIONARY - ELERPT

This report details the data elements and their associated information with the following fields:

1. Element name
2. Element number
3. Length
4. Creation date
5. Creator identification
6. Proponent
7. Edit format
8. Default value
9. Description
10. Transaction cross reference

As an option within the Data Element Dictionary, one of the following categories of elements may be chosen:

1. All elements in the data base
2. Range of element numbers
3. All elements of one proponent
4. All elements in one application

Additionally, the elements may appear in order as either:

1. Numeric by element number
2. Alphabetic by element name

3.2 TRANSACTION DICTIONARY - TRARPT

The Transaction Dictionary reports on the transactions and information related to them. The transactions appear in alphabetical order by transaction name. The Transaction Dictionary will provide the following information:

1. Transaction name
   (A transaction may consist of one or more elements)
2. Application
3. Length
4. Creation date
5. Creator identification
6. Element name
7. Element number
8. Prompt message
9. Error message
10. Help message

This report may optionally consist of:

1. All transactions in the data base
2. All transactions in an application

3.3 TRANSACTION SCREEN LAYOUT - SCRRPT

This report will provide a printed version of the individual transactions as they would appear during Process II in the screen mode of data entry. The screen layout of each transaction shows the transaction name, application and the placement of elements and their associated data fields.

The Transaction Screen Layout Report may consist of:

1. All transactions in the data base
2. All transactions in an application
3. One transaction

Screen displays are reported in alphabetical order by transaction name. Some transactions may not have a specification for a screen layout.

```
                                        COMPUTER AIDED SPECIFICATION TESTING SYSTEM
                                        *******************************************
DATE 81/03/06                                        AIRMICS GIT                                             PAGE     1

                                                DATA ELEMENT DICTIONARY
                                                ***********************

ALL ELEMENTS IN DATA BASE
                                           2         3          4           5                          6
  ELEMENT NAME--------------------------NUMBER--LENGTH--CREATE DATE--CREATOR ID--CONSTANT--PROPONENT---------------------------
1 ACCOUNT NUMBER                        CS-003    0001     80/11/25     DAN              NO    COMPUTER SCIENCE DIVISION
   7     FORMAT:        99999
   8     DEFAULT:
         DESCRIPTION:   ACCOUNT NUMBER IS A NUMBER FROM 1 TO THE MAXIMUM NUMBER OF
   9                    ACCOUNTS PERMITTED IN THE SYSTEM.  IT IS ASSIGNED BY THE SYSTEM
                        DURING THE PROCESSING OF A TRANSACTION TO ESTABLISH THE ACCOUNT.
                        IT MUST BE USED ON ALL TRANSACTIONS INVOLVING CHANGES TO ACCT STATUS
         TRANS X REF:   NEW DEPOSIT
  10                    DEPOSIT
                        CHECK

ELEMENT NAME--------------------------NUMBER--LENGTH--CREATE DATE--CREATOR ID--CONSTANT--PROPONENT---------------------------
BENT LEG SIT UP                         40        0003     81/01/21     GLENN            NO    AIRMICS
         FORMAT:        999
         DEFAULT:       000
         DESCRIPTION:   NUMBER OF BENT LEG SIT UPS COMPLETED
         TRANS X REF:   BPT

ELEMENT NAME--------------------------NUMBER--LENGTH--CREATE DATE--CREATOR ID--CONSTANT--PROPONENT---------------------------
CADET NUMBER                            20        0004     81/01/21     GLENN            NO    AIRMICS
         FORMAT:        9999
         DEFAULT:       9999
         DESCRIPTION:   CADET PERSONNEL NUMBER
         TRANS X REF:   AGC
                        BPT

ELEMENT NAME--------------------------NUMBER--LENGTH--CREATE DATE--CREATOR ID--CONSTANT--PROPONENT---------------------------
CHECK NUMBER                            CS-001    0004     80/11/25     DAN              NO    AIRMICS
         FORMAT:        9999
         DEFAULT:
         LOWER BOUND:   0
         UPPER BOUND:   9999
         DESCRIPTION:   CHECK NUMBER IS THE IDENTIFYING ELEMENT FOR ALL CHECKS FOR A PARTICULAR
                        ACCOUNT.  IT MUST BE UNIQUE WITHIN A GIVEN MONTH AND IN THE RANGE FROM
                        0 TO 9999.  ONLY NUMERIC CHARACTERS ARE ALLOWED.
         TRANS X REF:   NEW DEPOSIT
                        CHECK

ELEMENT NAME--------------------------NUMBER--LENGTH--CREATE DATE--CREATOR ID--CONSTANT--PROPONENT---------------------------
DECK                                    000124    0005     81/01/05     SAM              NO    CSTL
         FORMAT:        99999
         DEFAULT:       55555
         VALID VALUE:   55555
         VALID VALUE:   22222
         VALID VALUE:   11111
```

```
                                          COMPUTER AIDED SPECIFICATION TESTING SYSTEM
DATE 81/03/06                             *******************************************
                                                      AIRMICS GIT                                  PAGE    1

                                                  TRANSACTION DICTIONARY
                                                  **********************

ALL TRANSACTIONS IN DATA BASE
 1                                          2                                              3       4          5
TRANSACTION NAME---------------------------APPLICATION--------------------------------LENGTH----CREATED----CREATOR----
AGI                                         AGC                                           16    81/01/22   DAVID

 6  -----ELEMENTS-----
    UNIT ASSIGNED                      7   10
             8 PROMPT:            PLEASE ENTER UNIT ASSIGNED TO
                                  4-POSITION ALPHA-NUMERIC
             9 ERROR MESSAGE: SYNTAX ERROR
                                  THIS FIELD IS 4 CHARACTERS A/N

    CADET NUMBER                           20
               PROMPT:            ENTER CADET NUMBER(9999)
               ERROR MESSAGE: THIS FIELD IS A 4POSITION NUMERIC

    STATUS                                 222
               PROMPT:            ENTER STATUS
               ERROR MESSAGE: ERROR STATUS
            10 HELP MESSAGE:  STATUS FORMAT "X"

    SOURCE CODE                            5
               PROMPT:            ENTER SOURCE CODE
               ERROR MESSAGE: ERROR SOURCE CODE
               HELP MESSAGE:  SOURCE CODE FORMAT IS "9"

    GPA                                    5
               PROMPT:            ENTER GPA
               ERROR MESSAGE: ERROR GPA
               HELP MESSAGE:  GPA FORMAT IS "9"

    SCAT/CEEB TEST                         6
               PROMPT:            ENTER SCAT/CEEB TEST >>>>
               ERROR MESSAGE: ERROR TEST
               HELP MESSAGE:  TEST FORMAT IS "999"
TRANSACTION NAME---------------------------APPLICATION--------------------------------LENGTH----CREATED----CREATOR----
AFI                                         AIRMICS                                       18    81/01/21   GLENN

    -----ELEMENTS-----

    UNIT ASSIGNED                          10
               PROMPT:            PLEASE ENTER UNIT ASSIGNED TO
                                  4-POSITION ALPHA-NUMERIC
               ERROR MESSAGE: SYNTAX ERROR
                                  THIS FIELD IS 4 CHARACTERS A/N

    CADET NUMBER                           20
```

```
                              COMPUTER AIDED SPECIFICATION TESTING SYSTEM
                              *******************************************
DATE 81/03/26                               AIRMICS GIT                                        PAGE    1

                                      TRANSACTION SCREEN LAYOUT
                                      *************************

ALL TRANSACTIONS IN DATA BASE WITH SCREEN DEFINITIONS

HPT

                1         2         3         4         5         6         7         8
       12345678901234567890123456789012345678901234567890123456789012345678901234567890
      ----------------------------------------------------------------------------------
   01 :        ENTER UNIT ASSIGNED          XXXX                                        : 01      USER INTERACTION AREA
   02 :                                                                                 :
   03 :        CADET NUMBER?                9999                                        :
   04 :                                                                                 :
   05 :        NUMBER OF PUSH UPS?          999                                         :
   06 :                                                                                 :
   07 :        HOW MANY BENT LEG SIT UPS?   999                                         :
   08 :                                                                                 :
   09 :        MILE RUN- IN SECONDS?        999                                         :
   10 :                                                                                 :
   11 :        WAIVER CODE?                 A                                           :
   12 :                                                                                 :
   13 :                                                                                 :
   14 :                                                                                 :
   15 :                                                                                 : 15
   16 :                                                                                 :
   17 :                                                                                 : 17      ERROR MESSAGE AREA
   18 :                                                                                 :
   19 :                                                                                 : 19
   20 :                                                                                 :
   21 :                                                                                 :
   22 :                                                                                 :
   23 :                                                                                 :
   24 :                                                                                 :
      ----------------------------------------------------------------------------------

APPLICATION: AIRMICS
```

4.0 REPORTS FROM THE SIMULATION

4.0 REPORTS FROM THE SIMULATION (PROCESS II)

The reports pertaining to the simulation process provide performance information in the form of user interaction records and statistics. They are intended to assist the system designer in the evaluation of the specifications and any useful modifications. See sample reports at the end of this section.

4.1 TERMINAL SESSION SUMMARY REPORT

This report provides a summary of the user interaction with the simulation process during a single terminal session.

This report shall consist of:

1. Transactions used in this session
2. User ID with his time in/out
3. The number of transactions for which data has been entered
4. The number of elements, prompts, and error messages used during the session
5. Time per transactions

The report is to be short and automatically produced in a file, <user id>.SRF, at the end of the session. It provides a fixed set of statistical data. The report may be printed by, PRINT <user id>.SRF.

4.2 TERMINAL SESSION ERROR REPORT - ERRRPT

ERRRPT contains the errors made during the user interaction with the simulation process, reported in order by transaction name and transaction header time stamp.

The error messages are logged during simulation with the following information:

1. Transaction name and time stamp
2. Error logged, along with the name of element in error and the associated incorrect data input

The report may be produced by running the report generator, RUN ERRRPT, entering the <user id> of the session to be reported, and then printing the generated file, PRINT <user id>.ERF.

4.3 INTERACTION LOG REPORT - LOGRPT

This report is a printed version of the simulation interaction log. It consists of the complete interactive session with time stamps. The report may be produced by running the report generator, RUN LOGRPT, entering the <user id> of the session, and printing the report, PRINT <user id>.LRF.

4.4 APPLICATION RAW DATA REPORT - DATRPT

This report shall consist of a literal picture of the transactions as they are input by the user at the keyboard during the simulation.

The report may be produced by running the report generator, RUN DATRPT, entering the <user id> of the session, and printing the report, PRINT <user id>.DRF.

```
                                      COMPUTER AIDED SPECIFICATION TESTING SYSTEM
                                      *******************************************
     DATE 81/03/06                                AIRMICS GIT                                    PAGE     1

                                          TERMINAL SESSION SUMMARY REPORT
                                          *******************************
            (2)                                                               (2)
 USER ID: DH                                                         TIME IN 17:49:46   TIME OUT 17:58:16

     APPLICATION :  AIRMICS
                                                 (5)                                     (4)
                                       NUMBER  AVG TIME   NUMBER
                                        TRANS  PER TRANS  ELEMENTS      PROMPTS      ERROR MSGS     HELP MSGS
        TRANSACTIONS USED              ENTERED   (SEC)    IN TRANS   TOTAL  /ELEM   TOTAL  /ELEM   TOTAL  /ELEM
   (1)  -----------------              ---------------------------------------------------------------------------
                                      (3)
        AGC                                6       27         6        47   1.31      11   0.31        0   0.00
        HFT                                8       26         6        58   1.21      10   0.21        0   0.00
        ORDER-STOCK                        9       11         2        18   1.00       7   0.39        0   0.00

              3 DIFFERENT TRANSACTIONS ENTERED   ---    -----       ---     -----  -----   -----  -----   -----  -----

                          GRAND TOTALS            23       64        14       123   3.52      28   0.91        0   0.00

                     AVERAGE/TRANSACTION           8       21         5        41   1.17       9   0.30        0   0.00
```

```
                                   COMPUTER AIDED SPECIFICATION TESTING SYSTEM
                                   *******************************************
DATE 81/03/06                                  AIRMICS GIT   .                               PAGE     1

                                          TERMINAL SESSION ERROR REPORT
                                          *****************************

USER ID: 08


APPLICATION: AGC

TRANSACTION: AGC (1)

    1    17:32:06

         ELEMENT: UNIT ASSIGNED
             ERROR CONDITION:  NO SPECIAL CHARACTERS ALLOWED
     (2)→    BAD DATA DUMP:    &&&

         ELEMENT: CADET NUMBER
             ERROR CONDITION:  DATA MUST BE NUMERIC
             BAD DATA DUMP:    AX
             ERROR CONDITION:  DATA MUST BE NUMERIC
             BAD DATA DUMP:    890-3

         ELEMENT: STATUS
             ERROR CONDITION:  NO SPECIAL CHARACTERS ALLOWED
             BAD DATA DUMP:    *

         ELEMENT: SOURCE CODE
             ERROR CONDITION:  DATA MUST BE NUMERIC
             BAD DATA DUMP:    /


APPLICATION: AIRMICS

TRANSACTION: HPT

    1    17:21:09

         ELEMENT: CADET NUMBER
             ERROR CONDITION:  DATA MUST BE NUMERIC
             BAD DATA DUMP:    3

         ELEMENT: PUSH UP
             ERROR CONDITION:  DATA MUST BE NUMERIC
             BAD DATA DUMP:    50

    2    17:28:23

         ELEMENT: CADET NUMBER
```

```
                                              COMPUTER AIDED SPECIFICATION TESTING SYSTEM
                                              *******************************************
DATE 81/03/05                                           AIRMICS GIT                                        PAGE    1

                                                   INTERACTION LOG REPORT
                                                   **********************

USER ID: PB

APPLICATION :  AIRMICS

TRANSACTION       TIME          TYPE          TEXT
-----------       ----          ----          ----

AGC
                17:49:58      PROMPTS               CURRENT TRANSACTION : AGC
                17:50:00      PROMPTS       PLEASE ENTER UNIT ASSIGNED TO
                17:50:00      PROMPTS       4-POSITION ALPHA-NUMERIC
                17:50:03      INPUT         1234
                17:50:04      PROMPTS       ENTER CADET NUMBER(9999)
                17:50:07      INPUT         1234
                17:50:08      PROMPTS       ENTER STATUS
                17:50:09      INPUT         W
                17:50:10      PROMPTS       ENTER SOURCE CODE
                17:50:11      INPUT         E
                17:50:11      ERROR         >>> ERROR IN THIS ELEMENT ENTRY, COLUMN  1 -->>
                17:50:11      ERROR         E
                17:50:12      ERROR         ERROR SOURCE CODE
                17:50:12      ERROR         DATA MUST BE NUMERIC
                17:50:12      PROMPTS       ENTER SOURCE CODE
                17:50:13      INPUT         5
                17:50:14      PROMPTS       ENTER GPA
                17:50:16      INPUT         3
                17:50:17      PROMPTS       ENTER SCAT/CEEB TEST >>>>
                17:50:19      INPUT         A
                17:50:20      ERROR         >>> ERROR IN THIS ELEMENT ENTRY, COLUMN  1 -->>
                17:50:20      ERROR         A
                17:50:20      ERROR         ERROR TEST
                17:50:20      ERROR         DATA MUST BE NUMERIC
                17:50:21      PROMPTS       ENTER SCAT/CEEB TEST >>>>
                17:50:22      INPUT         123
                17:50:24      PROMPTS               CURRENT TRANSACTION : AGC
                17:50:24      PROMPTS       PLEASE ENTER UNIT ASSIGNED TO
                17:50:25      PROMPTS       4-POSITION ALPHA-NUMERIC
                17:50:37      INPUT         01UR8
                17:50:38      PROMPTS       ENTER CADET NUMBER(9999)
                17:50:40      INPUT         12345
                17:50:41      PROMPTS       ENTER STATUS
                17:50:45      INPUT         R
                17:50:45      PROMPTS       ENTER SOURCE CODE
```

```
                                          COMPUTER AIDED SPECIFICATION TESTING SYSTEM
                                          *******************************************
DATE 81/03/06                                            AIRMICS CIT                                   PAGE    1

                                                 APPLICATION RAW DATA REPORT
                                                 ***************************

USER ID: DF

APPLICATION :  AIRMICS

TRANSACTION
-----------

AGC                                         LENGTH:    16


                           1         2         3         4         5         6         7         8         9         1
                  1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
                                                                                                                   0
           1      12341234W54  123
           2      01UR1234R32  666
           3      01988907842  796
           4      78807890432  500
           5      56798889541  456
           6      908 7776732  234
SPI                                         LENGTH:    18


                           1         2         3         4         5         6         7         8         9         1
                  1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
                                                                                                                   0
           1      89071234060076360Z
           2      78006789067056400A
           3      8J075678080067500S
           4      78903456012013700R
           5      89077890234123500A
           6      78902345050050360A
           7      23452345050050400A
           8      23452345050050500Z
ORDER-STOCK                                 LENGTH:    51


                           1         2         3         4         5         6         7         8         9         1
                  1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
                                                                                                                   0
           1                         555555555                    20
           2                         12345678                     20
```

5.0 PROGRAM SUPPORT DOCUMENTS

5.0 PROGRAM SUPPORT DOCUMENTS

The programs that comprise the CASTS system are supported by internal program documentation, program narratives and hierarchy diagrams. The program narratives describe the programs, their organization and internal functioning including internal codes and error conditions. The hierarchy diagrams show graphically the functional hierarchy within each program and can be found in the appendices.

5.1 PROCESS I PROGRAM DESCRIPTIONS

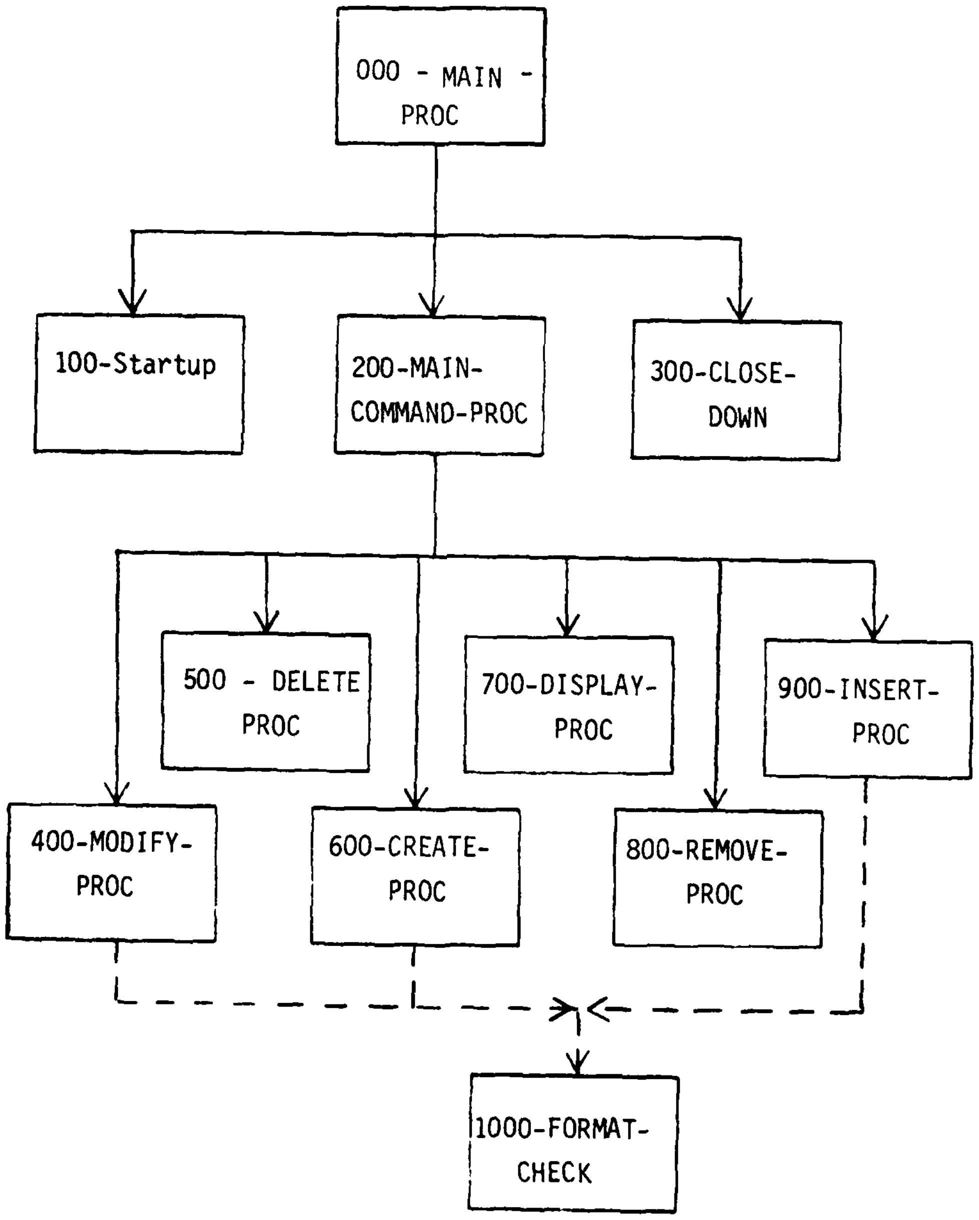
5.1.1 Process I Specification - CASTS1

The Process I Specification is the largest single program in the CASTS system. It is used by a system designer to model an interactive system. It is itself interactive and is written in COBOL. The specifications entered through the use of this program are stored in the data base, a description of which can be found in Section 1.3 of this document. The Process I Design Overview can be seen in Figure 3.

Process I provides an interface between the human system designer and the data that specifies an interactive system. The man-machine interface handles input to Process I with the COBOL verb ACCEPT. Output is sent to the terminal via the file TERM-OUT, which is assigned to the terminal. There are no other files utilized by Process I. The interaction with the data base is handled with the Data Manipulation Language for DBMS-11.

The Process I program operates as a command processor. It accepts input commands from the user at a terminal and then performs operations in response to the given commands. The start-up procedures, initializations, command

Figure 3. PROCESS I DESIGN OVERVIEW



decoding and terminal input/output procedures are located in the MAIN-PROGRAM section. The procedures to perform the different commands are each in separate sections named after the command.

Since the program is large it requires the use of overlays in order to run on the PDP 11/70. COBOL Segmentation is used to provide overlays. As stated by the COBOL Reference Manual and User's Guide, the non-overlayable or root segment consists of the COBOL Data Division and any section give› › number less than the segment limit. DEC COBOL allows different overlayable segments to reference each other as if they were all resident in storage together. The segment limit in Process I is 05. The Data Division and the MAIN-PROGRAM section (00) belong to the root segment and remain available without changing overlays throughout a program run. Hence the most often used routines are collected in the MAIN-PROGRAM section. The remaining sections compose overlayable segments and are assigned segment numbers as follows:

MODIFY-PROC	10
GET-PROCS	10
UPDATE-PROCS	12
DELETE-PROC	15
CREATE-PROC	20
DISPLAY-PROC	25
REMOVE-PROC	30
INSERT-PROC	35
DB-DUMP	40
FORMAT-PROC	45

For more information on COBOL overlays, see DEC PDP-11 COBOL Users' Guide, Chapter 9 Segmentation.

The MAIN-PROGRAM section performs the start-up procedure, requests an identification from the user and checks it and then performs the MAIN-COMMAND-PROC until it encounters an END command. The valid commands are HELP, BUILD, CREATE, DELETE, DISPLAY, MODIFY, REMOVE and INSERT. For information on the use and formats of these commands, refer to the CASTS

User's Manual. One additional command exists, DB, which is a debugging feature and causes status information to be displayed following each command. DB operates in a toggle fashion to turn on or off the debugging feature. It is useful only in system maintenance and/or enhancement development for problem solving. All of the remaining commands are for use in specifying an interactive system. They consist of two-field commands and three-field commands. The two-field commands are HELP, BUILD and CREATE. DELETE, DISPLAY, MODIFY, REMOVE and INSERT are three-field commands.

The data structure, COMMAND-CONTROL-BLOCK, holds information about the current command and the previous command. The previous command is used in the reference back feature so that an empty command position is automatically filled with the previous value. All commands are significant only to the first two characters.

The two paragraphs 231-GET-2FIELD-CMD and 232-GET-3FIELD-CMD make sure that the user has given a value for each command field and checks that only valid combinations of command name and item name have been used. (Command fields and item names are discussed in the CASTS User's Manual, Section 4.3.)

The 270-HELP-DISPLAY-PROC handles the display of help messages. Process I system help messages are stored in the data base in the TOPIC-AREA. The key for each help message consists of the PROCI-ID and a topic name of 40 characters. The user types a "?" any time he wishes further information on what input he is supposed to enter. The user may also type the question mark followed by a slash and a topic name:

```
?/<topic name>
```

to get information about a topic other than the specific input required. At the command level, HELP/<topic name> is a request for help information.

The error handling in Process I occurs on two levels. The high level data base errors are handled by 260-ERROR-PROC. Less important errors or those that can be corrected by reentry are handled at the point where they could occur. The RETURN-STATUS is a code for invalid data base access in response to a user command. The following values have meanings as indicated:

000	Initial
001	Complete
002	No transaction found with name given
003	Invalid item
004	Item not found
005	Data base error
006	No element found with name given

The code "005" is used primarily to check for non-recoverable data base errors that may have occurred when trying to GET a record subsequent to FINDing that record successfully. (For information regarding the DML, refer to the DBMS-11 COBOL Data Manipulation Language Reference Manual.)

The MODIFY-PROC section handles the modification of data that already exists in the data base, i.e., changes to specifications. If changes are requested in the transaction header, element header, screen definition or edit records the "options" for modifications are displayed and the user is requested to select one. This in effect selects the field within the record to be modified. The other records have only one field. In the transaction header no modifications are allowed to the transaction length since this is completely maintained by the program in response to insertion and removal of elements in a transaction. Modification of the constant field indicator for an element such that it changes to a constant (CONSTANT-FIELD in ELEMENT-DEMO becomes "Y") causes all range and value edit checks to be deleted, since the element is now a constant. Modifications to any of the screen boundary specifications in TRANS-ELE-DEFN are accompanied by a warning. It is left to the user to ensure that these boundaries are correct and make sense.

For each field that is to be modified, the current contents are displayed. The user is prompted for the new value and the data is modified in the data base records.

The GET-PROCS section includes all of the routines that are used to locate and access information in the data base as well as "display" routines to display information on the terminal. The "get" routines access the data base using the DML FIND and GET verbs and check the status of the data base subsequent to each access. If an error occurred, the program variable RETURN-STATUS is set to indicate the error.

The "display" routines are used to display the contents of the data base records, i.e., the specifications, on the terminal. These routines are used by several of the commands but are most heavily used by the DISPLAY command.

The routines that accept and check the specifications entered are in the UPDATE-PROCS section. These routines are used both when the data is initially entered and when it is modified. If the data entered is not valid, the program reprompts until the user enters valid data. The acceptable values are contained in the following table:

Field Formats and Valid Values

TRANS-REC

TRANS-NAME	40 characters
TRANS-LENGTH	$x \le 504$, numeric
APPL-NAME	40 characters
ENTRY-START-LINE	$1 \le x \le 20$, numeric
ENTRY-END-LINE	$1 \le x \le 21$, numeric, $x >$ ENTRY-START-LINE
ERMSG-START-LINE	$1 \le x \le 2$,, numeric, $x >$ ENTRY-END-LINE
ERMSG-END-LINE	$1 \le x \le 23$, numeric, $x >$ ERMSG-START-LINE
T-CREATE-DATE	6 digits
T-CREATOR-ID	9 characters, no special characters or all ƀ

TRANS-ELE-DEFN

START-PSN-TRANS	$x \le 504$, numeric
SCREEN-PROMPT-X	$1 \le x \le 72$, numeric
SCREEN-PROMPT-Y	ENTRY-START-LINE $\le x \le$ ENTRY-END-LINE, numeric
PROMPT-CONTENTS	72 characters
FIELD-X	$1 \le x \le 72$, numeric
FIELD-Y	ENTRY-START-LINE $\le x \le$ ENTRY-END-LINE, numeric

ELEMENT-DEMO

ELEMENT-NAME	30 characters
ELEMENT-DESC	288 charcters
PROPONENT	40 characters
ELEMENT-NUMBER	6 characters
CONSTANT-FIELD	Y, N
ELEMENT-LENGTH	$1 \le x \le 72$, numeric
E-CREATE-DATE	6 digits
E-CREATOR-ID	9 characters, no special characters or all ƀ

PROMPT-REC

PROMPT-LINE	72 characters

EDIT-REC

ELEMENT-FORMAT	Z, X, 9, A, ., ƀ, S
ELEMENT-DEFAULT	72 characters

ERROR-REC

ERR-MSG	72 characters

HELP-MSG-REC	
HELP-MSG	72 characters
TRANS-LIST-REC	
TRANS-NAME-L	40 characters
ELEM-LIST-REC	
ELEM-NAME-L	30 characters

The UPDATE-PROCS section also includes the routine to replace a data base record containing modifications.

The DELETE-PROC section performs deletions of elements, transactions and prompts, edit criteria, error messages and help messages for an element. An entire element cannot be deleted unless it does not participate in any transaction. If the user tries to delete such an element a message is printed that includes a list of the transactions containing the element in question. In addition, elements and transactions may only be deleted by a user with a user identification that matches the creator identification. Element deletion is accomplished by paragraph 510-DELETE-ELEM.

Transaction deletion is performed by 520-DELETE-TRANS. To delete a transaction, all elements in that transaction are removed and the transaction skeleton, consisting of the TRANS-REC and associated TRANS-ELE-DEFNs and ELEM-LIST-RECs, is erased from the data base. The elements remain intact and the name of the transaction deleted is removed from each of their lists of transactions in which the element participates. In other words, each element is disconnected from the transaction prior to the transaction deletion.

The 530-DELETE-ELE-MEMBERS handles the deletion of edit criteria and individual lines of prompt, error and help messages as well as the entire message. To reference a particular record in any of the sets; ELE-PROMPT-SET, ELE-EDIT-SET, ELE-ERROR-SET and ELE-HELP-SET, the item index is used. This number is given by the user as part of the item name in the second command field of three-field commands. If no item index is given (RECORD-COUNT = 0) all of the records in the set are deleted. The exception is the ELE-EDIT-SET, in which the first record containing the element format and default value must always be present. Thus, an item index of 1 for the item name EDIT is illegal. In all cases a successful deletion is indicated by a message on the terminal.

The BUILD and CREATE commands are both handled by the CREATE-PROC section. To store the TRANS-REC and the ELEMENT-DEMO in the data base, the same DML STORE verb is used. The distinction between the commands is made to reflect the logical difference between creating an element and putting together or building a transaction of several elements.

A transaction must be built with elements that already exist. If the user enters the name of an element that does not exist in the data base, a message is printed and the user is given the opportunity to create the element. After creating the element the user continues in the build process. The 637-OVERLAP-CHECK-AND-INCLUDE paragraph checks for element overlap within a transaction.

The 620-CREATE-ELEMENTS creates an element by creating the element header and the associated records that comprise the prompt message, edit criteria, error message and help message.

The edit criteria contains an element format, which specifies the format of the data considered to be valid input data for the element. The format characters are:

9	Numeric (0-9)
A	Alphabetic (A-Z)
X	Alphanumeric (0-9, A-Z)
Z	Zero suppression (0, space)
.	Decimal point
S	Sign (+, -, space)

Any character other than these is assumed to be a special character and the value in that position must be that special character. If no format is specified then data entered for the element will not be checked.

The DISPLAY-PROC section provides for the display of current information in the data base upon the terminal. The user may display the transaction, the element, a list of transactions to which an element belongs, the screen definition for an element in a transaction or any of the prompt, edit, error or help messages for an element. The 710-DISPLAY-RECORDS paragraph uses 430-DISPLAY-FIELD-CONTENTS in the GET-PROCS section to print the information on the terminal screen.

An element is removed from a transaction by the REMOVE-PROC section. It must be deleted from the list of elements belonging to a transaction, ELEM-LIST-REC. This does not delete the element. Elements may therefore exist without belonging to any transaction. The associated screen position information, TRANS-ELE-DEFN, must also be deleted. The name of the transaction must be deleted from the list of transactions to which the element belongs, TRANS-LIST-REC.

Element insertion into a transaction is handled by the INSERT-PROC section. The insertion process involves checking that the element is not inserted in a position on top of another element. The insertion of an element is not performed if it is determined that it will overlap another element.

Insertion of individual lines into prompts, edit criteria, error messages and help messages is handled in the INSERT-PROC section by the paragraph 920-INSERT-TO-ELEMENT. The item index given is the line after which the new line is inserted. An item index of zero will insert before the first line. An item index of zero is invalid for the edit criteria since the user may insert edit checks only after the first EDIT-REC, containing the element format and default value.

The FORMAT-CHECK section is utilized by several other sections. It checks that default values, range bounds and value check values entered conform to the format given for the element. The DATA-COND-PTR has the value GOOD-DATA when the format check determines that the input data matches the format. FORMAT-CHECK is peformed from MODIFY-PROC, CREATE-PROC and INSERT-PROC.

5.1.2 Process I Reports

The reports for Process I are each produced by a different program. These programs are similar due to their similar function. This section describes the features that are shared by all three Process I report programs.

The programs are written in COBOL and run interactively. They each utilize two files. One file is required for sorting purposes, SORTFIL.DAT. The other file contains the formatted output report, XXXXXX.TMP, where XXXXXX is the abbreviated name of the particular report. The program structure is similar for each program. The variable initializations, and program set up operations are performed. The report options are displayed on the terminal. The user is requested to choose amongst several options. For the option chosen, the appropriate transactions or elements and their associated information are retrieved from the data base. Records containing the information are released to the sort routine as the data is retrieved. Following the sorting process, the records are in sequential order as they should appear on the final report. As each record is returned from the sort, it is formatted into an output line of the report and written to the output report file.

Each program is organized in a hierarchical fashion, with a main line routine controlling the flow of execution. There are three COBOL sections, including the MAIN-PROGRAM, ELEM-RETRIEVAL or TRANS-RETRIEVAL depending on the program, and PRINT-OUT sections. Currently, the SORT, RELEASE and RETURN actions are accomplished by calls to external system routines. This is because the DEC COBOL does not have the verbs SORT, RELEASE and RETURN. In order to facilitate conversion to an environment in which the COBOL compiler supports the internal SORT, RELEASE and RETURN verbs, the ELEM-RETRIEVAL or TRANS-RETRIEVAL and PRINT-OUT sections correspond to the sort INPUT and OUTPUT PROCEDUREs. The MAIN-PROGRAM section contains the main line routine, the initialization routine, the report option display routines and the routines to accept selection information from the user.

The ELEM-RETRIEVAL section in the Data Element Dictionary report program provides for the retrieval of elements from the data base and includes a search for the required elements. The TRANS-RETRIEVAL section of the Transaction Dictionary and Transaction Screen Layout report programs retrieves the necessary transactions. For each data base access, the status of the data base is checked. A successful access returns an error code of zero, represented by the condition name DB-STATUS-OK. The end of set condition is represented by DB-END-OF-SET. If a data base error occurs, the error condition code contained in DB-STATUS is reported to the terminal. As the records are retrieved from the data base they are released to a sort procedure.

To use the PDP 11/70 IAS V.3 utility sorting routines the bytes within the sort key must be interchanged. The first byte becomes the last, the second byte is swapped with the next to the last, etc. The variables associated with this procedure: SORT-NAME-KEY, SORT-NUM-KEY, NAME-BYTE, NUM-BYTE, SWAP-BYTE, LOW-INDEX, HIGH-INDEX and the paragraphs: PREPARE-SORT-KEY and FLIP-BYTES are not required if the COBOL SORT verb is available. They simply are used to convert the sort key to the form acceptable to the system sort routines.

The third and final section in this program is the report output section, PRINT-OUT. This section takes the sorted records returned from the sort routine and formats them into report form with paging and appropriate headings. The COBOL AFTER ADVANCING clause is used to provide carriage control. No data base access is required in this section as all needed information is contained in the sort records.

5.1.2.1 Data Element Dictionary - ELERPT

This program produces the Data Element Dictionary, which is a report that can be obtained subsequent to the specification of an interactive system using the CASTS Process I Specification Program. The report details the content and form of particular data elements. ELERPT gives information pertinent to the control of elements. Information about the associated prompts, error messages and help messages is found in the Transaction Dictionary.

The user has two choices regarding the content and form of the report. First, the user specifies which data elements in the data base are to be included in the report. There are four options: 1) all elements in the data base, 2) the elements whose assigned element numbers fall within a certain range, 3) those elements for which a particular proponent is responsible, and 4) the elements within an application. Second, the user chooses whether the elements are to be reported in alphabetical order by element name or in ascending numerical order by element number. These choices are entered in response to questions displayed at the terminal. An invalid entry causes the program to ask for reentry of the choice. Further, if element option two, three or four is chosen, the program prompts for the additional information necessary to select the required elements. For the element number range, the upper and lower bounds are needed. The elements corresponding to the given bounding element numbers are included in the retrieval process. Selection of elements belonging to a proponent or in an application require the user to enter the proponent or application name, respectively.

The ELEM-RETRIEVAL section retrieves the appropriate elements from the data base and prepares the input records for the sort. The DML (Data Manipulation Language) statements access the data base and make the stored information available to the COBOL program. Two types of element search are needed due to the structure of the data base. If the elements are to be reported on the basis of participation in certain applications, the search must proceed from the transaction level. This is because transactions are associated with applications and as such, the application name is stored in the transaction record. If a transaction belongs to the requested application, then the component elements can be found and included in the report. Since an element may be in more than one transaction and more than one transaction may be in an application, any duplicate elements must be removed after the sorting procedure. For elements chosen by any of the other three criteria, the simpler element search is used. The information used to select the required elements is contained within the element record, thus each element may be examined in turn. A determination of eligibility is made directly and no element is selected twice.

Element retrieval based on the transaction type of search is directed by the paragraph 310-TRANS-SEARCH. Each transaction in the data base is inspected for the application selected. It finds the first transaction and successive transactions are located iteratively until the end of set condition is encountered. As each required transaction is found, the component elements are retrieved. This is done by stepping through the list of elements set, ELEM-LIST-SET, obtaining the element names from the record ELEM-LIST-REC and then locating the element header record, ELEMENT-DEMO, using the CALC method. In addition to information stored in the ELEMENT-DEMO record itself, the report requires data from the associated records EDIT-REC and TRANS-LIST-REC. As the information is accumulated and moved to sort file records, these records are released to the sort procedure.

The routines 324-RELEASE-ELEM, 325-RELEASE-FORMAT, 326-RELEASE-DESC and 327-RELEASE-TRANS release records to the sort for each type of line in the final report; element, format, description and transaction cross reference, respectively. These routines are also used in the element type of search for element retrieval. They move the key into the sort record, move the data from the data base record to the sort record and assign a record type indicator.

Two different sort key formats are required since the report can be ordered by element name or element number. There is a name key and a number key. Each key is composed of an element name or element number and a four digit numeric suffix. The suffix ensures that the order of the different types of report lines for each element is maintained during the sorting process. A zero is assigned to the first sort record corresponding to the first line of each element i.e. the ELEM-LINE. The suffix assigned is incremented by one for each successive line within an element. When a new element is encountered the counter is reset to zero.

As each sort record is released to the sort procedure, it is assigned a record type indicator. Each of the sort records corresponds to a different type of report line. The report lines can have a slightly different form if they are the first line in a group, hence a distinction is made in the record type codes. For instance, the first line of four possible lines of element description is prefaced by the title "DESCRIPTION:". The record type indicator codes and their meanings are:

REC-TYPE	Report Line Type	Contents
0	ELEM-LINE	Element header information
1	FORMAT-LINE	Element format
2	DEFAULT-LINE	Element default value
3	EDIT-CHECK-LINE	Element edit check
4	first DESC-LINE	Element description
5	subsequent DESC-LINEs	Element description continued
6	first TRANS-LINE	Transactions containing element
7	subsequent TRANS-LINEs	Transactions containing element

To find elements depending on element number, proponent or simply all elements, the element search is controlled by the paragraph 340-ELEM-SEARCH. The element search is more straightforward than the transaction oriented search since the set of all elements in the data base is inspected without first referencing the "parent" transaction. Hence, each element is retrieved once and there are no duplicates released to the sort. The data base interaction begins at the ELEMENT-DEMO level and elements to be reported are selected on the basis of the report options selected by the user. As previously mentioned, the data base status is continually checked and the routines that release sort records are utilized by both search methods.

The report output section, 500-PRINT-OUT, formats the sorted records into the final report form. If the user requested elements within a particular application, duplicate elements are removed.

The paragraph 501-SELECT-RECORDS provides the top level control of processing in the report preparation section. Each sorted record is returned from the sort file in a sequential manner until the end of file is reached. Any duplicate records should appear in a cluster. If the application report option is in effect, unique records are selected by 520-SELECT-AND-WRITE. This routine compares the current element name and numeric suffix to the previous one, and if they are identical the current record is skipped. If no element selectivity is required the selection routine is bypassed and 530-WRITE-REPORT-LINE directly writes the report line. The record type is determined and the corresponding report line format is used for the information. A running total of the number of elements is kept and printed at the end of the report. A line count is kept for the purpose of paging. Headings are printed at the beginning of a new page. If an element continues on a new page, the element name is printed again for ease of reading.

The final Data Element Dictionary is contained in the file "ELERPT.TMP". A copy of the report is obtained by printing this file.

5.1.2.2 Transaction Dictionary - TRARPT

The Transaction Dictionary is a report that can be obtained subsequent to the specification of an interactive system using the CASTS Process I Specification Program. The report contains transaction information, the constituent elements and their associated prompts, error messages and help messages.

In the Transaction Dictionary program the user makes just one choice regarding the content of the report. There are two options, either 1) all transactions in the data base, or 2) those transactions in a particular application. If the second option is chosen, the user is prompted for the application name. The transactions are reported in alphabetical order by transaction name. The elements within each transaction are printed in the order in which they appear in the transaction. This preserves the positional relationships among the elements.

The TRANS-RETRIEVAL section retrieves the transactions and associated records from the data base using the DML and prepares input records for the sort. In contrast to ELERPT, TRARPT requires only one type of search. Since the report deals with transactions, the search must proceed from the transaction level. Thus, an element may appear on the report more than one time if it is used in more than one transaction. If only those transactions within a particular application are requested, then each transaction is inspected to determine its application name. The transactions whose application names match the given application are released to the sort, along with the associated records.

The paragraph 310-TRANS-SEARCH in the TRANS-RETRIEVAL section directs the search for records. It is essentially the same as the transaction type of element search in ELERPT, except that information is obtained from the transaction header record, TRANS-REC, as well as from subordinate records. The first transaction is located and iteratively successive transactions are found until the end of set condition is encountered. Thus, each TRANS-REC in the data base is retrieved, and if the application option is in effect, those with the appropriate application name are selected. Otherwise all transactions are selected. As each required transaction is found, the component elements are retrieved. This is done by stepping through the list of elements set, ELEM-LIST-SET, obtaining the element names from the record ELEM-LIST-REC and then locating the element header record, ELEMENT-DEMO, using the CALC method. Information is required from the following records:

TRANS-REC, ELEMENT-DEMO, PROMPT-REC, ERROR-REC, and HELP-MSG-REC. The data is moved to sort file records, which are then released to the sort procedure. The routines to release records to the sort are: 324-RELEASE-TRAN, 325-RELEASE-ELEM, 326-RELEASE-PROMPT, 328-RELEASE-ERROR, and 330-RELEASE-HELP.

The routines that release records to the sort procedure move the sort key into the sort record, move the data from the data base record to the sort record and assign a record type indicator. The key consists of the transaction name and a four digit numeric suffix. There is only one key format, as opposed to the two types found in ELERPT. The suffix maintains the order of lines within a transaction because it is assigned in increments of one, beginning with zero for each transaction header line, TRAN-LINE.

Type indicators are also assigned to each sort record before release to the sort procedure, as in ELERPT. Each of the sort records corresponds to a different type of report line. The first line in a group, such as help lines, may be different than the subsequent lines, hence there are different record type codes.

REC-TYPE	Report Line Type	Contents
0	TRAN-LINE	Transaction header information
1	ELEM-LINE	Element name and number
2	First PROMPT-LINE	Prompt for element
3	Subsequent PROMPT-LINEs	Prompt for element
4	First ERROR-LINE	Error message for element
5	Subsequent ERROR-LINEs	Error message for element
6	First HELP-LINE	Help message for element
7	Subsequent HELP-LINE	Help message for element

The final section of the Transaction Dictionary program is 500-PRINT-OUT, which is the report output section. The control of program execution in the report section is handled at the highest level by the paragraph 501-SELECT-RECORDS. Each sorted record is returned from the sort file in a sequential manner until the end of file is reached. Unlike the ELERPT report section, there is no selection routine required to eliminate duplicate records. As each line is prepared, the record type is determined and the corresponding report line format is used.

The report in final form is contained in the file "TRARPT.TMP". A copy of the Transcation Dictionary can be obtained by printing this file.

5.1.2.3 Transaction Screen Layout - SCRRPT

Transaction Screen Layout is one of the three reports that can be obtained subsequent to the CASTS Process I specification of an interactive system. It provides a hard copy version of the screen layout of various transactions. This representation of the screen assists the system designer in visualizing the user input process in the screen data input mode. It can also be used to document the system design.

The user may choose one of: 1) all transactions in the data base, 2) those transactions within an application, or 3) one particular transaction. The second option is followed by a prompt for the application name. The third option causes a request for the individual transaction name. The transactions are reported in alphabetical order by transactior name, one screen layout per output page.

The TRANS-RETRIEVAL section performs the retrieval of transactions and associated records from the data base using the DML and prepares input records for the sort. As in TRARPT, one type of search is required, proceeding from the transaction level. If only those transactions within a particular application are requested, then each transaction is inspected to determine its application name. If a particular transaction is requested, that one transaction is retrieved directly rather than inspecting all transactions and checking for the correct one.

The search for records in the data base is directed by the paragraph 310-TRANS-SEARCH in the TRANS-RETRIEVAL section. It is similar to the search procedure used in TRARPT. Information is obtained from the transaction header record, TRANS-REC, and from subordinate records. Unless just one transaction is requested by the user, the first transaction is located and successive transactions are found iteratively until the end of set condition is encountered. Each TRANS-REC is retrieved in turn. If the application option is in effect, those with the given application name are selected, otherwise all transactions are selected. Note that transactions may satisfy the retrieval criteria yet have no screen layout specified.

The retrieval of a single transaction is handled by the paragraph 340-NAME-SEARCH. It uses the transaction name given by the user in response to the prompt following the transaction selection choice. If there is no transaction by the given name, the message "INVALID TRANSACTION NAME" is displayed and the program ends.

As each required transaction is found, the information from the component elements is retrieved. Screen placement data for each element is stored in the TRANS-ELE-DEFN record. There is one TRANS-ELE-DEFN for each element within a transaction and it is associated with that element implicitly by record position within two sets. There is a one-to-one correspondence between the records of TRANS-DEFN-SET and those of ELEM-LIST-SET. For each record of ELEM-LIST-SET, a record from TRANS-DEFN-SET is located, and using the element name from the ELEM-LIST-REC, the element header record, ELEMENT-DEMO is found.

The records from which information is required for the report are: TRANS-REC, TRANS-ELE-DEFN and ELEMENT-DEMO. The pertinent data is moved to sort file records, which are then released to the sort procedure. The records for each transaction are retained until processing of the transaction is complete. This is necessary for the process of placing element prompts and fields into the records that will represent the contents of each screen line to be displayed; directed by the paragraph PLACE-ELEM-ON-SCREEN. Upon completion of each transaction, the records are released to the sort procedure by the paragraphs RELEASE-RECORDS and RELEASE-SCREEN.

The routines that release records to the sort procedure move the sort key into the sort record, move the data from the data base record to the sort record and assign a record type indicator. The key consists of the transaction name and a four digit numeric suffix. There is only one key format, as opposed to the two types found in ELERPT. The suffix maintains the order of lines within a transaction because it is assigned in increments of one, beginning with zero for the first screen line of each transaction, SCREEN-LINE.

Record type indicators are assigned to each sort record prior to release to the sort procedure. When returned from the sort, this indicator allows proper interpretation of the record and its subdivision into fields. Each of the sort records corresponds to a different type of report line. The first line of the screen must be distinguished for report page format purposes. The record type codes are:

REC-TYPE	Report Line Type	Contents
0	First SCREEN-LINE	Screen display line
1	Subsequent SCREEN-LINEs	Screen display line
2	APPL-LINE	Application name

There are several assumptions and conventions within SCRRPT and in the report programs in general. All of the report programs expect numbers to be stored in the data base so as to have in place any leading zeroes required to right justify a number within its field. The SCRRPT in particular assumes that no screen prompts have been placed outside the user interaction area. Within SCRRPT, the variable CHAR-INDEX is used to count the characters that have been written. X keeps the horizontal place of the character in a line; it is a column. Y represents a row i.e. vertical placement on the screen. A screen is assumed to be 24 lines by 80 characters by the entire CASTS system. Screen area types in SCRRPT are represented by the following condition values:

AREA-TYPE	Area Name	Condition Name
1	User interaction area	USER
3	Error message area	ERR

The size of the areas on the screen may be determined in the specification process, however they must appear in the order shown. A transaction that has no screen definition specified is distinguished by the value 99 appearing in the ENTRY-START-LINE of the TRANS-ELE-DEFN.

The report output section of the Transaction Screen Layout program is 500-PRINT-OUT.

Sorted records are returned sequentially from the sort procedure until the end of file is reached. As each report line is prepared, the record type is determined and the corresponding report line format is used.

The report in final form is contained in the file "SCRRPT.TMP." A copy of the Transaction Screen Layout can be obtained by printing this file.

5.2 PROCESS II PROGRAM DESCRIPTIONS

5.2.1 Process II Simulation - CASTS2

The CASTS Process II software executes the transactions specified in Process I. Process II does this by querying the user for the transaction he wishes to execute. After receiving the transaction name from the user, Process II loads into memory the named transaction and elements listed therein. The prompts (if in the PROMPT mode) for each element in the transaction are executed and the data entered by the user. The data are checked and verified as entered. If the data are incorrect the user will be asked to re-enter the field. After it is determined that the data are valid and an entire transaction record (consisting of one each of n elements) is entered, the Process II will enter the record in the transaction file. Thus all elements of the transaction must be complete and valid before they are stored. When the user has entered all desired records into the transaction he enters an escape sequence, "@", to terminate the transaction. Upon termination the user is prompted for another transaction. He may end the application by typing "END". The session will end and a terminal summary report file will be generated. Terminal dialogues will be logged at the user's request.

Refer to Figure 4 for an overview of the Process II module configuration. A brief description of each module follows the overview figure.

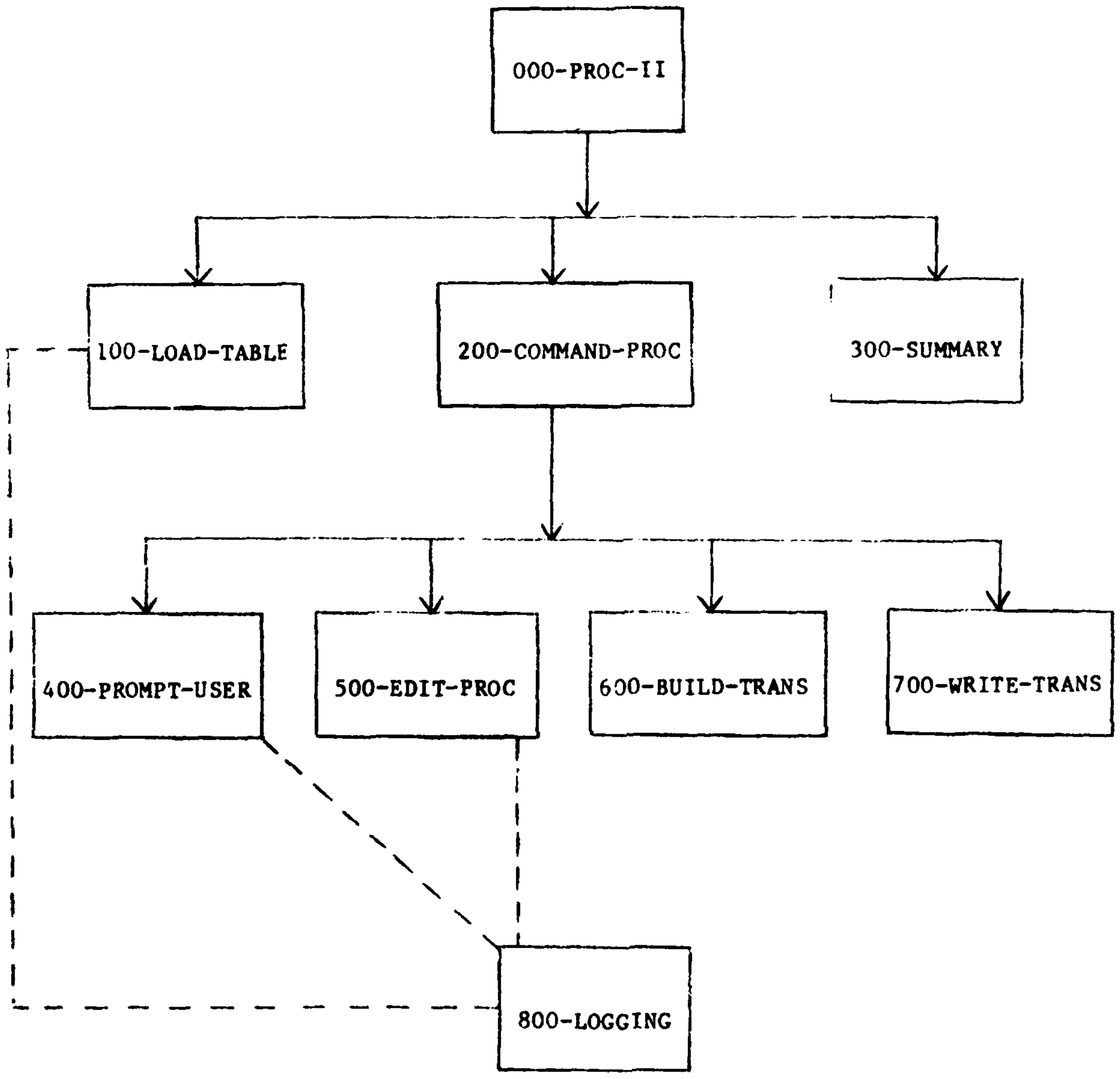


Figure 4

PROCESS II DESIGN OVERVIEW

000-PROC-II. This is the main module which controls Process II. The user is queried for user ID, and I/O mode.

Modules called:

100-LOAD-TABLES

200-COMMAND-PROC

300-SUMMARY

100-LOAD-TABLES. This module asks the user which transaction he wishes to execute. Upon entry of transaction name all specifications relevant to the transaction are loaded into the program. If the user enters END for a transaction name, the session will terminate when control returns to 000-PROC-II.

200-COMMAND-PROC. This module executes the transaction specification retrieved by 100-LOAD-TABLE keeping track of I/O mode and element numbers.

Modules called:

400-PROMPT-USER

500-EDIT-PROC

600-BUILD-TRANS

700-WRITE-TRANS

Control returns to 000-PROC-II when an escape sequence, "@", is entered signifying a new transaction is to be entered.

300-SUMMARY. This module is called by 000-PROC-II to write the Terminal Session Summary report upon termination of the application END command).

400-PROMPT-USER. This module receives the element number, mode switch and I/O port from 200-COMMAND-PROC and returns the escape switch and element data.

Modules called:

410-PROMPT-DIRECT

420-PROMPT-BATCH

430-PROMPT-PROMPT

440-PROMPT-SCEEN

410-PROMPT-DIRECT. This module is called by 400-PROMPT-USER when in the DIRECT MODE and not prompting because of error detection. The user is prompted to input the transaction record.

420-PROMPT-BATCH. This module is called by 400-PROMPT-USER when in the BATCH MODE. The routine accepts a transaction record from the input device.

430-PROMPT-PROMPT. This module is called by 400-PROMPT-USER when in the PROMPT MODE or an error was detected while in the DIRECT MODE. The user is prompted (and reprompted) to enter element data.

440-PROMPT-SCREEN. This routine is called by 400-PROMPT-USER when in the SCREEN MODE. The routine will call a COBOL subroutine which will display all prompts for a transaction record on the screen in a specified configuration.

500-EDIT-PROC. This routine is called by 200-COMMAND-PROC to validate the element data entered in 400-PROMPT-USER. Validation is accomplished by using the edit criteria specified for each element in Process I. If an error is detected, a data condition switch is set.

Modules called:

510-DISPLAY-PROC.

510-DISPLAY-PROC. This module will display the element error message (if any) and the generic error message associated with each data field.

600-BUILD-TRANS. This routine is called by 200-COMMND-PROC to build a trans-record-buffer of transaction records elements.

700-WRITE-TRANS. This routine is called by 200-COMMAND-PROC to write the entire trans-record-buffer to the disk file of the transaction.

800-LOGGING. This routine is called from various modules whenever there is dialogue between user and screen and the logging option is specified.

5.2.1.1 Terminal Session Summary Report

This report is a summary of the user interaction with the simulation process during a single terminal session. It is short, providing a fixed set of statistical data. It shows the transactions used during the session, the user id with his time in/out, the number of transactions for which data has been entered, time spent per transaction and the number of elements, prompts and error messages used or issued during the session. The report is formatted into the file <user id>.SRF.

5.2.1.2 Simulation Transaction Log

The transaction log of the user terminal session is labeled by the user id and the extention ".TLF". The file consists of sequential records. The records may be one of four types:

01 - application
02 - transaction
03 - terminal text
04 - error

When a new application is started a "01" record is written. A new transaction causes a "02" record to be written. After these two records a sequence of "03" and "04" records will follow until a new transaction is entered, which will restart the sequence.

The application (01) record contains:

1. time stamp
2. type marking
3. user id
4. application name
5. session I/O mode

The transaction (02) record contains:

1. time stamp
2. type marking
3. transaction name
4. transaction length

The terminal text (03) record contains:

1. time stamp
2. type marking
3. interaction type (prompt, help, etc.)

4. element name
5. text line from terminal

The error (04) record contains:

1. time stamp
2. type marking
3. error type
4. element name
5. bad data

The session log may be formatted into a report by running LOGRPT and then printing the file <user id>.LRF.

5.2.2 Terminal Session Error Report - ERRRPT

The Terminal Session Error Report provides a listing of errors made during the CASTS Process II simulation of an interactive system. The field edit errors are extracted from the simulation interaction log file, which records all simulation/user interactions during Process II.

The program, named ERRRPT, is written in COBOL. The program is intended to run from a terminal and any error messages are written to the terminal. ERRRPT utilizes three files in the course of producing the report. The simulation transaction log file, <user id>.TLF, is the file in which the error information is found. SORTFIL.DAT, as in the Process I reports, is a temporary file used for sorting purposes. The final formatted output report file is <user id>.ERF. The program begins with variable initializations, selects the appropriate data from the simulation transaction log file, releases records to the sorting process, sorts the records, and formats the information into report output lines.

The program is organized in a hierarchical modular form, with a main line routine that has high level control over execution of the modules. There are three COBOL sections, MAIN-PROGRAM, ERROR-RETRIEVAL and PRINT-OUT. The ERROR-RETRIEVAL and PRINT-OUT sections correspond to the INPUT and OUTPUT PROCEDUREs of the COBOL sort verb. Currently, external system routines are called to provide the sorting functions. The sectional structure of the program is set up to facilitate program conversion to an environment in which the COBOL internal SORT, RELEASE and RETURN verbs can be used.

The MAIN-PROGRAM section contains the main line routine and the initialization routine. The initialization routine queries the user for <user id>, opens files, obtains the date from the system, resets counters and identifies the program by displaying a message on the terminal. The main line routine directs the processing at a high level. It calls for the initialization to be performed, followed by the error data retrieval, the sorting process, the report formatting and finally ends the program.

The ERROR-RETRIEVAL section extracts the error-related information from the simulation interaction log file. The data is stored temporarily in sort records and released to the sort. In order to use the PDP 11/70 IAS V.3 utility sorting routines, the bytes within the sort key must be interchanged. The first byte becomes the last, the second type is swapped with the penultimate, etc. The variables associated with this procedure: KEY-FOR-SORT, KEY-BYTE, SWAP-BYTE, LOW-INDEX, HIGH-INDEX and the paragraphs: 202-PREPARE-SORT-KEY and 203-FLIP-BYTES are not required if the COBOL SORT verb is available. They simply are used to convert the sort key to the form acceptable to the system sort routines.

The selection of the information required from the simulation log file for the error report is directed at a high level by 201-RETRIEVE-ERRORS. The simulation log file is read sequentially. The first record is the application header. As each subsequent record is read, it is checked for record type and if it is a transaction header, the new transaction name and time stamp are recorded. When error records are encountered, error processing begins. The variable, CHECK-REC-TYPE is used to determine the appropriate records.

CHECK-REC-TYPE	
APPL-HDR	01
TRAN-HDR	02
IO-REC	03
ERR-REC	04

The routine that releases records to the sort procedure moves the sort key into the sort record and assigns a sort record type indicator. The key consists of the transaction name, the transaction header time stamp and a four digit suffix. The suffix maintains the order of report lines within a transaction and is assigned in increments of one, beginning with zero for each transaction header found in the simulation log file.

Record type indicators are assigned to each sort record prior to release to the sort procedure. Sort record types are distinct from simulation log record types. When records are returned from the sort, the indicator allows proper interpretation of the record and its formatted fields. Each of the sort records corresponds to a different type of report line. The sort record type codes are:

RECTYPE	REPORT LINE TYPE	CONTENTS
E	ELEM-LINE	Element name
R	ERROR-LINE	Field edit error, bad data

The report output section of the Terminal Session Error Report program is 400-PRINT-OUT. The sorted records are returned from the sort routine and formatted into report form with paging and appropriate headings. Carriage control is provided by the COBOL AFTER ADVANCING clause.

The high level paragraph within the report section is 401-SELECT-RECORDS. Sorted records are returned sequentially from the sort procedure until the end of file is reached. As each report line is prepared, the record type is determined and the corresponding report line format is used. A running total of the number of errors appearing on the report is kept and printed at the end of the report. Headings are printed automatically at the beginning of a new page.

The report in final form is contained in the file "<user id>.ERF". A copy of the Terminal Session Error Report can be obtained by printing this file.

5.2.3 Interaction Log Report - LOGRPT

LOGRPT is run after a terminal session to build a report file from the user's log file. The user must have requested logging in Process II for the log file to exist. LOGRPT is controlled by a main section paragraph. This paragraph performs calls to read and write routines until the end of the log file is reached.

The files are labeled uniquely for each user. The user is asked to enter the ID of the user whose log file will be processed into a report. The program then uses the <user id> and adds the proper extentions, i.e.:

.TLF	- transaction log file
.LRF	- log report file

If the program encounters errors in opening the files an error message will be output and processing will terminate. The program also checks for sequence. An "02" record (transaction) must follow every "01" record (application). If this sequence is out of order, processing will terminate.

The user may print the report file by entering, PRINT <user id>.LRF.

5.2.4 Application Raw Data Report - DATRPT

DATRPT is run after a terminal session to build a raw data report from the user's data file, <user id>.TDF. DATRPT is controlled by a main section paragraph. This paragraph performs calls to read and write routines until the end of the data file is reached.

The files are labeled uniquely for each user. The user is asked to enter the ID of the user whose data file will be processed into a report. The program then uses the <user id> and adds the proper extentions, i.e.:

.TDF	-	transaction data file
.DRF	-	data report file

If the program encounters errors in opening the files an error message will be output and processing will terminate.

The user may print the report file by entering, PRINT <user id>.DRF.

5.3 HELP MESSAGES

5.3.1 Help Message Text Editor - HELPED

The Help Message Text Editor is the means by which help message text may be created, stored in the data base, changed, copied or deleted. These help messages are system help messages and should not be confused with user specified help messages. The program is written in COBOL and utilizes the TOPIC-AREA of the data base. It is expected that this program will be primarily used by a CASTS System Administrator, along with the Help Messages Report program. It is not intended for use by the end user of CASTS, i.e., the system designer and his/her system test personnel, thus it is not mentioned in the CASTS User's Manual.

The program, called HELPED, uses three files. Both an input file TERM-IN and an output file TERM-OUT are assigned to the standard device, the terminal. A temporary file, TEMP.DAT, is used in the process of writing the help messages and storing them in the data base. The TOPIC-AREA of the data base contains all of the help messages for both Process I and Process II. The TOPIC-KEY consists of an application name and a topic name. Process I and Process II each have unique application names, stored as value constants in the variables PROC1-ID and PROC2-ID, respectively. The topic name is the name of the subject of the help message.

The program, similar to other editor programs, responds to commands from the user. The commands consist of one letter, indicating a function to be performed on the text of a help message. The commands are:

C	Create a new help message, enter text edit mode
T	Text edit an already existing help message
D	Delete a help message
R	Reproduce an existing help message with a new application name and topic name, i.e., copy
E	End the session

The paragraph 001-MAIN-PARAGRAPH is the command processor and performs the paragraph corresponding to the given command.

To create a help message, a new TOPIC-LINES-SET is created. Likewise, to delete a help message, an entire set is deleted. To reproduce a help message a new set is created and the INFO-LINE-RECs from the old help are copied into the new help.

A help message is changed and/or displayed by using the text edit command. In the text edit mode there are commands to manipulated the help message text. The commands are:

(1) C<delimiter><string 1><delimiter><string 2><delimiter>

Change all occurrences of the first string of text to the second. The delimiter is any printable character, including a space. If the second string is absent or null, the old string will be replaced with an empty string.

(2) P<space><start line number>[<space><last line number>]

Print the line(s) of text indicated by the parameter(s). If only one line is desired the options in braces are not necessary.

(3) I<space><line number>

Insert will create a new line to be placed after the line number given. It will continue to accept input lines, inserting them sequentially after each previous line. Input is terminated by typing a CNTL Z as the first character in a line of input.

(4) D<space><line number>

This deletes the line specified by <line number>. Do not try to use a range of lines for this command.

(5) +<line count>

Move forward the number of lines given by <line count>.

(6) -<line number>

Move backward the number of lines given by <line count>.

(7) E

Exit text edit mode.

A limited help facility is provided for listing the available commands with a brief description of their function.

5.3.2 Help Messages Report - HLPRPT

The Help Messages Report program provides a complete list of the system help messages that are stored in the TOPIC-AREA of the data base. It can be used to obtain a written copy of the system help messages so that the CASTS System Administrator may update the helps as necessary.

The program is written in COBOL and utilizes just one file; HLPRPT.DAT. This file contains the completed report following the program run. The program is simple and short. No sorting is performed, hence the help messages are reported in the order in which they are retrieved from the data base.

The program control resides in 000-MAIN-PARAGRAPH. It prepares the data base for retrieval of the TOPIC-RECs and INFO-LINE-RECs, writes the report header, retrieves and writes each help topic until none remain and then closes the report file and the data base. Searching through all of the records in the TOPIC-AREA takes quite some time. When finished, the report can be obtained by printing HLPRPT.DAT. This file should be deleted, after it is printed, to release disk space.

6.0 SUGGESTIONS FOR ENHANCEMENTS

6.0 SUGGESTIONS FOR ENHANCEMENTS

The CASTS system as specified under contract DAAK70-79-D-0087 is a viable pilot system for testing the adequacy of functional specifications. CASTS allows the user to design, build and simulate transactions with the aid of the computer. An enhancement to CASTS to analyze the simulations would make the system complete. The transaction designer would then have before him the necessary statistics to make changes to his design.

Other suggestions for future enhancements which would increase the functionality of CASTS are listed below:

1. Include a facility for batch mode input to Process I such that existing data bases could be placed in the schema arrangement for use with Process II.

2. In Process II, add the capability to enter an entire transaction on a screen before the data checking for individual elements is performed. This would allow the user faster access similar to the DIRECT MODE but with a screen template.

3. Allow numeric data to be entered in a free format similar to floating point.
4. Add the use of color to the SCREEN MODE to enhance the man-machine interface.

These enhancements should increase the productivity and the use of CASTS.

7.0 COMMENTS ON CONVERSION TO IBM SYSTEMS

7.0 COMMENTS ON CONVERSION TO IBM SYSTEMS

The CASTS system is written in ANSI 74 COBOL to ensure that the system could be ported to any computer installation supporting the ANSI 74 COBOL. Although the majority of the software conforms to this standard, there are several features of the development system, a PDP-11/70, that require consideration in performing the software implementation on a different computer system. These are listed below with a commentary to reduce the possible problems that might be encountered during the conversion process.

(1) CONFIGURATION SECTION.

The physical environment is defined by the source and object computer COBOL clauses. The SELECT statement contains the physical name of the unit to which the input/output operation is to be performed. This unit name can be a device or mass storage file name. The format appears in the programs as follows:

```
SELECT DUMMY-FILE ASSIGN TO "DUMMY.DAT."

                    or

SELECT PRINT-OUT ASSIGN TO "TI:."
```

(2) SORT VERB.

The PDP-11/70 IAS version of COBOL does not implement the SORT verb. The system sort facility is accessed through CALL statements to the sort subroutines. Unlike the versions of COBOL that have the SORT verb, PDP COBOL does not have the capability of INPUT-SECTIONS or OUTPUT-SECTIONS. Programs that use the sort subroutines are written in sections that correspond directly to INPUT AND OUTPUT sections. Details of individual program conversion to use an internal COBOL SORT are mentioned in the program descriptions (Section 5.1.2 and 5.2.2 of this document).

(3) TERMINAL INPUT/OUTPUT.

The I/O to interactive terminals in the IAS environment for CASTS is performed using the ACCEPT and DISPLAY verbs where possible. In IAS the end-of-file switch is not set on the reception of a blank line or a carriage return, but only when a CONTROL Z has been typed at the terminal. The programs test for the end-of-file condition by specifically checking for blank lines.

In CASTS1, the nonstandard DEC COBOL DISPLAY verb WITH NO ADVANCING option was used. It is isolated in a single COBOL paragraph for ease in conversion.

The cursor control in CASTS2 was written for a Digital Equipment VT-52 terminal and Hazeltine 1500. It is performed by COBOL subroutines that can easily be replaced by an in-house facility. FORTRAN subroutines were written that also perform the same function and they are included in the software package.

(4) DBMS VERSIONS AND REQUIREMENTS.

CASTS was developed under IAS PDS Version 3 with DBMS-11 V1.0 on a PDP-11/70 with 256K. The programs utilize the DML verbs supported by this version of DBMS-11 and require that the status and return flags from the DBMS conform to this version. During porting the differences between the target DBMS and DBMS-11 should be examined to determined if this will affect the operation of the CASTS system.

(5) TIME AND DATA REGISTERS.

Some of the CASTS programs request the date and time from the operating system. Under IAS these formats are:

Date - YYMMDD

Time - HHMMSS.ss (24 hour clock)

(6) OVERLAYS OF PROGRAMS.

Since the PDP allows only a 32K words (64K bytes) of work space for user programs, some of the programs have been overlayed to fit this region. Overlays can be removed if the target host system has plentiful main memory or does not have a region restriction. This will result in faster response time from the interactive portions of the CASTS system.

Overlays in CASTS1 are accomplished via the COBOL segmentation facility. CASTS2 overlays are handled external to the source code by using the overlay capability of the DEC COBOL compiler.

8.0 TESTING PROCEDURES

8.1 TESTING OF PROCESS I

Complete testing of Process I required a substantial test data base of test transactions and elements as well as considerable exercising of the various commands. Both legal and illegal input was used. The testing procedure consisted of testing each command one at a time. The command under inspection was exercised in such a way as to test the various conditions prossible when the command is invoked. Each valid form of each command was also tested.

The following example of the use of Process I attempts to demonstrate that the program works properly. However, it is not an exhaustive testing session since such a session would quickly become voluminous. Prior to the session shown several elements and transactions existed. The information contained in the data base is shown before the demonstration run, then the Process I run is included, followed by the data base after the run. The test data represents a fictional payroll and parts inventory system.

TEST DATA BEFORE DEMONSTATION OF PROCESS I

<u>Data Base Relationships</u>

Transaction:	EMP-INFO
Elements:	EMP-NUM FIRST-NAME PAY-RATE MGR

<u>Transaction</u>

Name:	EMP-INFO
Length:	43

Application:	PAYROLL-APPL
Entry Start Line:	99
Entry End Line:	-
Message Start Line:	-
Message End Line:	-

Elements

Name:	DEPT-NUM	EMP-NUM
Number:	000100	000101
Length:	3	9
Description:	DEPARTMENT NUMBER	EMPLOYEE SSN
Proponent:	PERSONNEL	PERSONNEL
Constant:	N	N
Format:	XXX	999999999
Default	AAC	000000000
Prompt:	ENTER DEPARTMENT #: (XXX)	ENTER EMPLOYEE NUMBER (SAME AS SSN)
Error Message:	MUST BE 3 CHARACTERS	MUST BE OF THE FORM 9999999999
Help Message:	THE DEPT NUM CONSISTS OF 3 ALPHANUMERICS	THE EMPLOYEE # IS HIS/HER SSN

Name:	LAST-NAME	FIRST-NAME
Number:	000102	000103
Length:	20	10
Description:	EMPLOYEE SURNAME	EMPLOYEE FIRST NAME
Proponent:	PERSONNEL	PERSONNEL
Constant:	N	N
Format:	AAAAAAAAAAAAAAAAAAAAA	AAAAAAAAA

Default:	ƀ	ƀ
Prompt:	LAST NAME?	FIRST NAME?
Error Message:	NOT ALPHA	NOT ALPHA
Help Message:	LAST NAME CONSISTS OF ALPHABETICS	FIRST NAME CONSISTS OF ALPHABETICS

Name:	PAY-RATE	MGR
Number:	000104	000105
Length:	4	20
Description:	RATE OF PAY/HOUR	DEPT MANAGER
Proponent:	PERSONNEL	PERSONNEL
Constant:	N	N
Format:	Z999	AAAAAAAAAAAAAAAAAAAA
Default:	0350	ƀ
Prompt:	PAY RATE PER HOUR	WHO IS THE MANAGER?
Error Message:	YOU DIDN'T USE 4 DIGITS	YOU BLEW IT
Help Message:	FORMAT IS Z999	20 ALPHABETIC CHARACTERS ALLOWED

Screen Definitions

Transaction:	EMP-INFO	FIRST-NAME	PAY-RATE
Element:	EMP-NUM	FIRST-NAME	PAY-RATE
Start Position:	1	10	30
Prompt X:	-	-	-
Prompt Y:	-	-	-
Prompt Contents:	-	-	-
Data Field X:	-	-	-
Data Field Y:	-	-	-

Transaction:	EMP-INFO
Element:	MGR
Start Position:	34
Prompt X:	-
Prompt Y:	-
Prompt Contents:	-
Data Field X:	-
Data Field Y:	-

```
PDS> RUN CASTSI
15:35:11

******     COMPUTER AIDED SPECIFICATION TESTING SYSTEM     ******
                      PROCESS I SPECIFICATION
                          AIRMICS, GIT


WELCOME TO CASTS!
IF YOU WOULD LIKE HELP, PLEASE TYPE:
        HELP/CASTS -        FOR A DESCRIPTION OF THE CASTS SYSTEM
        HELP/HELP -         FOR INFORMATTON ON THE 'HELP' FACILITY
        HELP/COMMANDS -     FOR A LIST OF AVAILABLE COMMANDS
        HELP/START -        HOW TO GET STARTED IN PROCESS I


FIRST, PLEASE ENTER YOUR ID (UP TO 7 CHARACTERS)?
DH
IS THIS CORRECT:  DH          TYPE 'Y' OR 'N'
Y


COMMAND> HELP/CASTS

        CASTS IS A TOOL FOR USE IN THE DEVELOPMENT OF INTERACTIVE
DATA ENTRY SYSTEMS.  THE CASTS SYSTEM ALLOWS A USER TO SPECIFY
AND THEN TEST INTERACTIVE DIALOGUES.

        CASTS IS COMPOSED OF TWO PROCESSES.  THE PROCESS I SPECIFICA-
TION ENABLES A SYSTEM DESIGNER TO ENTER SPECIFICATIONS TO BUILD A
MODEL OF ELEMENTS AND TRANSACTIONS.  THE PROCESS II SIMULATION TESTS
THE SPECIFICATIONS GIVEN BY SIMULATING THE INTERACTIVE DIALOGUE.

        THE ADVANTAGE OF CASTS IS THE ABILITY TO TEST THE
MODEL (PROCESS II) AND TO MODIFY THE SPECIFICATIONS (PROCESS I)
REPEATEDLY IN RESPONSE TO PROBLEMS FOUND.  NO CODING IS REQUIRED
DURING THIS CYCLE.


FOR MORE INFORMATION, TYPE:
        HELP/PROCESS I              HELP/ELEMENT
        HELP/PROCESS II             HELP/COMMANDS
        HELP/TRANSACTION            HELP/HELP


COMMAND> HELP/HELP

        THE 'HELP' FACILITY FOR CASTS CONSISTS OF TWO METHODS OF
OBTAINING ADDITIONAL INFORMATTON.  IN RESPONSE TO THE PROMPT
'COMMAND>' YOU MAY ENTER THE HELP COMMAND FOLLOWED BY THE TOPIC
ON WHICH YOU ARE REQUESTING INFORMATION.  THE SYNTAX IS:
                HELP/<NAME OF TOPIC>
```

```
          THE OTHER TYPE OF HELP IS FOR USE WHILE ENTERING SPECIFI-
CATIONS IN PROCESS I.  IN RESPONSE TO ANY PROMPT YOU MAY TYPE A '?'
FOR AN EXPLANATION OF THE INFORMATION TO BE ENTERED.  E.G.
                    ENTER ELEMENT NAME   .
                    ?
OR TO OBTAIN HELP ON OTHER SUBJECTS:
                    ?/<NAME OF TOPIC>

          AT THE END OF EACH HELP MESSAGE ARE SUGGESTIONS
FOR OBTAINING FURTHER RELATED INFORMATION.


RELATED TOPICS:
          TOPIC


COMMAND> HELP/START

          TO GET STARTED USING THE CASTS PROCESS I YOU WILL WANT TO
CREATE ELEMENTS AND BUILD TRANSACTIONS.  ELEMENTS SHOULD EXIST BEFORE
THEY ARE BE BUILT INTO TRANSACTIONS.  USE THE 'CREATE' COMMAND TO
CREATE AN ELEMENT AND YOU WILL BE PROMPTED FOR INFORMATION ABOUT
THE ELEMENT, THE PROMPT YOU WANT FOR IT AND ANY EDIT CRITERIA,
ERROR MESSAGES OR HELP MESSAGES YOU MAY WISH TO SPECIFY.  A BLANK
LINE TERMINATES THE INPUT.

          USE THE 'BUILD' COMMAND TO BUILD TRANSACTIONS.  YOU WILL
BE ASKED TO ENTER INFORMATION ABOUT THE TRANSACTIONS AND TO LIST
THE ELEMENTS TO BE INCLUDED.  EXPERIENCED USERS MAY OPTIONALLY
SPECIFY INFORMATION RELATING TO A SCREEN MODE OF INPUT FOR USE
IN PROCESS II.

          THE 'DISPLAY' COMMAND ALLOWS YOU TO CHECK THE SPECIFICATIONS
ENTERED AND THE 'MODIFY' COMMAND MAKES IT POSSIBLE TO CHANGE THEM.
USE THE 'HELP' COMMAND OR '?' ANYTIME YOU ARE UNCERTAIN AS TO WHAT
TO ENTER.


COMMAND> HELP/COMMANDS

THE AVAILABLE COMMANDS ARE:
          BUILD
          CREATE
          DELETE
          DISPLAY
          [illegible]
          [illegible]
          [illegible]
```

```
COMMAND> CREATE
ENTER ELEMENT NAME
PART-NUM

PLEASE ENTER ELEMENT HEADER INFORMATION:
ENTER ELEMENT DESCRIPTION (4 LINES)
PART NUMBER

ENTER PROPONENT
INVENTORY
ENTER ELEMENT NUMBER
000001
ENTER CONSTANT FIELD INDICATOR
?

CFD
      THE CONSTANT FIELD INDICATOR (CFD) ALLOWS THE USER TO SPECIFY
THAT A CERTAIN ELEMENT IS A CONSTANT.  THE CONSTANT VALUE IS THE DEFAULT
VALUE SPECIFIED IN THE EDIT INFORMATION.  THE CONSTANT FIELD INDICATOR
MUST BE 'Y' OR 'N'.

ENTER CONSTANT FIELD INDICATOR
N
ENTER ELEMENT LENGTH
7
HEADER COMPLETE FOR ELEMENT:     PART-NUM

PLEASE ENTER PROMPT LINES:
ENTER PROMPT LINE
WHAT IS THE PART
ENTER PROMPT LINE
    NUMBER ?
ENTER PROMPT LINE


PLEASE ENTER EDIT INFORMATION:
ENTER ELEMENT FORMAT
?

FOR
      ELEMENT FORMAT (FOR) DESCRIBES THE EDIT CRITERIA BY WHICH
PROCESS II CHECKS THE ELEMENT.  THERE ARE 6 FORMAT CHARACTERS.  EACH
CHARACTER POSITION SHOULD HAVE A VALUE INDICATED BY THE FORMAT GIVEN:
        9    NUMERIC
        A    ALPHABETIC
        X    ALPHANUMERIC
        Z    ZERO SUPPRESSION
        .    DECIMAL POINT
        S    SIGN I.E. '+' OR '-' OR ' '
FOR EXAMPLE, AN ELEMENT FORMAT OF 'ZZ9.99' MEANS THAT THE ELEMENT IS
NUMERIC WITH TWO DIGITS TO THE RIGHT OF THE DECIMAL POINT AND IF LESS
THAN 100.00 THE LEADING ZEROS MAY BE REPLACED BY SPACES (  9.65
INSTEAD OF 009.65).  NO ENTRY WILL MEAN THAT NO EDITING WILL BE
PERFORMED ON THIS ELEMENT.  ANY SPECIAL CHARACTER CAUSES THAT
```

```
POSITION TO BE THAT CHARACTER.

ENTER ELEMENT FORMAT
9999999
ENTER ELEMENT DEFAULT VALUE
0000000
NUMERIC ELEMENTS MAY HAVE RANGE OR VALUE CHECKS.
RANGE CHECKS, VALUE CHECKS OR NEITHER?  (R/V/N)
N

PLEASE ENTER ANY ERROR MESSAGE LINES:
ENTER ERROR LINE
MUST BE A NUMBER
ENTER ERROR LINE
    UP TO 7 DIGITS
ENTER ERROR LINE


PLEASE ENTER ANY HELP MESSAGE LINES:
ENTER HELP LINE
FORMAT IS NUMERIC
ENTER HELP LINE

ELEMENT CREATION COMPLETE:       PART-NUM


COMMAND> DISPLAY/ELEM/PART-NUM
CURRENT ELEMENT DESCRIPTION (4 LINES)
  PART NUMBER




CURRENT PROPONENT
  INVENTORY
CURRENT ELEMENT NUMBER
  000001
CURRENT ELEMENT CREATE DATE
  810303
CURRENT ELEMENT CREATOR ID
  DH
CURRENT CONSTANT FIELD INDICATOR
  N
CURRENT ELEMENT LENGTH
  0007

CURRENT VALUE FOR PROMPT MESSAGE
  WHAT IS THE PART
      NUMBER ?

CURRENT ELEMENT FORMAT
  9999999
CURRENT ELEMENT DEFAULT VALUE
  0000000
```

```
CURRENT VALUE FOR ERROR MESSAGE
  MUST BE A NUMBER
      UP TO 7 DIGITS

CURRENT VALUE FOR HELP MESSAGE
  FORMAT IS NUMERIC


COMMAND> DU
CA001:  INVALID COMMAND


COMMAND> BUILD/DEPT-INFO
PLEASE ENTER TRANSACTION HEADER INFORMATION:
ENTER APPLICATION NAME
PARTS-APPL
DO YOU WISH TO HAND-TAILOR A SCREEN FORMAT?
    NOTE:  THIS REQUIRES SPECIAL KNOWLEDGE.
           PLEASE SEE USER MANUAL, SECTION 4.6.
SCREEN FORMAT?  (Y/N)
N
HEADER COMPLETE FOR TRANSACTION: DEPT-INFO

PLEASE ENTER THE ELEMENTS FOR THIS TRANSACTION:
    NOTE:  AN EMPTY LINE INDICATES END OF INPUT

ENTER ELEMENT NAME
DEPT-NUM

ENTER START POSITION IN TRANSACTION
1

ENTER ELEMENT NAME
MGR

ENTER START POSITION IN TRANSACTION
4

ENTER ELEMENT NAME
EMP-NUM

ENTER START POSITION IN TRANSACTION
24

ENTER ELEMENT NAME


TRANSACTION BUILD COMPLETE:      DEPT-INFO


COMMAND> BUILD
ENTER TRANSACTION NAME
PARTS-INFO
```

```
PLEASE ENTER TRANSACTION HEADER INFORMATION:
ENTER APPLICATION NAME
PARTS-APPL
DO YOU WISH TO HAND-TAILOR A SCREEN FORMAT?
    NOTE:  THIS REQUIRES SPECIAL KNOWLEDGE.
           PLEASE SEE USER MANUAL, SECTION 4.6.
SCREEN FORMAT?  (Y/N)
Y
ENTER ENTRY START LINE
1
ENTER ENTRY END LINE
19
ENTER ERROR MESSAGE START LINE
20
ENTER ERROR MESSAGE END LINE
26
CA013:  INVALID VALUE FOR THIS FIELD
REENTER
?

MEL
      THE ERROR MESSAGE END LINE (MEL) SPECIFIES THE ENDING LINE
NUMBER OF THE ERROR MESSAGE AREA ON THE CRT SCREEN, FOR USE IN THE
SCREEN MODE OF INTERACTION DURING PROCESS II SIMULATION.  IT MUST
BE GREATER THAN THE ERROR MESSAGE START LINE AND LESS THAN OR
EQUAL TO 23.

RELATED TOPICS:
      MODE                        SCREEN MODE

REENTER
23
HEADER COMPLETE FOR TRANSACTION: PARTS-INFO

PLEASE ENTER THE ELEMENTS FOR THIS TRANSACTION:
    NOTE:  AN EMPTY LINE INDICATES END OF INPUT

ENTER ELEMENT NAME
PART-NUM

ENTER START POSITION IN TRANSACTION
1
ENTER SCREEN PROMPT POSITION(X)
1
ENTER SCREEN PROMPT POSITION(Y)
2
ENTER SCREEN PROMPT CONTENTS
PART # :
ENTER DATA FIELD POSITION(X)
11
ENTER DATA FIELD POSITION(Y)
2

ENTER ELEMENT NAME
DEPT-NUM

ENTER START POSITION IN TRANSACTION
8
```

```
ENTER SCREEN PROMPT POSITION(X)
1
ENTER SCREEN PROMPT POSITION(Y)
4
ENTER SCREEN PROMPT CONTENTS
DEPT # :
ENTER DATA FIELD POSITION(X)
11
ENTER DATA FIELD POSITION(Y)
4

ENTER ELEMENT NAME
QTY
CA003:  ELEMENT BY THIS NAME DOES NOT EXIST
        IF YOU MISTYPED THE NAME, AND WISH TO
        REENTER, TYPE 'R'.  TO CREATE THE
        ELEMENT NOW, TYPE 'C'.
REENTER OR CREATE?  (R/C)
C

PLEASE ENTER ELEMENT HEADER INFORMATION:
ENTER ELEMENT DESCRIPTION (4 LINES)
QUANTITY ON HAND

ENTER PROPONENT
INVENTORY
ENTER ELEMENT NUMBER
000002
ENTER CONSTANT FIELD INDICATOR
N
ENTER ELEMENT LENGTH
4
HEADER COMPLETE FOR ELEMENT:      QTY

PLEASE ENTER PROMPT LINES:
ENTER PROMPT LINE
HOW MUCH ?
ENTER PROMPT LINE


PLEASE ENTER EDIT INFORMATION:
ENTER ELEMENT FORMAT
9999
ENTER ELEMENT DEFAULT VALUE
0000
NUMERIC ELEMENTS MAY HAVE RANGE OR VALUE CHECKS.
RANGE CHECKS, VALUE CHECKS OR NEITHER?  (R/V/N)
N

PLEASE ENTER ANY ERROR MESSAGE LINES:
ENTER ERROR LINE
NOT NUMERIC
ENTER ERROR LINE
```

```
PLEASE ENTER ANY HELP MESSAGE LINES:
ENTER HELP LINE
4 DIGIT NUMBER
ENTER HELP LINE

ELEMANT CREATION COMPLETE:        QTY

ENTER START POSITION IN TRANSACTION
11
ENTER SCREEN PROMPT POSITION(X)
1
ENTER SCREEN PROMPT POSITION(Y)
6
ENTER SCREEN PROMPT CONTENTS
QUANTITY :
ENTER DATA FIELD POSITION(X)
11
ENTER DATA FIELD POSITION(Y)
6

ENTER ELEMENT NAME


TRANSACTION BUILD COMPLETE:       PARTS-INFO


COMMAND> DISPLAY/TRANS/PARTS-INFO
CURRENT TRANSACTION LENGTH
  0014
CURRENT APPLICATION NAME
  PARTS-APPL
CURRENT ENTRY START LINE
  01
CURRENT ENTRY END LINE
  19
CURRENT ERROR MESSAGE START LINE
  20
CURRENT ERROR MESSAGE END LINE
  23
CURRENT TRANSACTION CREATE DATE
  810303
CURRENT TRANSACTION CREATOR ID
  DH
ELEMENT(S) IN THIS TRANSACTION:
                                         START  LENGTH
  PART-NUM                                 1       7
  DEPT-NUM                                 8       3
  QTY                                     11       4


COMMAND> MODIFY
ENTER ITEM NAME
ERROR(1)
ENTER ELEMENT NAME
LAST-NAME
```

```
   NOT ALPHA
ENTER ERROR LINE
NOT ALPHABETIC

MODIFICATION COMPLETE


COMMAND> DISPLAY/ERROR/LAST-NAME
CURRENT VALUE FOR ERROR MESSAGE
   NOT ALPHABETIC


COMMAND> INSERT/ERROR(1)/LAST-NAME
ENTER ERROR LINE
      MUST BE A-Z
INSERTION COMPLETE


COMMAND> INSERT/ERROR(2)/LAST-NAME
ENTER ERROR LINE
      UP TO 20 CHARACTERS
INSERTION COMPLETE


COMMAND> DISPLAY/ERROR//
CURRENT VALUE FOR ERROR MESSAGE
   NOT ALPHABETIC
         MUST BE A-Z
         UP TO 20 CHARACTERS


COMMAND> DELETE/ERROR(1)/LAST-NAME
ITEM DELETED

DELETION COMPLETE


COMMAND> DISPLAY/ERROR/LAST-NAME
CURRENT VALUE FOR ERROR MESSAGE
         MUST BE A-Z
         UP TO 20 CHARACTERS


COMMAND> HELP/MODIFY

MODIFY/<ITEM NAME>/<ELEM OR TRANS NAME>

          'MODIFY' IS A 3 FIELD COMMAND THAT ALLOWS THE USER TO CHANGE
THE SPECIFICATIONS THAT HAVE BEEN ENTERED.  THE ITEM NAMES [illegible]
'ELEM' AND 'LIST' ARE INVALID AS THEY HAVE NO MEANING [illegible]
COMMAND.  HOWEVER, 'THDR', 'SCREEN', 'EHDR', [illegible]
'ERROR' AND 'HELP' ARE VALID ITEM NAMES.
          THE PROMPT, EDIT CRITERIA, ERROR MESSAGE [illegible] MAY
BE MODIFIED BY PUTTING THE LINE NUMBER IN PARENS [illegible] THE ITEM
NAME AS THE ITEM [illegible] EXAMPLES, TO MOD[illegible] THIRD HELP LINE
```

```
                    [illegible] <ELEM NAME>


COMMAND: MODIFY/EHDR/LAST-NAME

THE OPTIONS FOR EHDR (ELEMENT HEADER) ARE:
          DES     ELEMENT DESCRIPTION
          PRO     PROPONENT
          NUM     ELEMENT NUMBER
          CFD     CONSTANT FIELD INDICATOR
          LEN     ELEMENT LENGTH
FOR MORE INFORMATION, THESE CAN ALL BE USED AS HELP TOPICS.

ENTER OPTION
DES
CURRENT ELEMENT DESCRIPTION (4 LINES)
  EMPLOYEE SURNAME



ENTER ELEMENT DESCRIPTION (4 LINES)
EMPLOYEE LAST NAME
WITH A MAXIMUM OF 20 CHARACTERS
    INCLUDING SPACES OR DASHES


MODIFICATION COMPLETE


COMMAND: DISPLAY/ELEM/LAST-NAME
CURRENT ELEMENT DESCRIPTION (4 LINES)
  EMPLOYEE LAST NAME
  WITH A MAXIMUM OF 20 CHARACTERS
      INCLUDING SPACES OR DASHES


CURRENT PROPONENT
  PERSONNEL
CURRENT ELEMENT NUMBER
  000102
CURRENT ELEMENT CREATE DATE
  [illegible]
CURRENT ELEMENT CREATOR ID
  DH
CURRENT CONSTANT FIELD INDICATOR
  N
CURRENT ELEMENT LENGTH
  0020

CURRENT VALUE FOR PROMPT MESSAGE
  LAST NAME ?

CURRENT ELEMENT FORMAT
  AAAAAAAAAAAAAAAAAAAA
CURRENT ELEMENT DEFAULT VALUE
CURRENT VALUE FOR ERROR MESSAGE
      MUST BE A-Z
      UP TO 20 CHARACTERS

CURRENT VALUE FOR HELP MESSAGE
  LAST NAME CONSISTS OF
      ALPHABETICS


COMMAND> INSERT
ENTER ITEM NAME
ELEM
ENTER TRANSACTION NAME
EMP-INFO
ENTER ELEMENT NAME
LAST-NAME
INSERT AFTER WHICH ELEMENT?
DEPT-NUM
CA014:  ELEMENT NOT USED IN THIS TRANSACTION
CA016:  ITEM NOT FOUND


COMMAND> DI/TR/EMP-INFO
CURRENT TRANSACTION LENGTH
  0053
CURRENT APPLICATION NAME
  PAYROLL-APPL
CURRENT TRANSACTION CREATE DATE
  810227
CURRENT TRANSACTION CREATOR ID
  DH
ELEMENT(S) IN THIS TRANSACTION:
                                    START  LENGTH
  EMP-NUM                              1       9
  FIRST-NAME                          10      10
  PAY-RATE                            30       4
  MGR                                 34      20

COMMAND> REMOVE/ELEM/EMP-INFO
ENTER ELEMENT TO BE REMOVED
FIRST-NAME

ELEMENT REMOVAL COMPLETE


COMMAND> INSERT/ELEM
ENTER TRANSACTION NAME
EMP-INFO
ENTER ELEMENT NAME
LAST-NAME
INSERT AFTER WHICH ELEMENT?
EMP-NUM
```

```
ENTER START POSITION IN TRANSACTION
10
ELEMENT INSERTED


COMMAND: INSERT/ELEM/DSP-INFO
ENTER ELEMENT NAME
ITEM NUM
INSERT AFTER WHICH ELEMENT?
[illegible]

ENTER START POSITION IN TRANSACTION
[illegible]
NEW ELEMENT OVERLAPS ANOTHER ELEMENT.
   ELEMENT NOT INCLUDED IN TRANSACTION.


COMMAND: REMOVE/ELEM FOR INFO
ENTER ELEMENT TO BE REMOVED
[illegible]

ELEMENT REMOVAL COMPLETE.


COMMAND: INSERT/ELEM/FMT-INFO
ENTER ELEMENT NAME
[illegible]
INSERT AFTER WHICH ELEMENT?
[illegible]-RA[illegible]

ENTER START POSITION IN TRANSACTION
[illegible]
ELEMENT INSERTED


COMMAND: [illegible]
CURRENT TRANSACTION LENGTH
  [illegible]
CURRENT APPLICATION NAME
  [illegible]
CURRENT TRANSACTION [illegible]
  [illegible]
CURRENT [illegible] LOCATION IS
  [illegible]
[illegible]
                                                    [illegible]     LENGTH
  [illegible]                                           1          9
  [illegible]                                       [illegible]   20
  [illegible]                                       [illegible]    4
  [illegible]                                       [illegible]    3


[illegible]
[illegible]
[illegible]
[illegible]
```

```
COMMAND [illegible]-NAME
[illegible]

[illegible]-NAME
[illegible] CPUS [illegible]

[illegible] END
[illegible] CPUS [illegible]

[illegible]
```

TEST DATA AFTER DEMONSTRATION OF PROCESS I

Data Base Relationships

Transactions:	EMP-INFO	DEPT-INFO	PARTS-INFO
Elements:	EMP-NUM LAST-NAME PAY-RATE DEPT-NUM	DEPT-NUM MGR EMP-NUM	PART-NUM DEPT-NUM QTY

Transactions

Name:	EMP-INFO	DEPT-INFO	PARTS-INFO
Length:	36	32	14
Application:	PAYROLL-APPL	PARTS-APPL	PARTS-APPL
Entry Start Line:	99	99	1
Entry End Line:	-	-	19
Message Start Line:	-	-	20
Message End Line:	-	-	23

Elements

All elements remain the same except LAST-NAME, in which the error message and description are changed. New elements are PART-NUM and QTY.

Name:	PART-NUM	QTY
Number:	000001	000002
Length:	7	4
Description:	PART NUMBER	QUANTITY ON HAND
Proponent:	INVENTORY	INVENTORY
Constant:	N	N
Format:	9999999	9999

Default:	0000000	00000
Prompt:	WHAT IS THE PART NUMBER?	HOW MUCH?
Error Message:	MUST BE A NUMBER UP TO 7 DIGITS	NOT NUMERIC
Help Message:	FORMAT IS NUMERIC	4 DIGIT NUMBER

Name:	LAST-NAME
Number:	000102
Length:	20
Description:	LAST NAME OF EMPLOYEE
Proponent:	PERSONNEL
Constant:	N
Format:	AAAAAAAAAAAAAAAAAAAA
Default:	b̸
Prompt:	LAST NAME?
Error Message:	LAST NAME: (UP TO 20 CHARACTERS)
Help Message:	LAST NAME CONSISTS OF ALPHABETICS

Screen Definitions

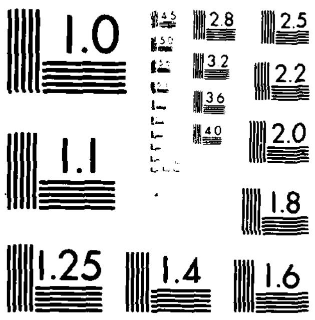
Transaction:	EMP-INFO	EMP-INFO	EMP-INFO
Element:	EMP-NUM	FIRST-NAME	PAY-RATE
Start Position:	1	10	30
Prompt X:	-	-	-
Prompt Y:	-	-	-
Prompt Contents:	-	-	-

AD-A110 120   GEORGIA INST OF TECH ATLANTA   F/G 5/5
COMPUTER AIDED SPECIFICATION TESTING SYSTEM (CASTS). VOLUME I. --ETC(U)
MAR 81 D E HUMPHREY, D P MILLARD   DAAK70-79-D-0087

UNCLASSIFIED   NL

2 OF 2
AD
A110120

END
DATE
FILMED
3-82
DTIC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0

Data Field X:	-	-	-
Data Field Y:	-	-	-

Transaction:	EMP-INFO	DEPT-INFO	DEPT-INFO
Element:	DEPT-NUM	DEPT-NUM	MGR
Start Position:	34	1	4
Prompt X:	-	-	-
Prompt Y:	-	-	-
Prompt Contents:	-	-	-
Data Field X:	-	-	-
Data Field Y:	-	-	-

Transaction:	DEPT-INFO	PARTS-INFO	PARTS-INFO
Element:	EMP-NUM	PART-NUM	DEPT-NUM
Start Position:	24	1	8
Prompt X:	-	1	1
Prompt Y:	-	2	4
Prompt Contents:	-	PART #:	DEPT #:
Data Field X:	-	11	11
Data Field Y:	-	2	4

Transaction:	PARTS-INFO
Element:	QTY
Start Position:	11
Prompt X:	1
Prompt Y:	6
Prompt Contents:	QUANTITY:
Data Field X:	11
Data Field Y:	6

8.2 TESTING OF PROCESS II

To test all the facets of the Process II program, CASTS2, a comprehensive data base of test transactions and elements was required. The data base must contain elements of every format, which are constant, values, and ranges. This was accomplished with seven elements listed on the sheet entitled Element Data. The data was entered into the data base via the Process I, CASTS1 program. The data base was completed by defining six transactions. These six transactions cover a variety of screen, and prompt mode layouts. These transaction data were entered via Process I (see CASTS Test Data).

The test of Process II is conducted by running each of the four modes, SCREEN, PROMPT, BATCH and DIRECT. The user can enter various amounts of data in each mode and then view the Terminal Session Summary Report, the Log File and the Raw Data File to check the accuracy of the data entry.

TEST TRANSACTION LAYOUTS

```
                                     POSITION
TRANS                      1                   2                   3
NAME     1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
-----------------------------------------------------------------------------

TRANS 1  A A A     S 9 9 . 9     X X X

TRANS 2  $ 9 . 9 9         X X X X                       Z Z 9

TRANS 3  S 9 9 . 9 Z Z 9   X X X X   $ 9 . 9 9   X X X             A A A

TRANS 4  X X X X           S 9 9 . 9

TRANS 5  A A A             S 9 9 . 9 X X X     Z Z 9     X X X X   $ 9 . 9 9

TRANS 6  A A A     9 9 9 9           X X X X
```

ELEMENT DATA

ELEMENT NUMBER	FORMAT	MISCELLANEOUS
1	AAA	
2	S99.9	
3	XXX	CONSTANT
4	ZZ9	
5	$9.99	
6	XXXX	VALUES
7	9999	RANGE

CASTS
<u>TEST DATA</u>

PROCESS I

```
    DAVID <CR>
    Y <CR>
--------------------------------------------------------------------------------
    CR <CR>
    ELEM 1 <CR>
    FIRST ELEM <CR>
    <CR>
    FIRST PROPONENT <CR>
    ELEM NO 1 <CR>
    N <CR>                                    Constant field indicator
    3 <CR>                                    Element length
    ENTER ELEM  1 >> <CR>
    <CR>
    AAA <CR>                                  Format
    QES <CR>                                  Default
    <CR>                                      No values
    ERROR ELEM 1 <CR>
    <CR>
    HELP ELEM 1 <CR>
    FORMAT IS "AAA" <CR>
    <CR>
--------------------------------------------------------------------------------
    CR <CR>
    ELEM 2 <CR>
    SECOND ELEM <CR>
    <CR>
    SECOND PROPONENT <CR>
    ELEM NO 2 <CR>
    N <CR>
    5 <CR>
    ENTER ELEM 2 >> <CR>
    <CR>
```

```
S99.9 <CR>
-22.2 <CR>
N <CR>                                              No values
ERROR ELEM 2 <CR>
<CR>
HELP ELEM 2 <CR>
FORMAT IS "S99.9" <CR>
<CR>
-------------------------------------------------------------------------------
CR <CR>
ELEM 3 <CR>
THIRD ELEM <CR>
<CR>
THIRD PROPONENTS <CR>
ELEM NO 3 <CR>
Y <CR>                                              Constant field
3 <CR>
ENTER ELEM 3 >> <CR>
<CR>
XXX <CR>
ZZZ <CR>                                            Default
ERROR ELEM 3 <CR>
<CR>
HELP ELEM 3 <CR>
THE FORMAT IS "XXX" <CR>
<CR>
-------------------------------------------------------------------------------
CR <CR>
ELEM 4 <CR>
FORTH ELEM <CR>
<CR>
FORTH PROPONENT <CR>
ELEM NO 4 <CR>
N <CR>
3 <CR>
ENTER ELEM 4 >> <CR>
```

```
<CR>
ZZ9 <CR>
025 <CR>                                    Default
N<CR>                                       No values
ERROR ELEM 4 <CR>
<CR>
HELP ELEM 4 <CR>
FORMAT IS "ZZ9" <CR>
<CR>
-------------------------------------------------------------------------------
CR <CR>
ELEM 5 <CR>
FIFTH ELEM <CR>
<CR>
FIFTH PROPONENT <CR>
ELEM NO 5 <CR>
N <CR>                                      Not a constant
5 <CR>
ENTER ELEM 5 <CR>
<CR>
$9.99 <CR>
$5.55 <CR>
ERROR ELEM 5 <CR>
<CR>
HELP ELEM 5 <CR>
FORMAT IS "$9.99" <CR>
<CR>
-------------------------------------------------------------------------------
CR <CR>
ELEM 6 <CR>
SIXTH ELEM <CR>
<CR>
SIXTH PROPONENT <CR>
ELEM NO 6 <CR>
N <CR>
4 <CR>
```

```
ENTER ELEM 6 <CR>
<CR>
XXXX <CR>
A12B <CR>
ONE <CR>
TWO <CR>
THREE <CR>
1 FT <CR>
1 IN <CR>
Z IN <CR>
<CR>
ERROR ELEM 6 <CR>
<CR>
HELP ELEM 6 <CR>
FORMAT IS 'XXXX' <CR>
<CR>
```

```
CR <CR>
ELEMENT <CR>
SEVENTH ELEM <CR>
<CR>
SEVENTH PROPONENT <CR>
ELEM NO 7 <CR>
N <CR>
4 <CR>
ENTER ELEM 7 >> <CR>
<CR>
9999 <CR>
1256 <CR>
R <CR>                                  Ranges
1000 <CR>                               Low
2000 <CR>                               High
ERROR ELEM 7 <CR>
<CR>
HELP ELEM 7 <CR>
FORMAT IS "9999" <CR>
```

```
RANGE 1000 - 2000 <CR>
<CR>
--------------------------------------------------------------------------------
BU <CR>
TRANS 1 <CR>
APPLIC 1 <CR>
Y <CR>                                       Screen
1 <CR>                                       Start line
16 <CR>                                      End line
17 <CR>                                      Start line err msg
23 <CR>                                      End line err msg
ELEM 1 <CR>
1 <CR>                                       Start position in trans
1 <CR>                                       Screen prompt position (X)
5 <CR>                                       Screen prompt position (Y)
ELEM 1 >> <CR>                               Prompt
11 <CR>                                      Data start position (X)
5 <CR>                                                         (Y)
ELEM 2 <CR>
6 <CR>
10 <CR>
10 <CR>
ELEM 2 >> <CR>
27 <CR>
10 <CR>
ELEM 3 <CR>
13 <CR>
20 <CR>
15 <CR>
ELEM 3 >> <CR>
31 <CR>
15 <CR>
<CR>
--------------------------------------------------------------------------------
BU <CR>
TRANS 2 <CR>
```

```
APPLIC 2 <CR>
Y <CR>
1 <CR>
16 <CR>
17 <CR>
22 <CR>
ELEM 4 <CR>
25 <CR>
55 <CR>
16 <CR>
ELEM 4 >> <CR>
66 <CR>
16 <CR>
ELEM 5 <CR>
1 <CR>
30 <CR>
10 <CR>
ELEM 5 >> <CR>
41 <CR>
10 <CR>
ELEM 6 <CR>
10 <CR>
10 <CR>
1 <CR>
ELEM 6 ?? <CR>
21 <CR>
1 <CR>
<CR>
--------------------------------------------------------------------------------
BU <CR>
TRANS 3 <CR>
APPLIC 3 <CR>
N <CR>                                          No screen
ELEM 2 <CR>
1 <CR>
ELEM 4 <CR>
```

```
6 <CR>
ELEM 6 <CR>
10 <CR>
ELEM 5 <CR>
15 <CR>
ELEM 3 <CR>
21 <CR>
ELEM 1 <CR>
30 <CR>
<CR>
------------------------------------------------------------------------
BU <CR>
TRANS 4 <CR>
APPLIC 1 <CR>                                Same application as TRANS 1
Y <CR>
5 <CR>
10 <CR>
18 <CR>
22 <CR>
ELEM 2 <CR>
10 <CR>
10 <CR>
10 <CR>
ELEM 2 >> <CR>
20 <CR>
10 <CR>
ELEM 6 <CR>
1 <CR>
30 <CR>
5 <CR>
ELEM 6 >> <CR>
42 <CR>
5 <CR>
<CR>
------------------------------------------------------------------------
```

```
BU <CR>
TRANS 5 <CR>
APPLIC 2 <CR>
Y <CR>
1 <CR>
15 <CR>
18 <CR>
22 <CR>
ELEM 1 <CR>
1 <CR>
1 <CR>
1 <CR>
ELEM 1 >>
11<CR>
1 <CR>
ELEM 2 <CR>
10 <CR>
55 <CR>
12 <CR>
ELEM 2 >> <CR>
66 <CR>
12 <CR>
ELEM 3 <CR>
15 <CR>
30 <CR>
7 <CR>
ELEM 3 >> <CR>
41 <CR>
7 <CR>
ELEM 4 <CR>
20 <CR>
1 <CR>
12 <CR>
ELEM 4 >> <CR>
11 <CR>
12 <CR>
```

```
ELEM 5 <CR>
30 <CR>
55 <CR>
1 <CR>
ELEM 5 >> <CR>
66 <CR>
1 <CR>
ELEM 6 <CR>
25 <CR>
30 <CR>
5 <CR>
ELEM 6 >> <CR>
41 <CR>
5 <CR>
<CR>
```

```
BU <CR>
TRANS 6 <CR>
APPLIC 1 <CR>
N <CR>
ELEM 1 <CR>
1 <CR>
ELEM 7 <CR>
6 <CR>
15 <CR>
<CR>
```

APPENDICES

APPENDIX A

LIST OF ERROR MESSAGES

1.0 Process I Error Messages

CA001: INVALID COMMAND

The command entered was invalid. The valid commands are BUILD, CREATE, DELETE, DISPLAY, MODIFY, REMOVE, INSERT, HELP and END. Only the first two letters of each command are used and the remaining letters are insignificant. For more information, see CASTS User's Manual Section 4.3.

CA002: TRANSACTION BY THIS NAME DOES NOT EXIST

There is no transaction specified with the name given. Check that the transaction name was typed correctly. Common mistakes that result in this message are typographical errors and giving an element name instead of a transaction name.

CA003: ELEMENT BY THIS NAME DOES NOT EXIST

There is no element specified with the name given. Check that the element name was typed correctly. Common mistakes that result in this message are typographical errors and giving a transaction name instead of an element name.

CA004: INVALID ITEM GIVEN

The item name entered in the second field of the command was invalid. The valid item names are TRANS, THDR (transaction header), ELEM, EHDR (element header), PROMPT, EDIT, ERROR, HELP, LIST and SCREEN. Only the first two letters of each item name are used and the remaining letters are insignificant. For more information, see CASTS User's Manual Section 4.3.

CA005: UNKNOWN SYSTEM ERROR - SEE ANALYST

A return code internal to the program (RETURN-STATUS) has become an unexpected value (i.e. other than 000 to 006). Please write down what you were doing and see the CASTS System Administrator.

CA006: INVALID OPTION

The three letter option given was invalid. The option entered must be one of those displayed just prior to the ENTER OPTION prompt and it must be typed exactly as shown. If both of these were done correctly and the message still appears, write down what you were doing and see the CASTS System Administrator.

CA007: TRANSACTION WITH THIS NAME ALREADY EXISTS

A transaction already exists with the transaction name given. There cannot be more than one transaction built with the same name. Display the transaction and check if you already built the transaction or try a different transaction name.

CA008: ELEMENT WITH THIS NAME ALREADY EXISTS

An element already exists with the element name given. There cannot be more than one element created with the same name. Display the element and check if you already created the element and/or wish to use it as is. If not, try a different element name.

CA009: NO HELP AVAILABLE

There is no help available for the topic requested.

CA010: ONLY THE CREATOR IS AUTHORIZED TO DELETE

The user id of the person currently running Process I must match the creator id of an element or a transaction in order to be able to delete it.

CA011: ITEM NAME ILLEGAL FOR THIS COMMAND

The item name given may be valid, however it cannot be used in conjunction with the command given. Use the HELP command with the name of the command in question to find out what item names are valid for that command, e.g. HELP/<command>.

CA012: ELEMENT NOT DELETED

The request to delete an element has not been completed. An element must be removed from all transactions before it can be successfuly deleted.

CA013: INVALID VALUE FOR THIS FIELD

The value given does not satisfy the criteria for this data field. Use the HELP command (HELP or ?) to get information about what should be entered. For more information, see User's Manual Section 4.1.

CA014: ELEMENT NOT USED IN THIS TRANSACTION

The element name given is that of an element that does not belong to the transaction given.

CA015: DATA BASE ACCESS ERROR

An error has been made in accessing the stored specifications. Check to see how much of the operation that was in progress was completed. Try to recover by making modifications to complete the operation. If unrecoverable, see the CASTS System Administrator.

CA016: ITEM NOT FOUND

This message appears when an item requested cannot be found in the stored specifications. If the item requested was a prompt, edit range or value check, error message or help message, there may not be one for the element given. If a particular line of an item was requested, that item may not have a line with the given line number.

CA017: ITEM INDEX INVALID

The item index given was invalid. Either it contains illegal characters in an illegal format or it refers to a line number of an item, such as a prompt, where that item does not have a line with the given line number.

CA018: INVALID CHARACTER DETECTED IN INPUT
Non-printable and certain special characters are not allowed as input. The following special characters are valid: !"#$%&'()-=\/?.,<>;+:*@[].

2.0 Process II Error Messages

Errors can occur in Process II when raw application data entered does not conform to the edit criteria specified in Process I. Data are checked position by position against the format specification and the first error found is flagged. The message states the type of error and if applicable, the position where it was encountered.

DATA MUST BE NUMERIC - The character in the position stated must be numeric, that is, a digit from 0 to 9.

DATA MUST BE ALPHABETIC - The character in the position stated must be alphabetic, A through Z.

NO SPECIAL CHARACTERS ALLOWED - The character in the position stated can be alphabetic, A through Z, or a digit, 0 through 9; but it cannot be a special character. Special characters include period, comma, semi-colon, percent, etc.

SPECIAL CHARACTER DID NOT MATCH FORMAT - The character in the position stated must be the particular special character that was given in the element format

DATA MUST BE ONE OF THE FOLLOWING VALUES - The data given for the element satisfied the format specification but was not one of the values given in the value check list of valid values.

DATA MUST FALL WITHIN THE RANGE - The data given for the element satisfied the format specification but failed the range check. It must fall on or within the upper and lower bounds given.

DATA MUST BE A DECIMAL POINT - The character in the position stated must be a decimal point, as specified in the format.

DATA MUST BE A PLUS OR MINUS SIGN - The character in the position stated must be a numeric sign, either plus (+) or minus (-).

Data base error and input/output error messages for Process II are stored as help topics and may be tailored to the installation by the CASTS System Administrator.

APPENDIX B

COMPILATION AND LINKAGE OF PROGRAMS

The process of transforming source code into executable code is made easier by the use of command files and default file name conventions. The following steps describe this process for the PDP-11/70.

(1) To edit the source code using the EDT editor use the following format (in this case, the file name extension must be used):

```
EDI/EDT  <file name>.<extension>
```

All source code for programs that access the DBMS is contained in files with the extension ".DML", indicating that they include Data Manipulation Language (DML) statements.

(2) Prior to COBOL compilation, the programs must be run through the DML preprocessor. This converts the DML statements to COBOL calls to routines that combine to form the DBMS.

```
PDS> @DML
RUN [11, 121]DML
DML PROCESSOR SIGNON PROTOCOL:
OUTPUT, LISTING = INPUT
DML> <File name>, <file name> = <file name>
```

The file name given should be the same in each position in the above command. The correct extensions will be assumed. The input file is the source code file containg DML statements and having the ".DML" extension.

The listing produced will be stored in <file name>.PRT and the altered source code will be in <file name>.CBL, the output file.

Process II report programs do not use the DBMS and therefore do not need to be preprocessed. Their source code files have the COBOL source code extension, ".CBL".

(3) The next step is to compile the COBOL program. There is a COBOL compile command file for each program. The first three letters of compilation commands are "COB". The remaining letters identify the program and usually correspond to the first letters of the source code file name. For example, to compile the Data Element Dictionary program, ELERPT, the command file name is COBELE. The extension for command files is ".CMD" and is assumed when the file name is precede by "@", meaning 'execute.' To compile the CASTS Process I program:

```
PDS> @COBCASTS1
```

The compilation step produces a compiler listing of the source code in <file name>.LST and an object file in <file name>.OBJ.

The command @COBCASTS2 compiles CASTS2 without a listing. To get a listing, but no object code and no map, use

```
COBOL/LI:CASTS2/SW:(/CVF)/NOOBJ CASTS2
```

Alternatively, to obtain a listing and a map, but no object code use

```
COB/LI:CASTS2/SW:(/CVF/CSEG:600/OV/PFM:15/MAP/NOOBJ
```

and when the system prompts for the file name, enter "CASTS2".

(4) This step is only performed for CASTS1 AND CASTS2, which have overlays. For all other programs, skip this step. Programs that are overlaid must have an ovewrlay description file to indicate how to organize the overlays. The directives are called the overlay description language and the extension on the file containing them is ".ODL". An ODL file is created automatically for the program by the COBOL compiler. It must be merged with another file describing the data base overlay structure. To do this for CASTS1, use the merge facility as shown.

```
PDS> MRG
PLEASE ENTER FILE SPECIFICATION FOR OUTPUT FILE
C1
DO YOU WANT A DEFAULT MERGE?
PLEASE ANSWER Y(ES), N(O), OR H(ELP) N
DO YOU WANT AN ABBREVIATED OR MERGED ODL FILE?
PLEASE ANSWER A(BBREVIATED), M(ERGED), OR H(ELP) M
DO YOU WANT TO INCLUDE THE COBOL DEBUGGER (CID)?
PLEASE ANSWER Y(ES) OR N(O) N
DO YOU WANT TO OVERLAY I/O SUPPORT ROUTINES?
PLEASE ANSWER Y(ES), N(O), OR H(ELP) Y
DO YOU WNAT TO USE THE DEC SUPPLIED I/O OVERLAY STRUCTURE
PLEASE ANSWER Y(ES), N(O), OR H(ELP) Y
PLEASE ENTER FILE SPECIFICATION FOR INPUT ODL FILE
[11,120]DBMSCB
PLEASE ENTER OBJECT FILE DEVICE AND UIC
<CR>
ANY MORE INPUT ODL FILES?
PLEASE ANSWER Y(ES) OR N(O) Y
PLEASE ENTER FILE SPECIFICATION FOR INPUT ODL FILE
CASTS1
PLEASE ENTER OBJECT FILE DEVICE AND UIC
<CR>
ANY MORE INPUT ODL FILES?
```

```
PLEASE ANSWER Y(ES) OR N(O) N
ODL FILE MERGE COMPLETE
MERGED ODL FILE IS: C1
```

The same procedure is used to produce a merged ODL file for CASTS2, except that the output file should be "C2" instead of "C1". There are also more input ODL files, since those of the subroutines for CASTS2 must be included. The input ODL files are:

```
[11,120]DBMSCB
CASTS2
CHECK
CLEAR
POSN
```

(5) The next step is to link the program and the libraries, and if necessary the data base. There is a command file for each program to perform this function. The link command files are named in the same way as the compile commands, with "LNK" being the first three letters. To link CASTS1:

```
PDS> @LNKCASTS1
```

The executable code is stored in <file name>.TSK. In this case, it would be CASTS1.TSK.

(6) The final step is to execute or run the program.

```
PDS> RUN <file name>
```

Again, the extension (".TSK") is assumed.

APPENDIX C

DIRECTORY OF FILES

All of the files associated with a program have the same name with the appropriate standard extension. The command files for compilation have "COB" as the first three letters and for linkage they have "LNK". Most of the overlay description files contain "ODL" in the file name. "RPT" in the filename indicates that the program is a report program. The possible extensions are:

.CBL	COBOL source code (conventional format)
.DML	Source code with DML statements
.LST	Compilation listing of source code
.OBJ	Object code (compiled program)
.ODL	Overlay description
.PRT	Listing produced by DML processor
.TRM	COBOL source code (terminal format)
.TSK	Executable code

List of files:

C1.ODL	Merged ODL file for CASTS1
C2.ODL	Merged ODL file for CASTS2
CASTS1.CBL	CASTS Process I
CASTS1.DML	"
CASTS1.LST	"
CASTS1.OBJ	"
CASTS1.ODL	"
CASTS1.PRT	"
CASTS1.TSK	"
CASTS2.CBL	CASTS Process II
CASTS2.DML	"

CASTS2.LST	"
CASTS2.OBJ	"
CASTS2.ODL	"
CASTS2.PRT	"
CASTS2.TSK	"
CHECK.CBL	Subroutine for CASTS Process II
CHECK.LST	"
CHECK.OBJ	"
CHECK.ODL	"
CHECK.TRM	"
CLEANDIR.CMD	Command to delete all but latest version of files
CLEAR.CBL	Subroutine for CASTS Process II
CLEAR.LST	"
CLEAR.OBJ	"
CLEAR.ODL	"
CLEAR.TRM	"
COBCASTS1.CMD	Compile CASTS1
COBCASTS2.CMD	Compile CASTS2
COBCHECK.CMD	Compile CHECK
COBCLEAR.CMD	Compile CLEAR
COBDAT.CMD	Compile DATRPT
COBELE.CMD	Compile ELERPT
COBERR.CMD	Compile ERRRPT
COBHELPED.CMD	Compile HELPED
COBHLP.CMD	Compile HLPRPT
COBLOG.CMD	Compile LOGRPT
COBPOSN.CMD	Compile POSN
COBSCR.CMD	Compile SCRRPT
COBTRA.CMD	Compile TRARPT
CS2.CBL	Version of CASTS2 using computational arithmetic
CS2.DML	"
CS2.OBJ	"
CS2.ODL	"
CS2.PRT	"
CS2.TSK	"
CS2ODL.ODL	Merged ODL file for CS2
CSTDMC.DMC	CASTS Device Media Control for DBMS
CSTDMC.LST	"
CSTDMC.MAC	"
CSTDMC.OBJ	"
CSTDMC.PRT	"
CSTDMC.TSK	"
CSTSCH.LST	CASTS Schema for DBMS
CSTSCH.SCH	"
CSTSSC.MAC	CASTS Subschema for DBMS
CSTSSC.OBJ	"
CSTSSC.PRT	"
CSTSSC.SSC	"
CSTSSC.TSK	"
DATRPT.CBL	Application Raw Data Report
DATRPT.LST	"
DATRPT.OBJ	"
DATRPT.ODL	"
DATRPT.TSK	"

DML.CMD	Command to start DML processor
ELEODL.ODL	Merged ODL File for ELERPT
ELERPT.CBL	Data Element Dictionary (Report)
ELERPT.DML	"
ELERPT.LST	"
ELERPT.OBJ	"
ELERPT.ODL	"
ELERPT.PRT	"
ELERPT.TMP	"
ELERPT.TSK	"
ERRRPT.CBL	Terminal Session Error Report
ERRRPT.LST	"
ERRRPT.OBJ	"
ERRRPT.ODL	"
ERRRPT.TSK	"
HE.ODL	Merged ODL File for HELPED
HELPED.CBL	Help Message Text Editor
HELPED.DML	"
HELPED.LST	"
HELPED.OBJ	"
HELPED.ODL	"
HELPED.PRT	"
HELPED.TSK	"
HLPODL.ODL	Merged ODL File for HLPRPT
HLPRPT.CBL	Helped Messages Report
HLPRPT.DML	"
HLPRPT.LST	"
HLPRPT.OBJ	"
HLPRPT.ODL	"
HLPRPT.PRT	"
HLPRPT.TSK	"
LNKCASTS1.CMD	Link CASTS1
LNKCASTS2.CMD	Link CASTS2
LNKCS2.CMD	Link CS2
LNKDAT.CMD	Link DATRPT
LNKELE.CMD	Link ELERPT
LNKERR.CMD	Link ERRRPT
LNKHELPED.CMD	Link HELPED
LNKHLP.CMD	Link HLPRPT
LNKLOG.CMD	Link LOGRPT
LNKSCR.CMD	Link SCRRPT
LNKTRA.CMD	Link TRARPT
LOGIN.CMD	Command to set terminal attributes to a VT52
LOGRPT.CBL	Simulation Interaction Log Report
LOGRPT.LST	"
LOGRPT.OBJ	"
LOGRPT.ODL	"
LOGRPT.TSK	"
POSN.CBL	Subroutine for CASTS Process II
POSN.LST	"
POSN.OBJ	"
POSN.ODL	"
POSN.TRM	"
SCREEN.FTN	Subroutine for CASTS Process II

SCREEN.LST	"
SCREEN.OBJ	"
SCREEN.ODL	"
SCRODL.ODL	Merged ODL file for SCRRPT
SCRRPT.CBL	Transaction Screen Layout (Report)
SCRRPT.DML	"
SCRRPT.LST	"
SCRRPT.OBJ	"
SCRRPT.ODL	"
SCRRPT.PRT	"
SCRRPT.TMP	"
SCRRPT.TSK	"
SORTFIL.DAT	File used to sort records in report programs
TRAODL.ODL	Merged ODL file for TRARPT
TRARPT.CBL	Transaction Dictionary (Report)
TRARPT.DML	"
TRARPT.LST	"
TRARPT.OBJ	"
TRARPT.ODL	"
TRARPT.PRT	"
TRARPT.TMP	"
TRARPT.TSK	"

APPENDIX D

SCHEMA INSTALLATION

The following is an abbreviated guide to building a data base using DBMS-11 V1.0 on the PDP-11/70 under the operating system IAS PDS Version 3.

Edit the schema, DMCL and subschema source programs. Refer to DBA guide for syntax.

Make certain that the necessary DBCS tasks are installed (DBM, CAMP, DBL, ABDUMP).

```
PDS>INSTALL [11,123]NETSS
PDS>INSTALL [11,123]NETDM
```

If the data base is signed on:

```
SCI>DBO SIGNOFF
SCI>DBO UNLOAD
```

Log in to DBA124. Run DBINIT to allocate and initialize the data dictionary file. CAMP must be the only DBCS component loaded. The DMCL NETDM maps only the data dictionary.

```
SCI>DBO LOAD CAMP

PDS>RUN [11,120]DBINIT
INI>NETDM
INI>TOTAL
INI>Y
```

Run the schema compiler, which will make entries into the data dictionary.

```
SCI>DBO SIGNON DMCL:NETDM

PDS>RUN [11,121]SCHEMA
SCH>yourschema.LST=yourschema.SCH
```

Run the DMCL compiler which will also make entries into the data dictionary and will allow you to map the additional files and areas specified in your schema.

```
PDS>RUN [11,122]CBLSHR
CBX>[11,121]DMCL
DMCL>yourdmcl.MAC,yourdmcl.PRT=yourdmcl.DMC
CBX>CNTL Z
```

Assemble the resultant macro source program created by the DMCL compiler.

```
PDS>MAC/OBJ:yourdmcl.OBJ [11,121]DMCLMC/LI+yourdmcl
PDS>MAC/LI:yourdmcl/NOOB [11,121]DMCLMC/LI+yourdmcl
```

Edit [11,124]SAMPDM.LNK replacing all references to a DMCL name with yourdmcl (list file to find old DMCL name).

```
                PDS>COPY [11,124]SAMPDM.LNK SAMPDM.LNK
                PDS>EDIT SAMPDM.LMK
                *LIST
                *PA/olddmcl/yourdmcl
                *EXIT
```

Taskbuild yourdmcl

```
                PDS>@SAMPDM.LNK
```

Remove old task: REMOVE/TASK taskname
Install the new DMCL task

```
                PDS>INSTALL yourdmcl
```

Run DBINIT on all areas and files mapped by the new DMCL. This will allocate the necessary files and initialize them. CAMP must be the only DBCS component loaded.

```
                SCI>DBO SIGNOFF
                SCI>DBO UNLOAD
                SCI>DBO LOAD CAMP

                PDS>RUN [11,120]DBINIT
                INI>yourdmcl
                INI>AREAS=schema-area,schema-area...
```

NOTE: Files instead of areas may be specified in this step but in no case should the data dictionary area (DDLDML-019) or the data dictionary file (DDLDML.DBS) be specified.

Sign on the new data base with the new DMCL.

```
                SCI>DBO SIGNON DMCL:yourdmcl [NO JOURNAL]
```

Run DBCLUC to put the DML protocols into the data dictionary.

```
                PDS>RUN [11,122]CBLSHR
                CBX>[11,121]DBCLUG
                CLUC>SAMPCL.LST=[11,123]DBMSCL.CLU/CV
                CBX>CNTL Z
```

To compile subschemas for the new data base:

```
                PDS>RUN [11,121]SUBSCH
                SUBSCH>yourssc.MAC,yourssc.PRT=yourssc.SSC
```

Assemble the resultant macro source program:

```
PDS>MAC/LI:yourssc/OB:yourssc yourssc
```

To taskbuild the resultant object program edit [11,124]SAMPSS.LNK replacing all references to an old subschema with yourssc.

```
PDS>@SAMPSS.LNK
```

To get reports about your new data base structure, run DBREPS.

```
PDS>RUN [11,122]CBLSHR
CBX>[11,121]DBREPS
REPS>SAMPRF.LST=TI
REPS>CNTLZ
```

NOTE: Refer to page 2-10 of the DBA Guide for information about DBREPS.

Print the report file generated by DBREPS.

```
PDS>PRINT SAMPRF.LST
```

If your data base exists and is complete but is not installed, the following sequence will install it.

```
SCI>DBO SIGNOFF
SCI>DBO UNLOAD DMCL

PDS>INS/PAR:GEN yourssc
PDS>INSTALL yourdmcl

SCI>DBO LOAD DMCL:yourdmcl
SCI>DBO SIGNON
```

APPENDIX E

FUNCTIONAL HIERARCHY DIAGRAMS

Functional hierarchy diagrams illustrate the hierarchical structure of the functions that comprise a system or program. They show the top-down functional breakdown of program modules pictorially, much the same as a business organizational chart or a HIPO Visual Table of Contents. They are initially drawn during the design, phase and evolve through the coding and testing of programs, finally serving as a method of documentation.

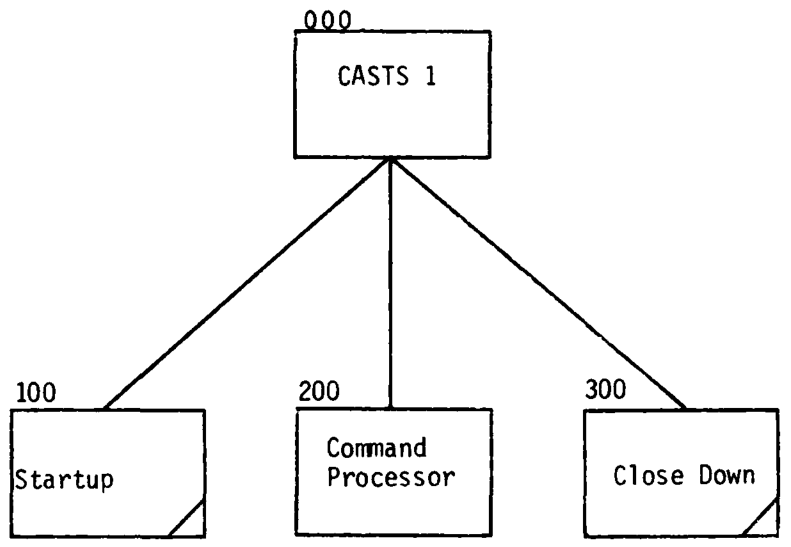
The functional hierarchy diagrams describe the system or program down to a basic or primitive function level. At this level, a function module may generally be implemented by a single COBOL paragraph of code. However, the diagrams do not necessarily depict all code modules within a program. A basic level function is indicated on the diagram by a diagonal slash across the lower right corner of a function box.

Conventions adhered to by the functional hierarchy diagrams for CASTS facilitate a straightforward transfer of information. The abovementioned slash designates a basic level function. A shaded upper right corner of a function box indicates that the function is used more than once within the program structure. The diagram identification number corresponds to that of the function described by that diagram. The function identification number is the same as the numeric prefix of the COBOL paragraph name for the implementing section of code. The date of the diagram is the date of the program.

The following functional hierarchy diagrams describe the CASTS programs.

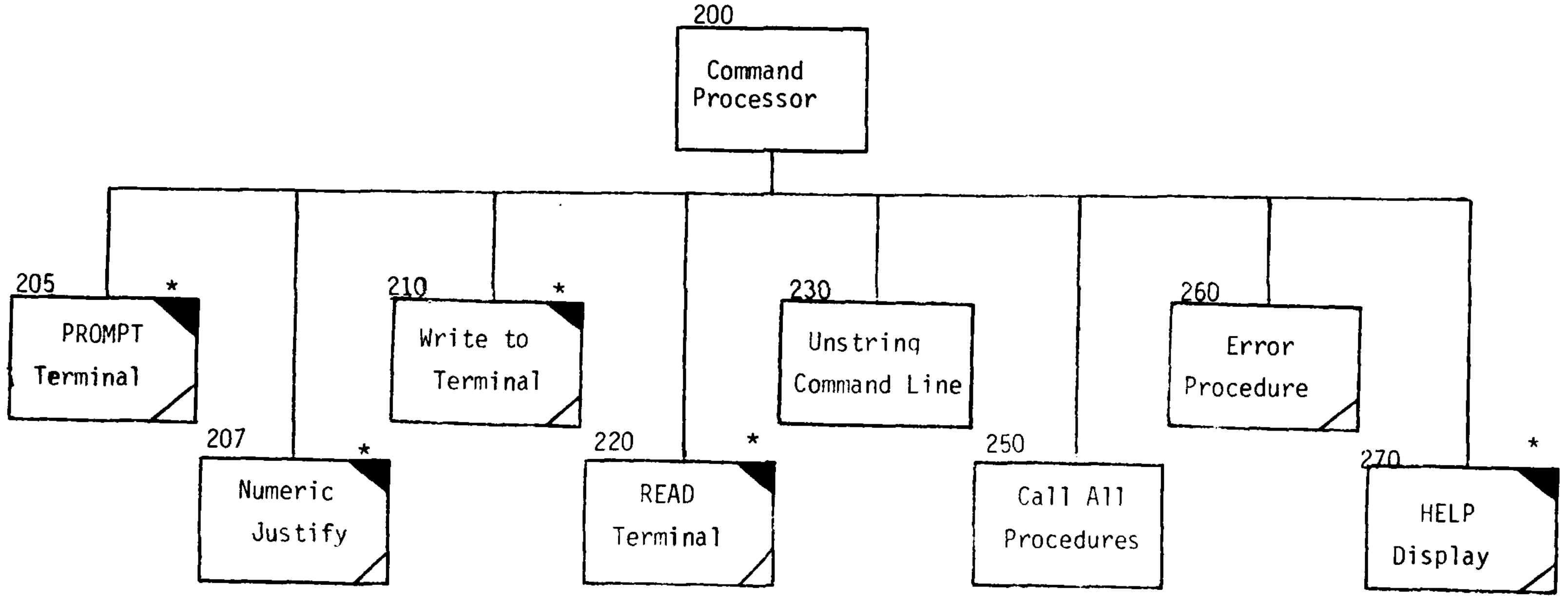
CASTS	System/Program: CASTS1	Date: 12/80 Page: 1 of 10
Diagram ID: 000	Name: CASTS Process I	Description: Specification Process

Functional Hierarchy Diagram



CASTS	System/Program: CASTS1	Date: 12/80	Page: 2 of 10
Diagram ID: 200	Name: CASTS Process I	Description: Specification Process	

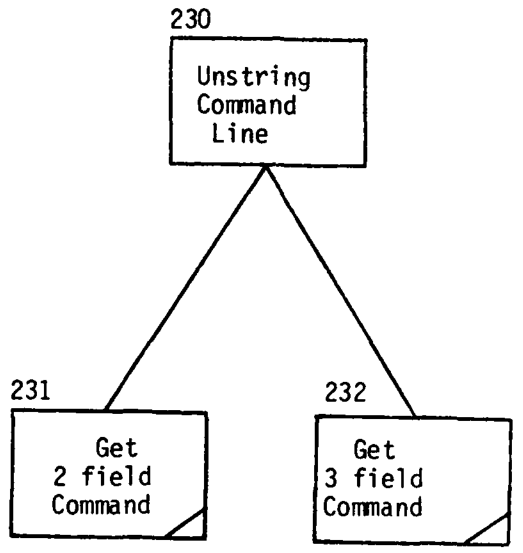
Functional Hierarchy Diagram



* These are service routines and are referenced throughout CASTS1.

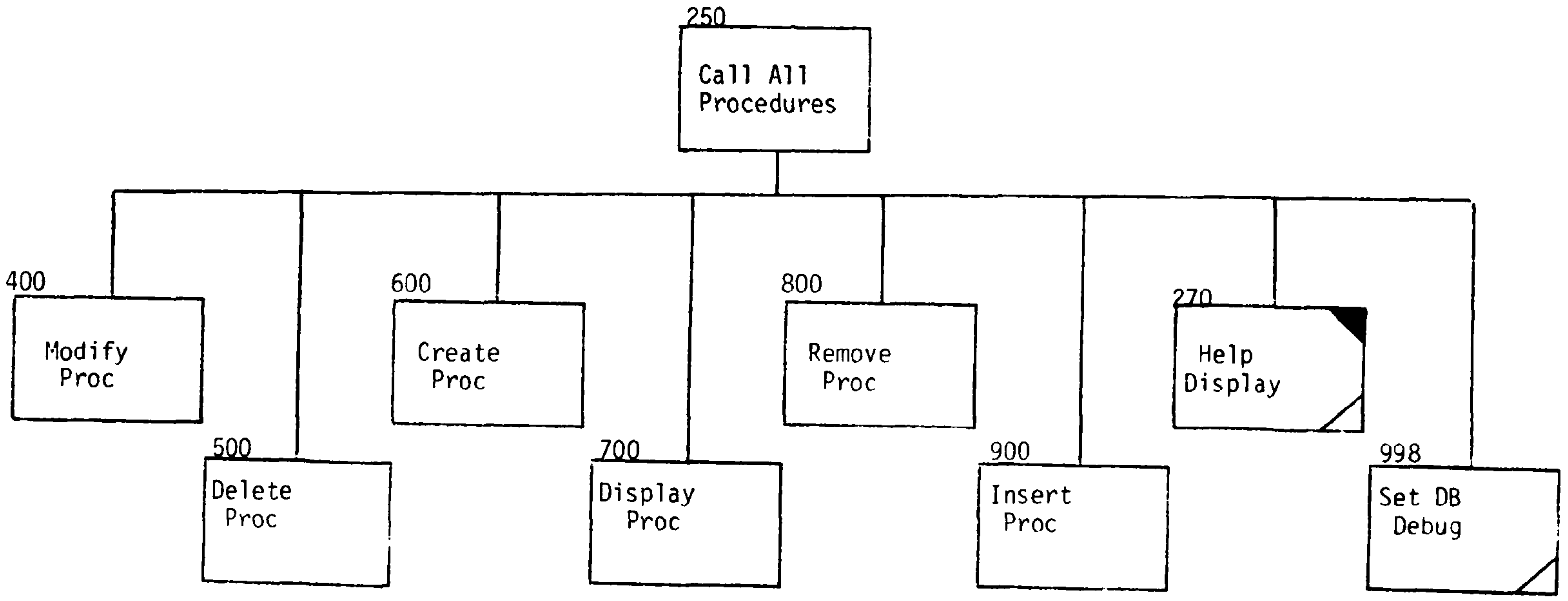
CASTS	System/Program: CASTS1	Date: 12/80	Page: 3 of 10
Diagram ID: 230	Name: CASTS Process I	Description: Specification Process	

Functional Hierarchy Diagram



CASTS	System/Program: CASTS1	Date: 12/80	Page: 4 of 10
Diagram ID: 250	Name: CASTS Process I	Description: Specification Process	

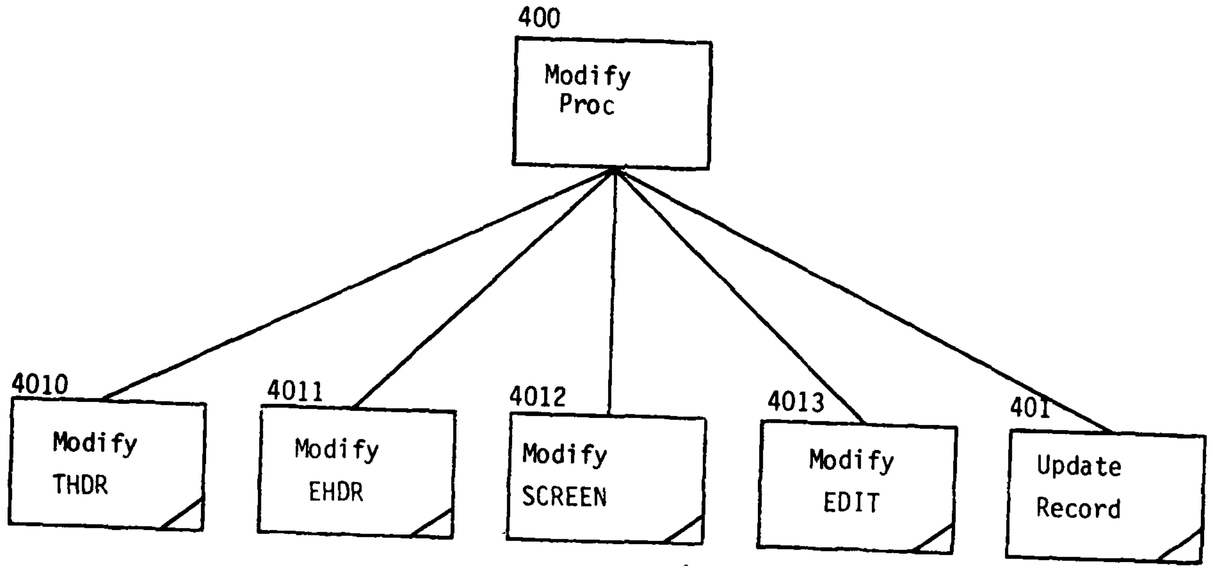
Functional Hierarchy Diagram



CASTS

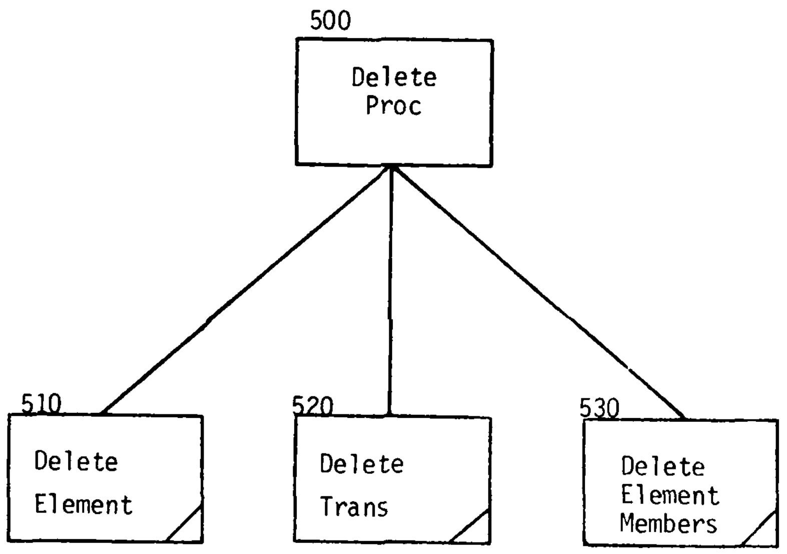
	System/Program: CASTS1	Date: 12/80	Page: 5 of 10
Diagram ID: 400	Name: CASTS Process I	Description: Specification Process	

Functional Hierarchy Diagram



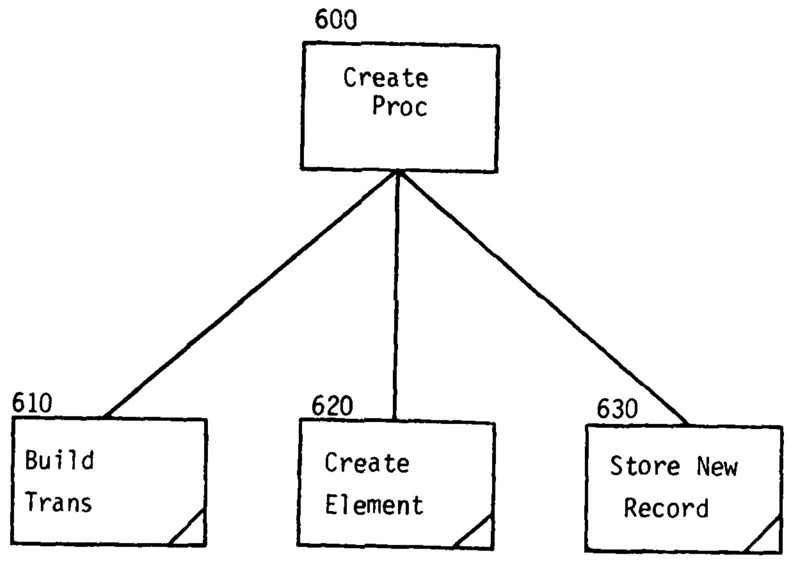
CASTS	System/Program: CASTS1	Date: 12/80 Page: 6 of 10
Diagram ID: 500	Name: CASTS Process I	Description: Specification Process

Functional Hierarchy Diagram



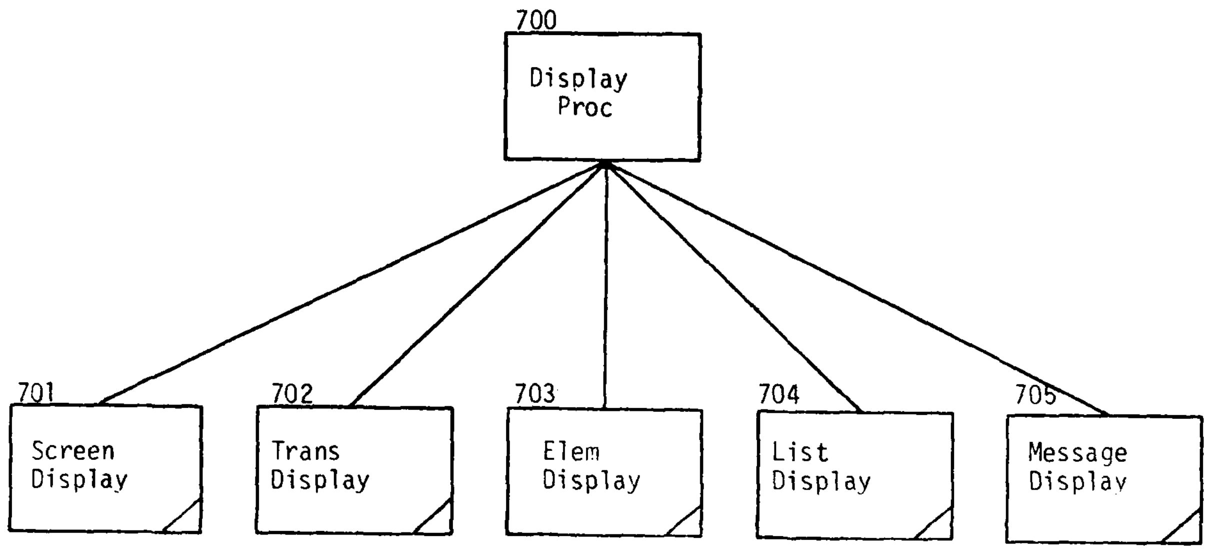
CASTS	System/Program: CASTS1	Date: 12/80	Page: 7 of 10
Diagram ID: 600	Name: CASTS Process I	Description: Specification Process	

Functional Hierarchy Diagram



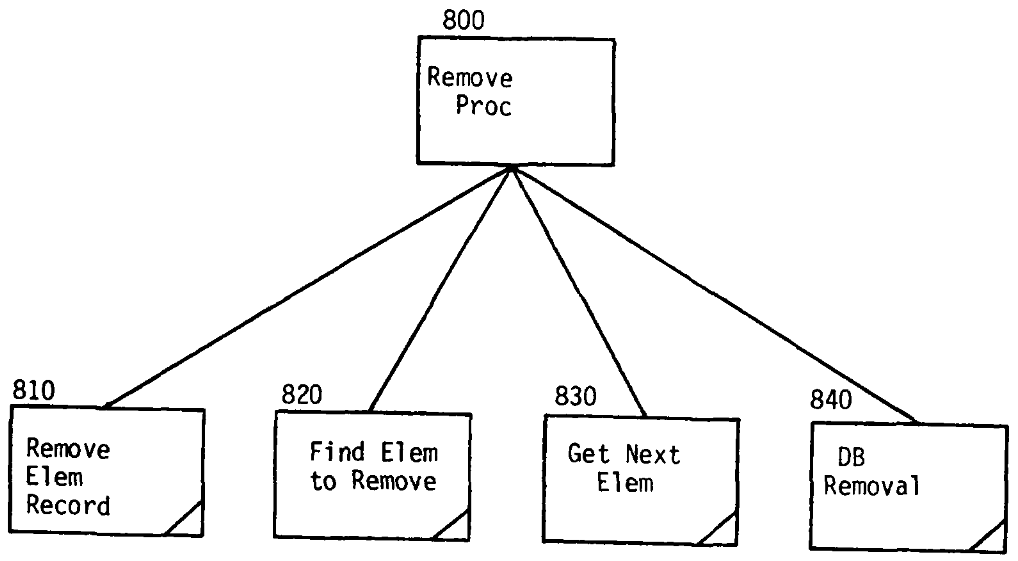
CASTS	System/Program: CASTS1	Date: 12/80	Page: 8 of 10
Diagram ID: 700	Name: CASTS Process I	Description: Specification Process	

Functional Hierarchy Diagram



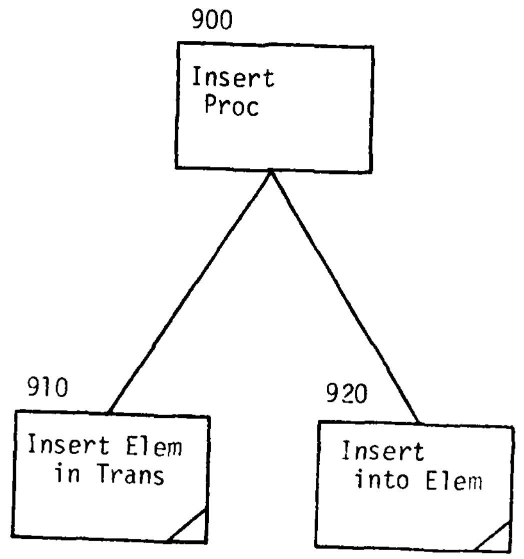
CASTS	System/Program: CASTSI	Date: 12/80	Page: 9 of 10
Diagram ID: 800	Name: CASTS Process I	Description: Specification Process	

Functional Hierarchy Diagram



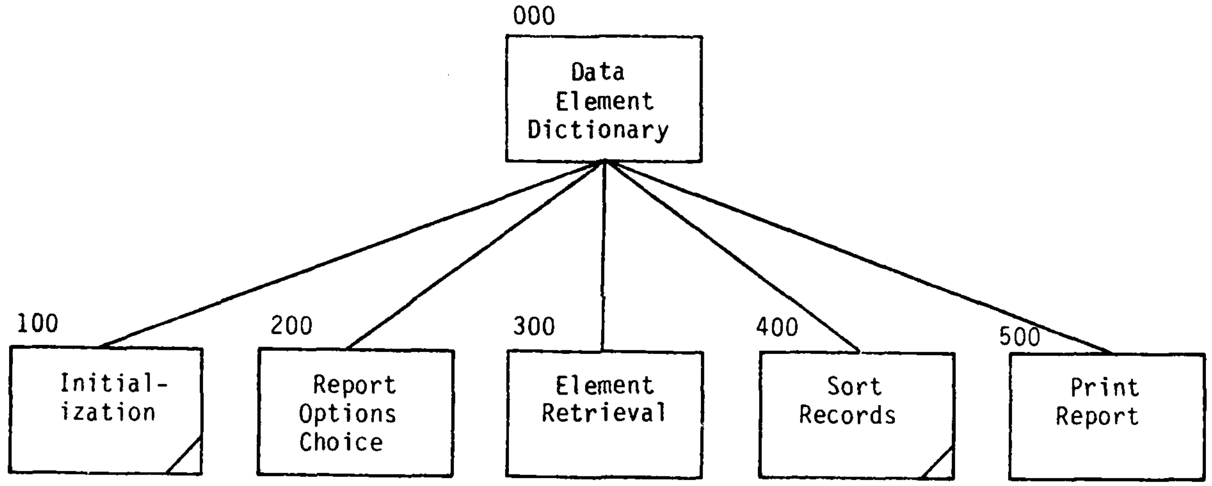
CASTS	System/Program: CASTSI	Date: 12/80	Page: 10 of 10
Diagram ID: 900	Name: CASTS Process I	Description: Specification Process	

Functional Hierarchy Diagram



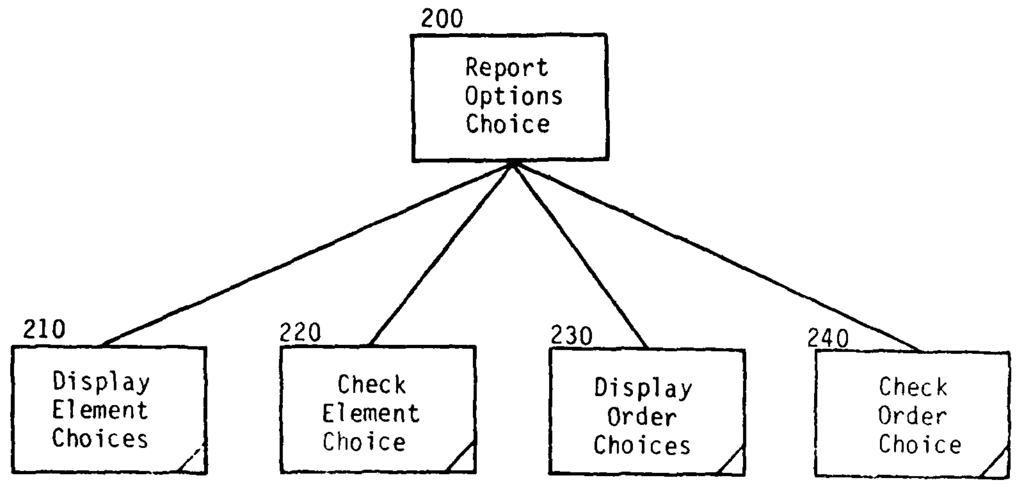
CASTS	System/Program: ELERPT	Date: 4/80	Page: 1 of 11
Diagram ID: 000	Name: Data Element Dictionary	Description: Process I Report	

Functional Hierarchy Diagram



CASTS	System/Program: ELERPT	Date: 4/80	Page: 2 of 11
Diagram ID: 200	Name: Data Element Dictionary	Description: Process I Report	

Functional Hierarchy Diagram

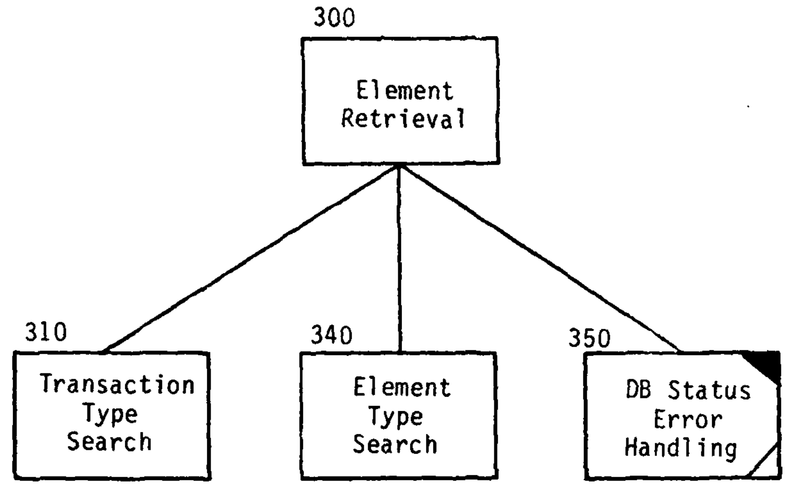


Element Selection Options:
1) All elements in the data base
2) Range of element numbers
3) Elements belonging to a proponent
4) Elements in an application

Element Order Options:
1) Numeric by element number
2) Alphabetic by element name

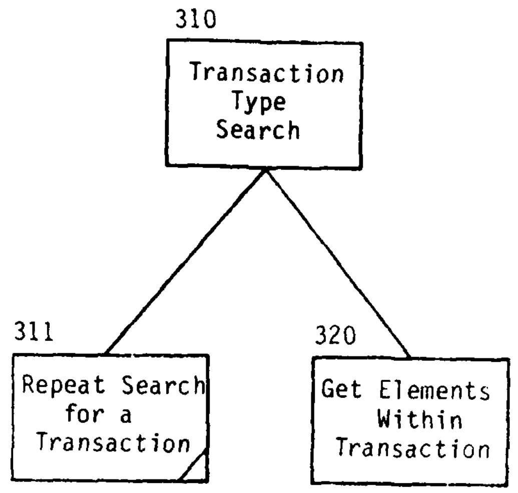
CASTS	System/Program: ELERPT		Date: 4/80	Page: 3	of 11
Diagram ID: 300	Name: Data Element Dictionary		Description: Process I Report		

Functional Hierarchy Diagram



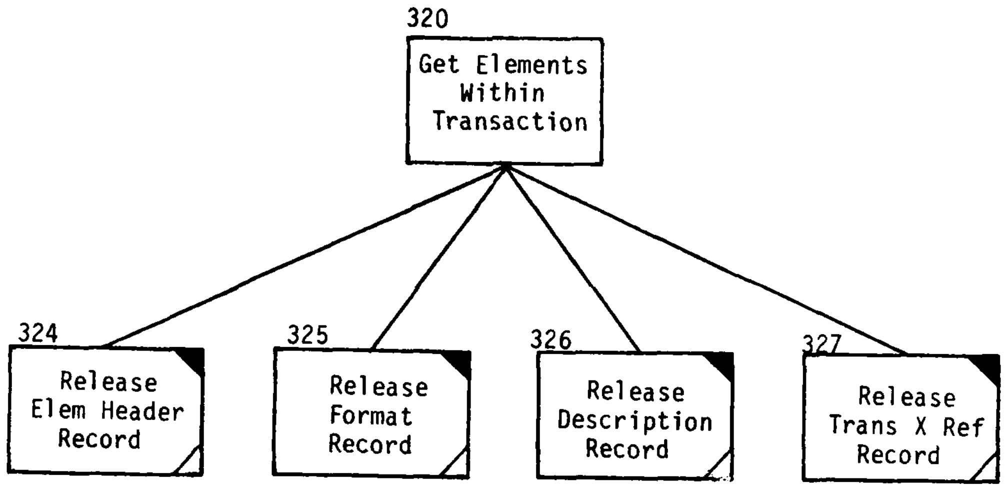
CASTS	System/Program: ELERPT	Date: 4/80	Page: 4 of 11
Diagram ID: 310	Name: Data Element Dictionary	Description: Process I Report	

Functional Hierarchy Diagram


310
Transaction Type Search
311
Repeat Search for a Transaction
320
Get Elements Within Transaction

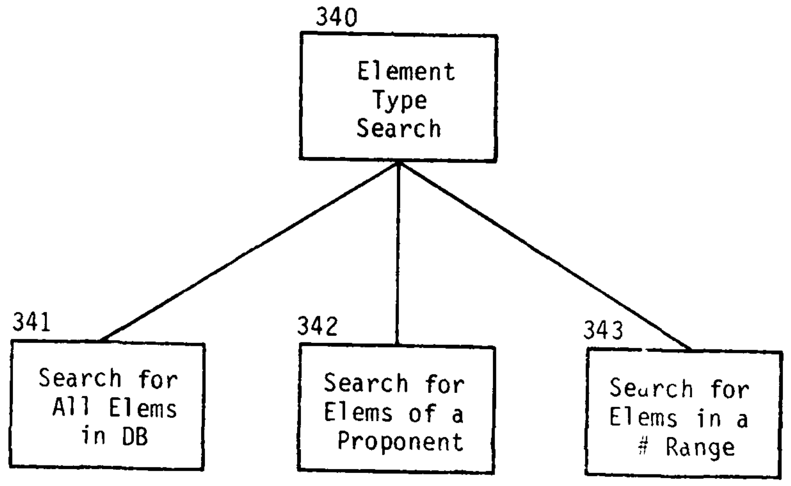
CASTS	System/Program: ELERPT	Date: 4/80	Page: 5 of 11
Diagram ID: 320	Name: Data Element Dictionary	Description: Process I Report	

Functional Hierarchy Diagram


320
Get Elements Within Transaction
324
Release Elem Header Record
325
Release Format Record
326
Release Description Record
327
Release Trans X Ref Record

CASTS	System/Program: ELERPT	Date: 4/80	Page: 6 of 11
Diagram ID: 340	Name: Data Element Dictionary	Description: Process I Report	

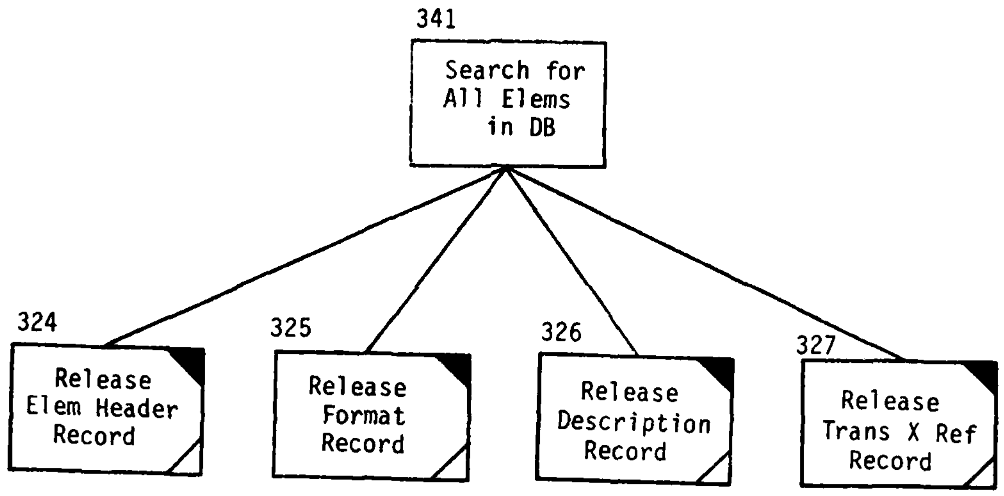
Functional Hierarchy Diagram


340
Element Type Search
341
Search for All Elems in DB
342
Search for Elems of a Proponent
343
Search for Elems in a # Range

Note: Do one of 341, 342, 343

CASTS	System/Program: ELERPT	Date: 4/80	Page: 7 of 11
Diagram ID: 341	Name: Data Element Dictionary	Description: Process I Report	

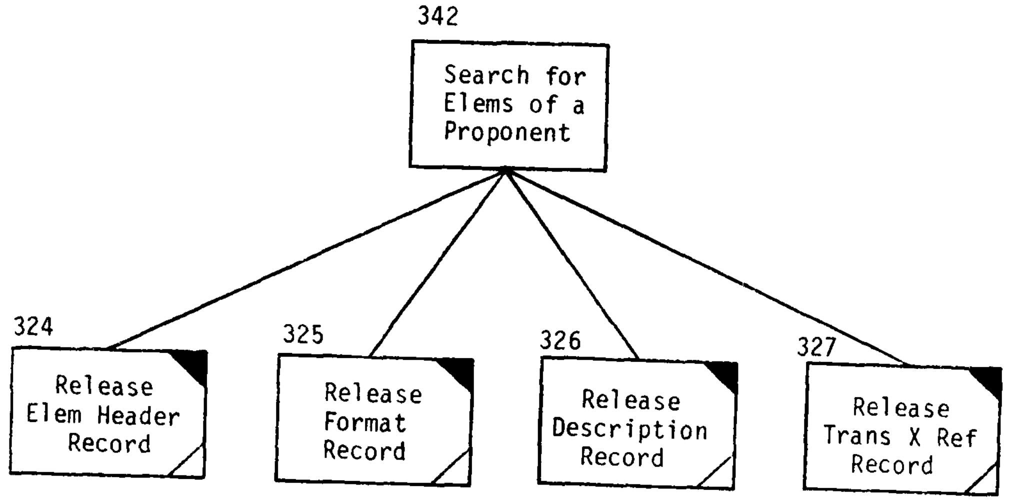
Functional Hierarchy Diagram


341
Search for All Elems in DB
324
Release Elem Header Record
325
Release Format Record
326
Release Description Record
327
Release Trans X Ref Record

CASTS

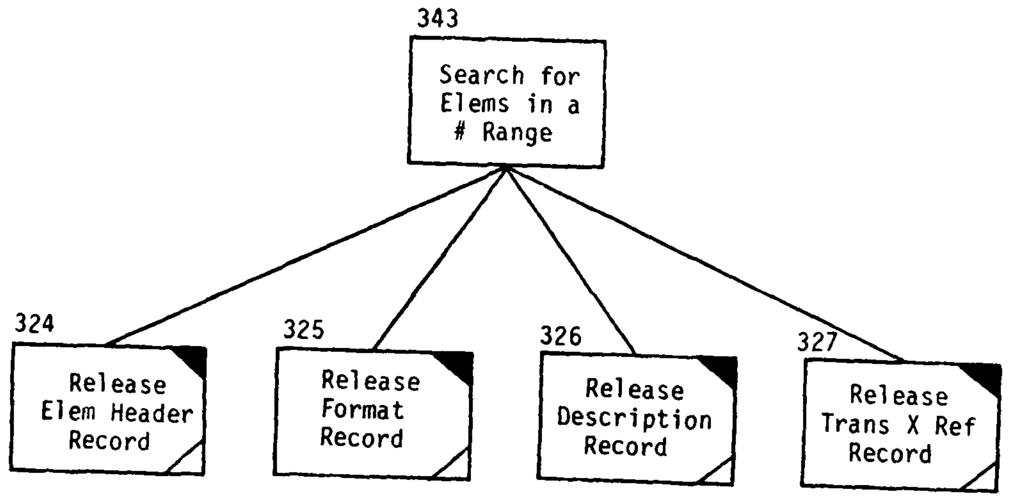
System/Program: ELERPT	Date: 4/80	Page: 8	of 11
Diagram ID: 342	Name: Data Element Dictionary	Description: Process I Report	

Functional Hierarchy Diagram


342
Search for Elems of a Proponent
324
Release Elem Header Record
325
Release Format Record
326
Release Description Record
327
Release Trans X Ref Record

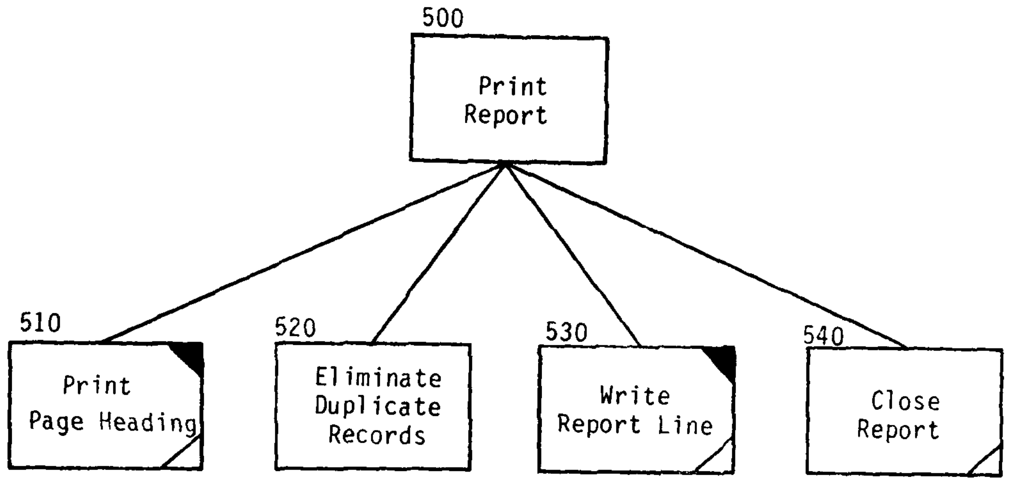
CASTS	System/Program: ELERPT	Date: 4/80	Page: 9 of 11
Diagram ID: 343	Name: Data Element Dictionary	Description: Process I Report	

Functional Hierarchy Diagram


343
Search for Elems in a # Range
324
Release Elem Header Record
325
Release Format Record
326
Release Description Record
327
Release Trans X Ref Record

CASTS	System/Program: ELERPT	Date: 4/80	Page: 10 of 11
Diagram ID: 500	Name: Data Element Dictionary	Description: Process I Report	

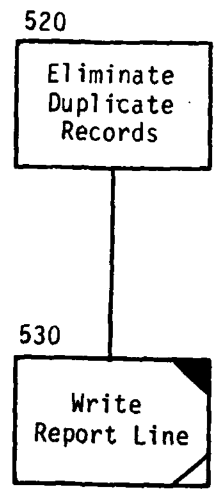
Functional Hierarchy Diagram



Note: 1) Do 520 or 530
2) 520 is only needed if trying to list all elements in an application, since each transaction must be checked for application name and all elements for that transaction are included in the sort, whether duplicates or not.

CASTS	System/Program: ELERPT	Date: 4/80	Page: 11 of 11
Diagram ID: 520	Name: Data Element Dictionary	Description: Process I Report	

Functional Hierarchy Diagram

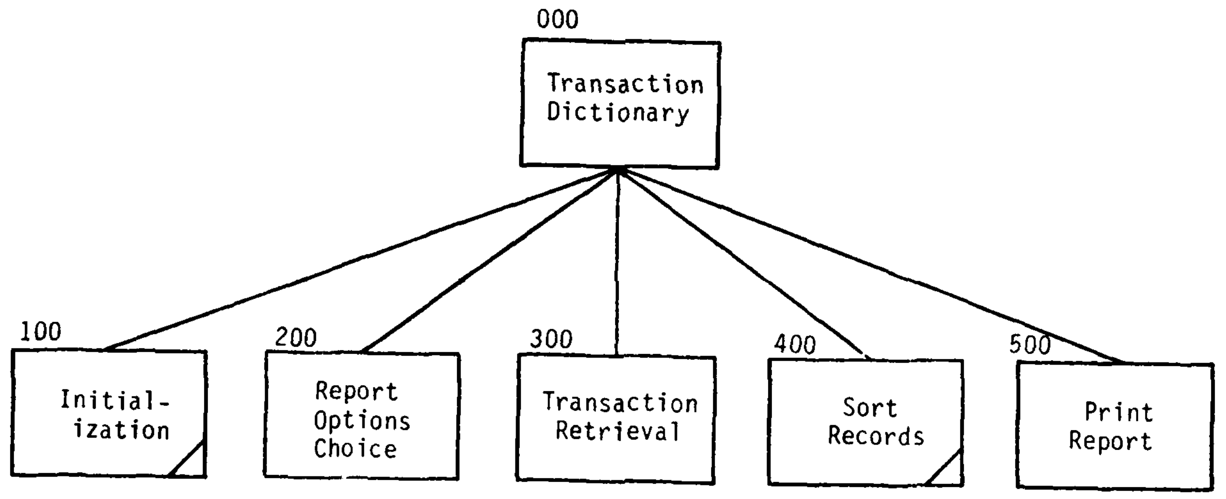


CASTS

System/Program: TRARPT **Date:** 5/80 **Page:** 1 **of** 6

Diagram ID: 000 **Name:** Transaction Dictionary **Description:** Process I Report

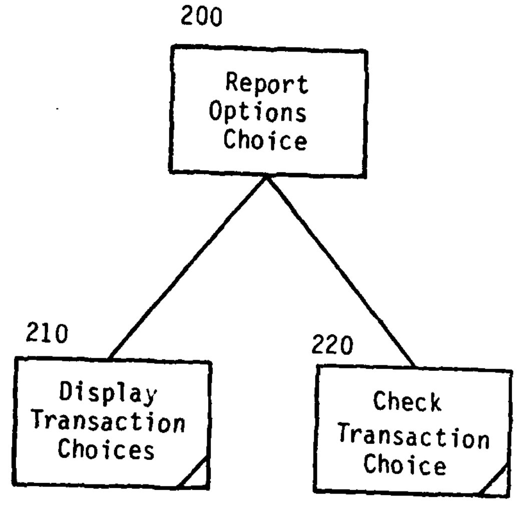
Functional Hierarchy Diagram



Note: *Transactions are reported in ascending alphabetical order by transaction name.*

CASTS	System/Program: TRARPT	Date: 5/80	Page: 2 of 6
Diagram ID: 200	Name: Transaction Dictionary	Description: Process I Report	

Functional Hierarchy Diagram

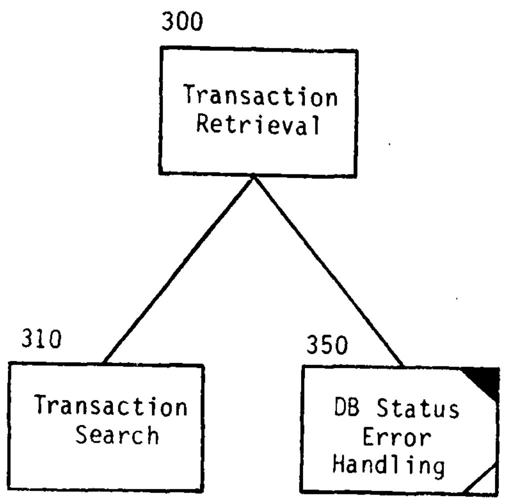


Transaction Selection Options:

1) All transactions in data base
2) Transactions in an application

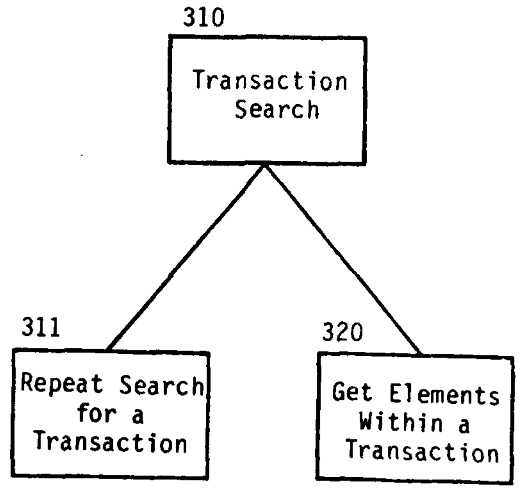
CASTS	System/Program: TRARPT	Date: 5/80	Page: 3 of 6
Diagram ID: 300	Name: Transaction Dictionary	Description: Process I Report	

Functional Hierarchy Diagram



CASTS	System/Program: TRARPT	Date: 5/80	Page: 4 of 6
Diagram ID: 310	Name: Transaction Dictionary	Description: Process I Report	

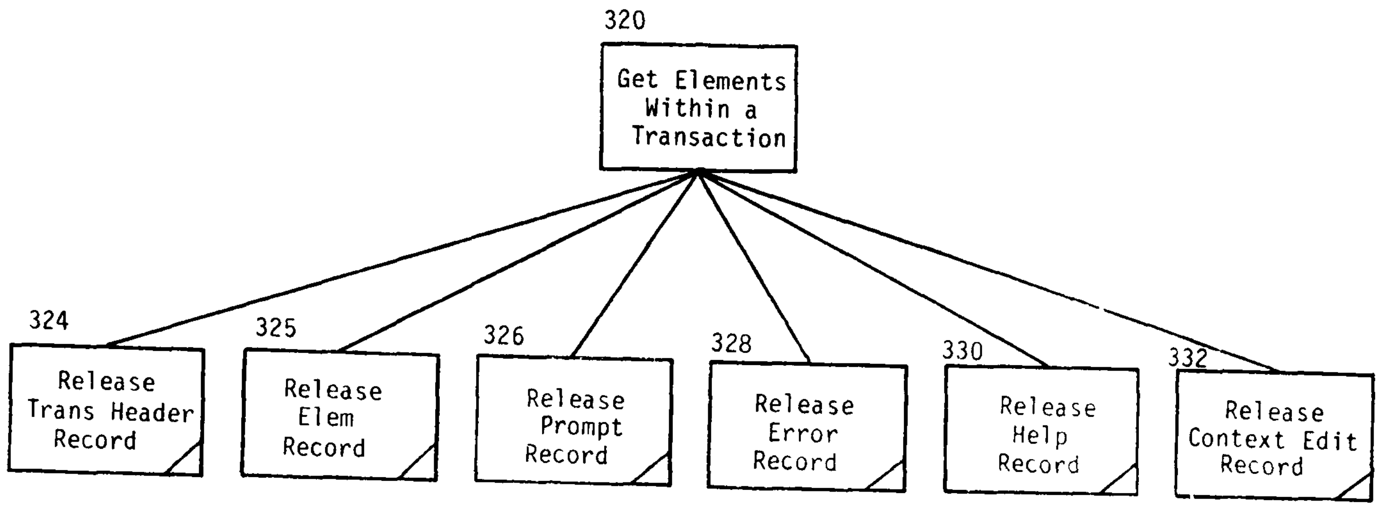
Functional Hierarchy Diagram



310
Transaction Search
311
Repeat Search for a Transaction
320
Get Elements Within a Transaction

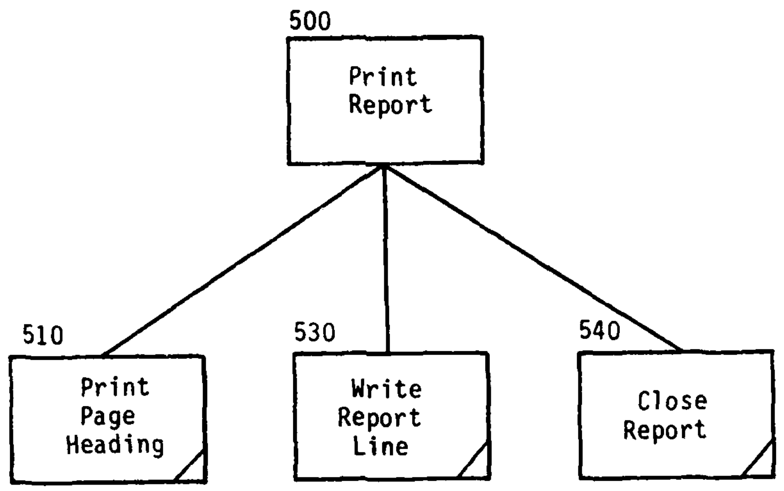
CASTS	System/Program: TRARPT	Date: 5/80	Page: 5 of 6
Diagram ID: 320	Name: Transaction Dictionary	Description: Process I Report	

Functional Hierarchy Diagram



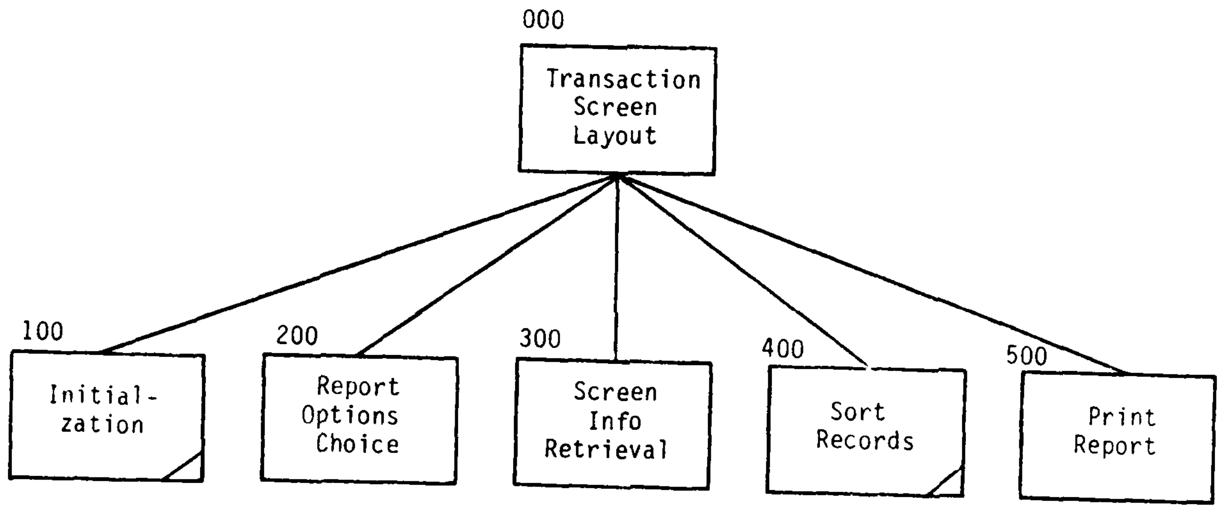
CASTS	System/Program: TRARPT		Date: 5/80	Page: 6	of 6
Diagram ID: 500	Name: Transaction Dictionary	Description: Process I Report			

Functional Hierarchy Diagram



CASTS	System/Program: SCRRPT	Date: 7/80	Page: 1 of 9
Diagram ID: 000	Name: Transaction Screen Layout	Description: Process I Report	

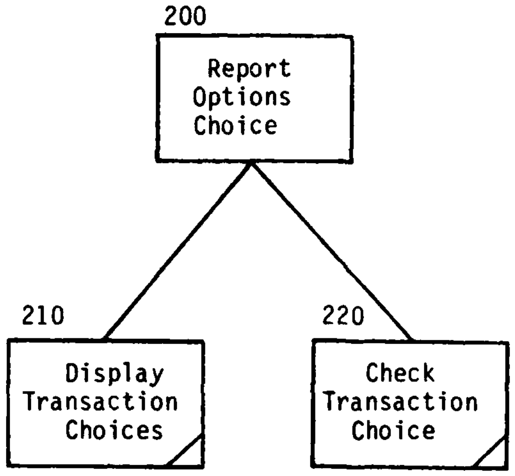
Functional Hierarchy Diagram



Note: Transactions are reported in ascending alphabetical order by transaction name.

CASTS	System/Program: SCRRPT	Date: 7/80	Page: 2 of 9
Diagram ID: 200	Name: Transaction Screen Layout	Description: Process I Report	

Functional Hierarchy Diagram

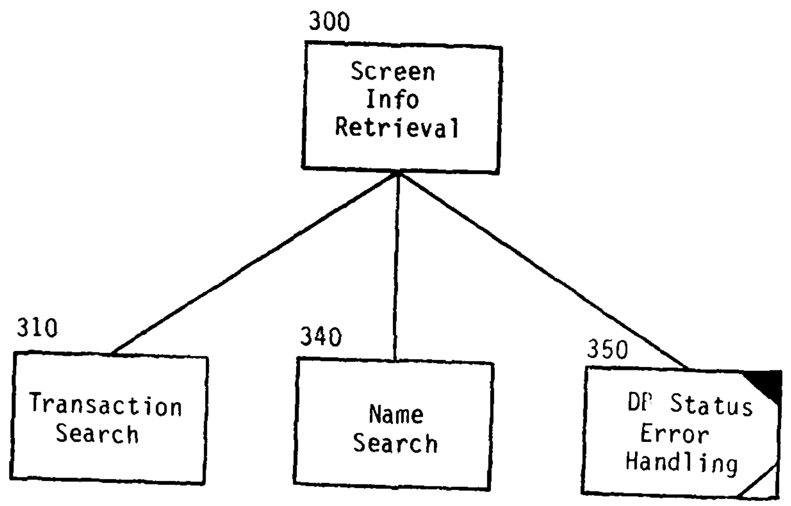


Transaction Selection Options:

1) All transactions in data base
2) Transactions in an application
3) One transaction, by name

CASTS	System/Program: SCRRPT	Date: 7/80	Page: 3 of 9
Diagram ID: 300	Name: Transaction Screen Layout	Description: Process I Report	

Functional Hierarchy Diagram

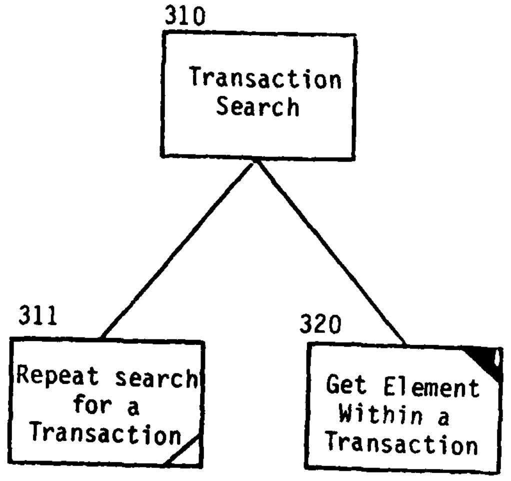


Note: 1) 310 searches for all transactions or those within a particular application.

2) 340 searches for exactly one transaction by transaction name.

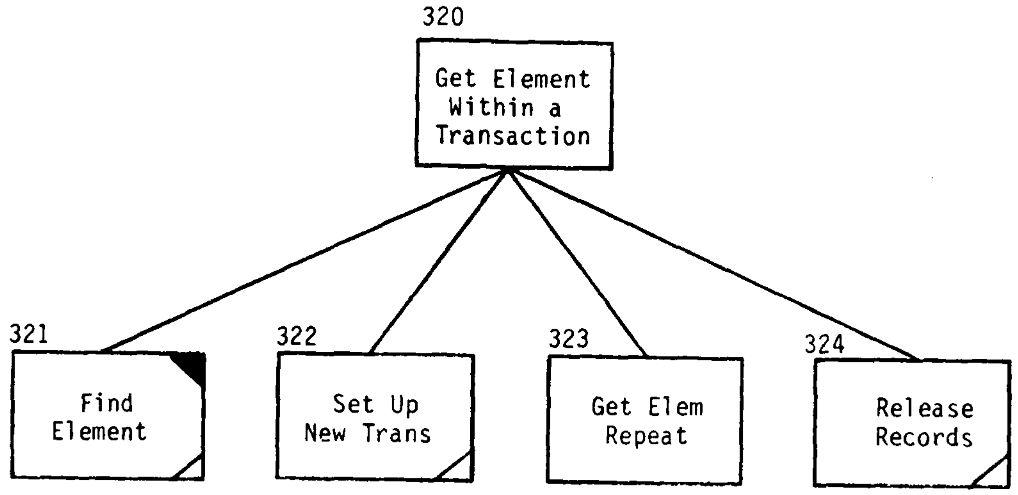
CASTS	System/Program: SCRRPT	Date: 7/80	Page: 4 of 9
Diagram ID: 310	Name: Transaction Screen Layout	Description: Process I Report	

Functional Hierarchy Diagram


310
Transaction Search
311
Repeat search for a Transaction
320
Get Element Within a Transaction

CASTS	System/Program: SCRRPT	Date: 7/80	Page: 5 of 9
Diagram ID: 320	Name: Transaction Screen Layout	Description: Process I Report	

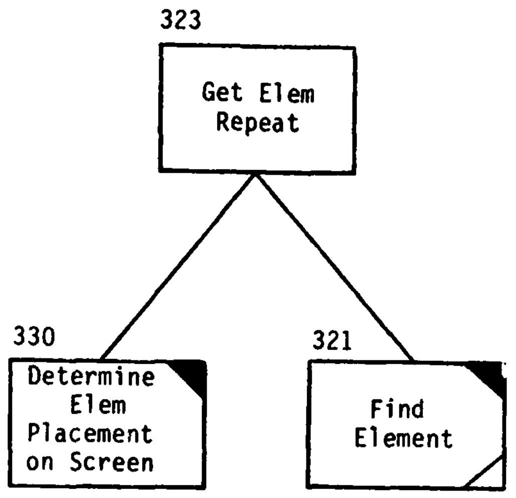
Functional Hierarchy Diagram



CASTS System/Program: SCRRPT Date: 7/80 Page: 6 of 9

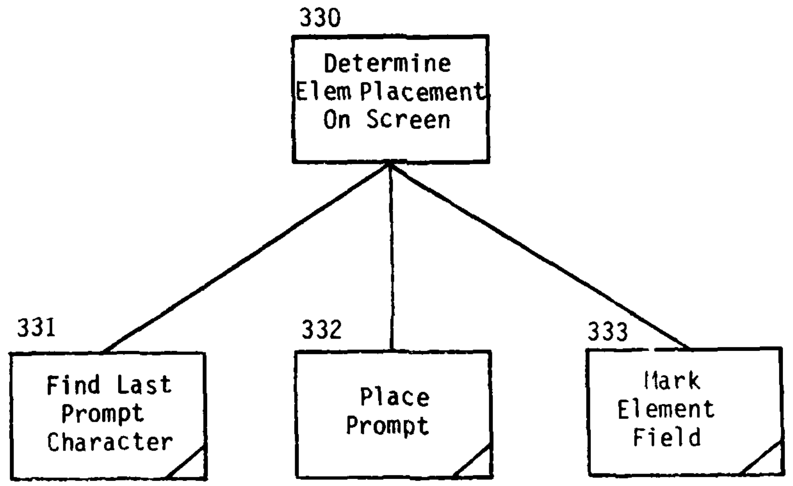
Diagram ID: 323 Name: Transaction Screen Layout Description: Process I Report

Functional Hierarchy Diagram



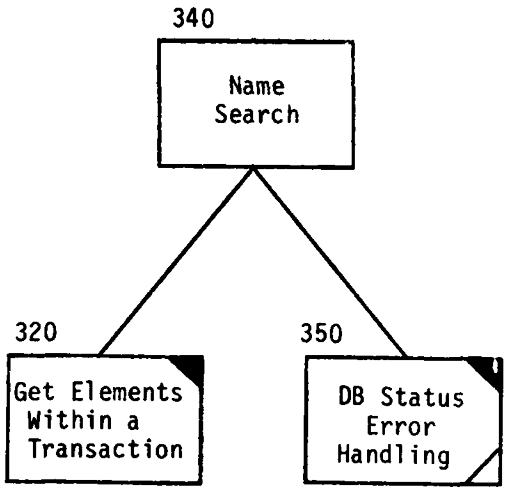
CASTS	System/Program: SCRRPT	Date: 7/80	Page: 7 of 9
Diagram ID: 330	Name: Transaction Screen Layout	Description: Process I Report	

Functional Hierarchy Diagram


330
Determine Elem Placement On Screen
331
Find Last Prompt Character
332
Place Prompt
333
Mark Element Field

CASTS	System/Program: SCRRPT	Date: 7/80	Page: 8 of 9
Diagram ID: 340	Name: Transaction Screen Layout	Description: Process I Report	

Functional Hierarchy Diagram

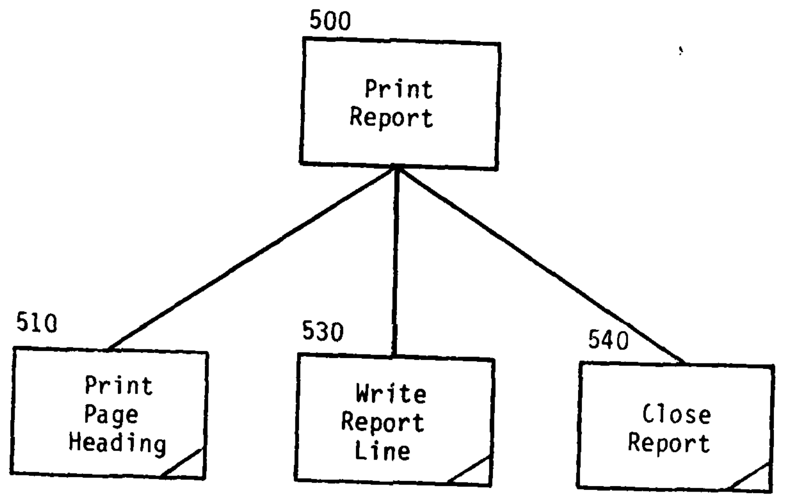


CASTS

System/Program: SCRRPT **Date:** 7/80 **Page:** 9 **of** 9

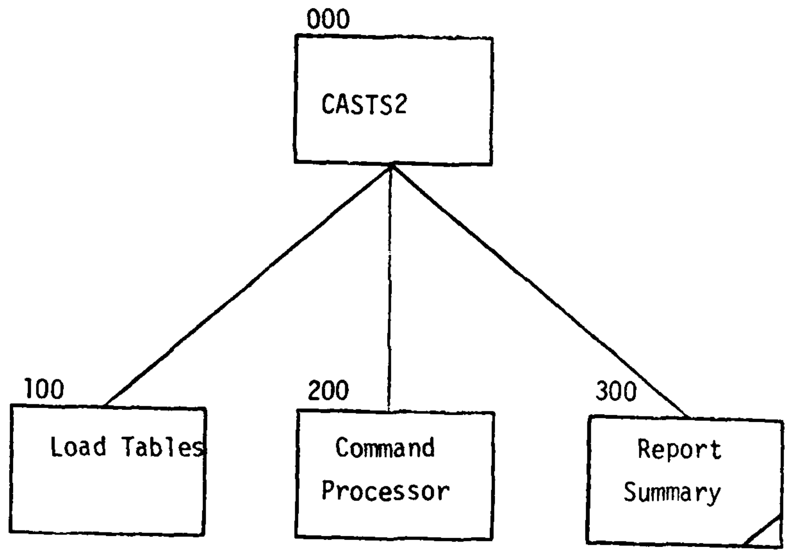
Diagram ID: 500 **Name:** Transaction Screen Layout **Description:** Process I Report

Functional Hierarchy Diagram



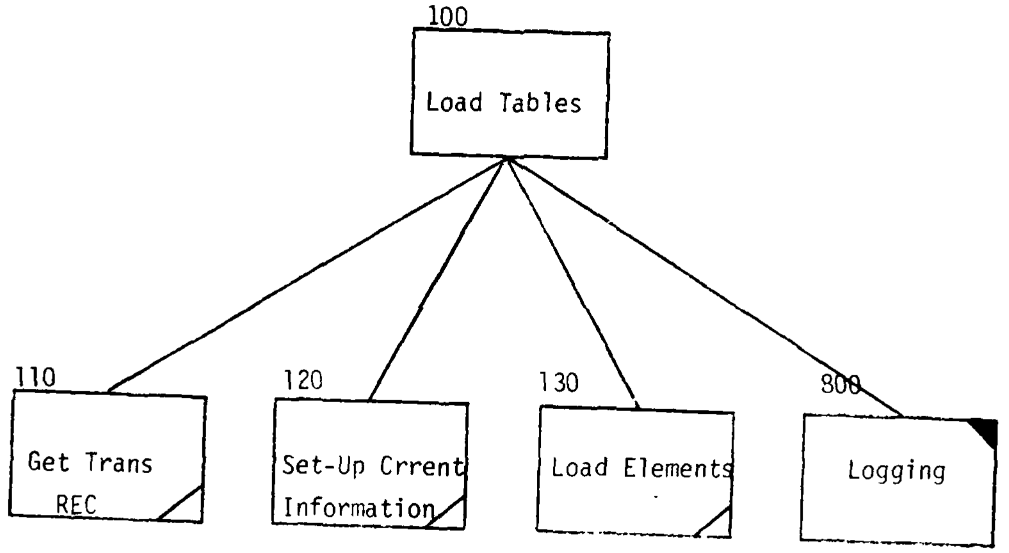
CASTS	System/Program: CASTS2	Date: 12/80	Page: 1 of 11
Diagram ID: 000	Name: CASTS Process II	Description: Simulation Process	

Functional Hierarchy Diagram



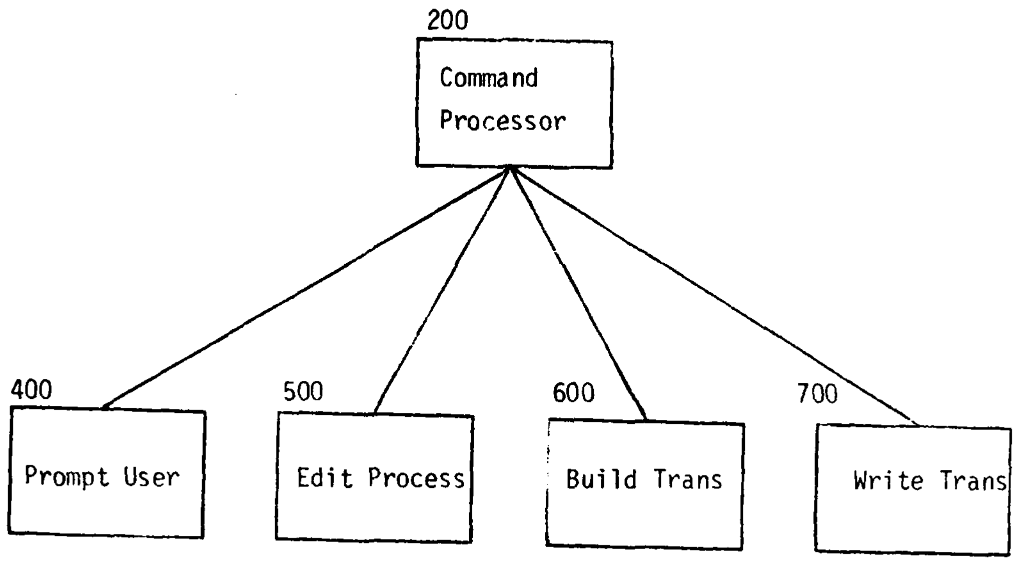
CASTS	System/Program: CASTS2	Date: 12/80	Page: 2 of 11
Diagram ID: 100	Name: CASTS Process II	Description: Simulation Process	

Functional Hierarchy Diagram



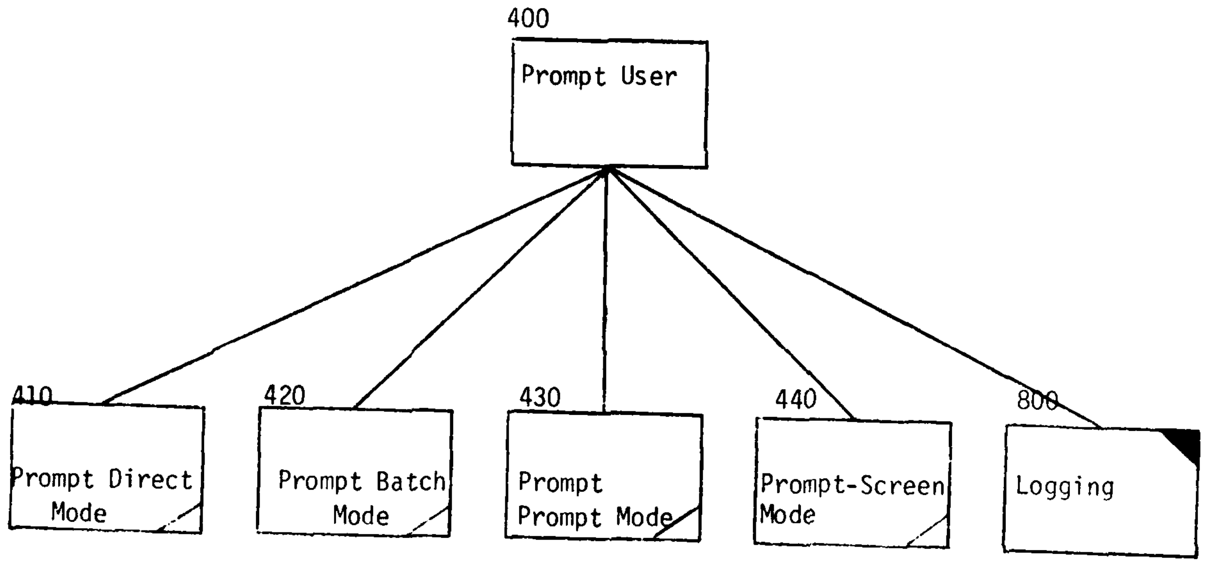
CASTS	System/Program: CASTS2	Date: 12/80	Page: 3 of 11
Diagram ID: 200	Name: CASTS Process II	Description: Simulation Process	

Functional Hierarchy Diagram



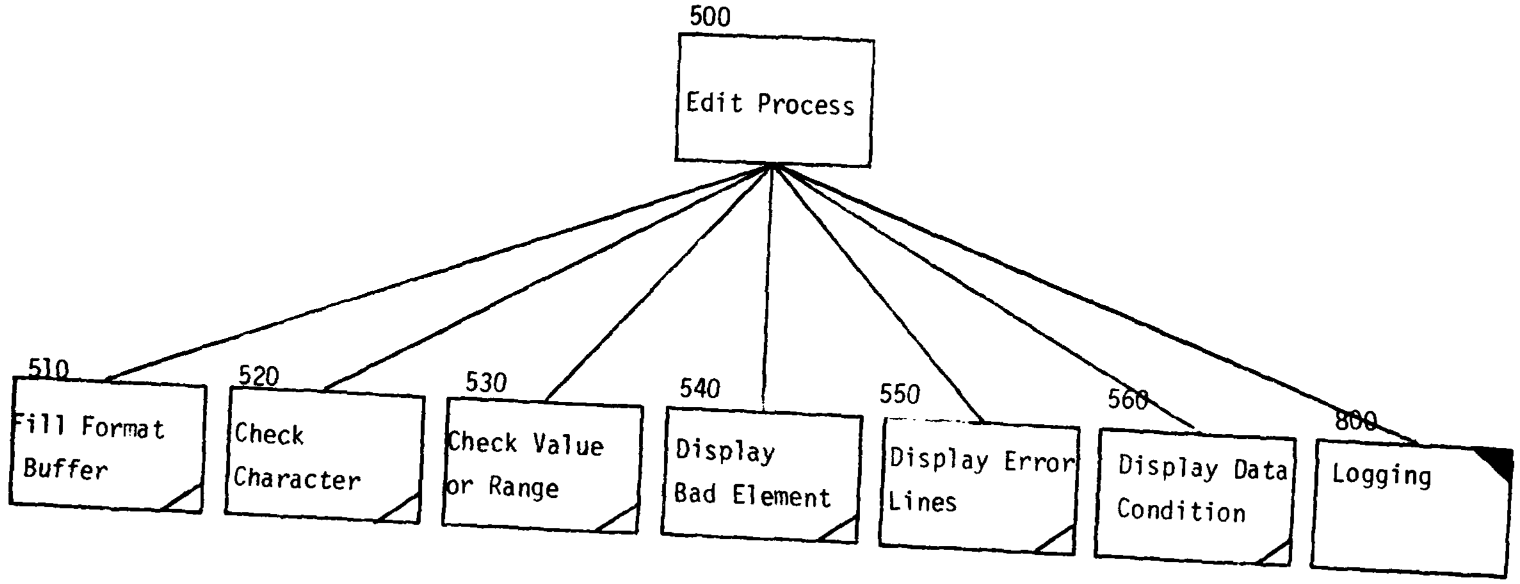
CASTS	System/Program: CASTS2	Date: 12/80	Page: 4 of 11
Diagram ID: 400	Name: CASTS Process II	Description: Simulation Process	

Functional Hierarchy Diagram



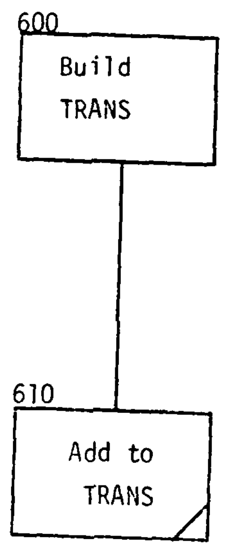
CASTS	System/Program: CASTS2		
Diagram ID: 500	Name: CASTS Process II	Date: 12/80	Page: 5 of 11
		Description: Simulation Process	

Functional Hierarchy Diagram



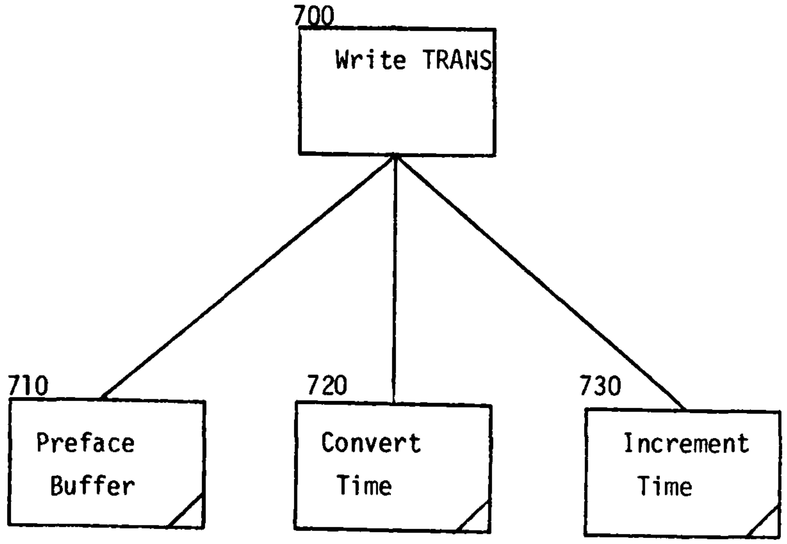
CASTS	System/Program: CASTS2	Date: 12/80	Page: 6 of 11
Diagram ID: 600	Name: CASTS Process II	Description: Simulation Process	

Functional Hierarchy Diagram



CASTS	System/Program: CASTS2	Date: 12/80	Page: 7 of 11
Diagram ID: 700	Name: CASTS Process II	Description: Simulation Process	

Functional Hierarchy Diagram

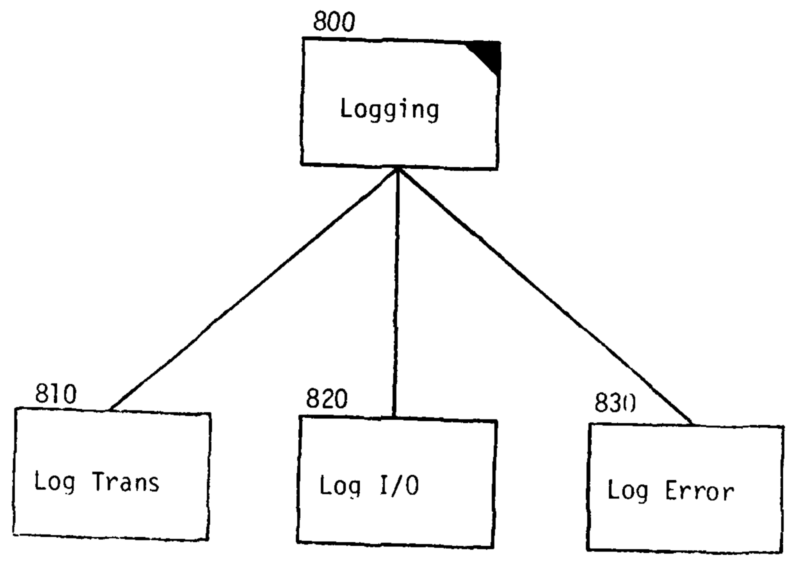


CASTS

System/Program: CASTS2 Date: 12/80 Page: 8 of 11

Diagram ID: 800 Name: CASTS Process II Description: Simulation Process

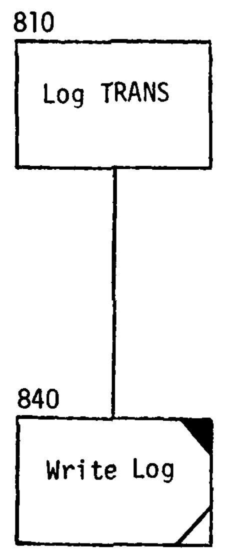
Functional Hierarchy Diagram



NOTE: 800-Logging is called by 100-Load-Table, 400-Prompt-User and 500-Edit-Process.

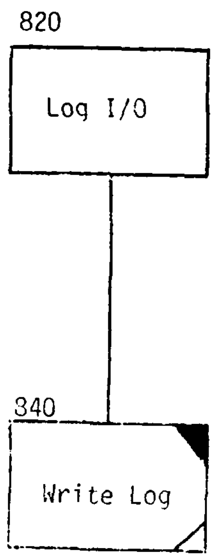
CASTS	System/Program: CASTS2	Date: 12/80	Page: 9 of 11
Diagram ID: 810	Name: CASTS Process II	Description: Simulation Process	

Functional Hierarchy Diagram



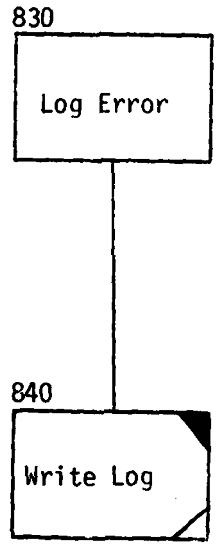
CASTS	System/Program: CASTS2	Date: 12/80	Page: 10 of 11
Diagram ID: 820	Name: CASTS Process II	Description: Simulation Process	

Functional Hierarchy Diagram



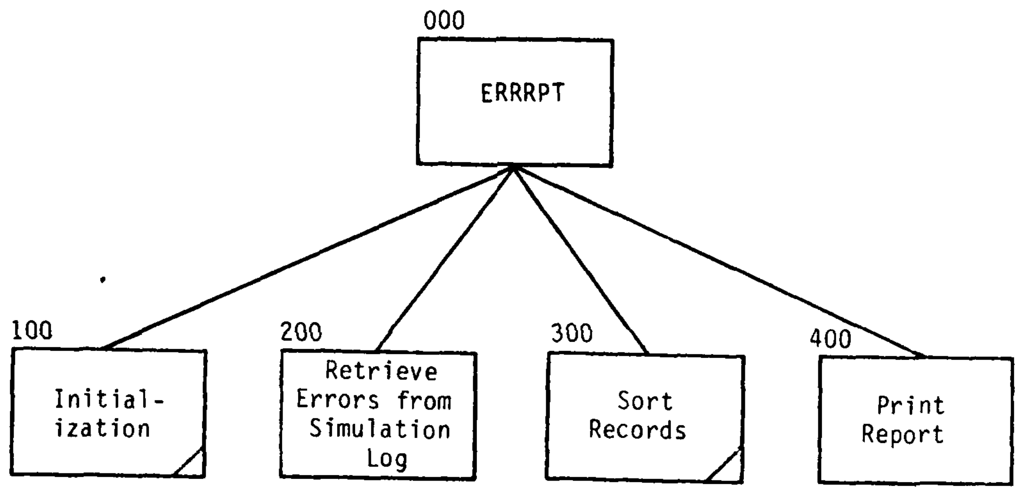
CASTS	System/Program: CASTS2	Date: 12/80	Page: 11 of 11
Diagram ID: 830	Name: CASTS Process II	Description: Simulation Process	

Functional Hierarchy Diagram



CASTS	System/Program: ERRRPT	Date: 9/80	Page: 1 of 5
Diagram ID: 000	Name: Terminal Session Error Report	Description: Process II Report	

Functional Hierarchy Diagram



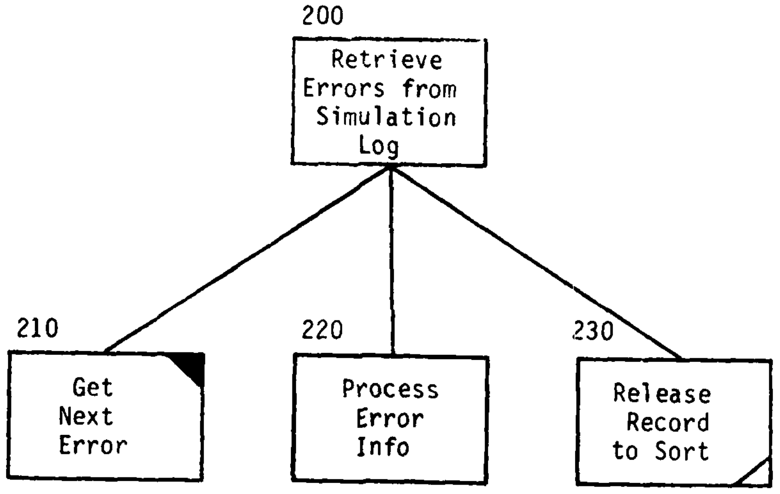
Note: Errors are reported according to the transactions within which they occur. Transaction types are arranged in ascending alphabetical order by transaction name. Individual occurrences of transactions within a transaction type are ordered by the transaction header time stamp.

CASTS

System/Program: ERRRPT **Date:** 9/80 **Page:** 2 **of** 5

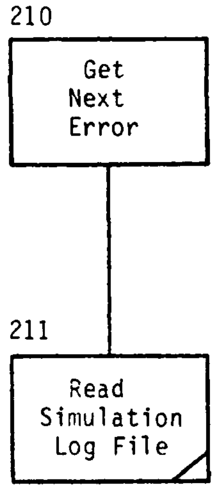
Diagram ID: 200 **Name:** Terminal Session Error Report **Description:** Process II Report

Functional Hierarchy Diagram



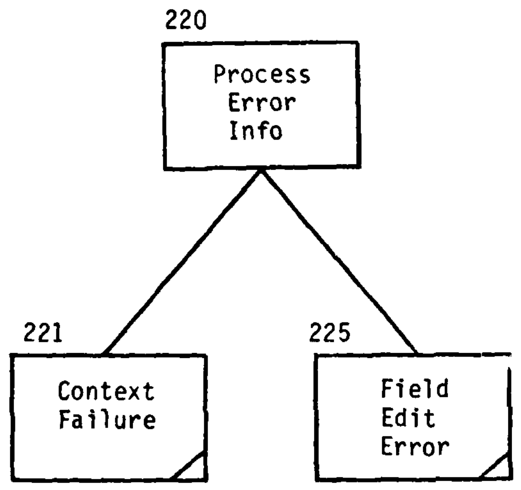
CASTS	System/Program: ERRRPT	Date: 9/80	Page: 3 of 5
Diagram ID: 210	Name: Terminal Session Error Report	Description: Process II Report	

Functional Hierarchy Diagram



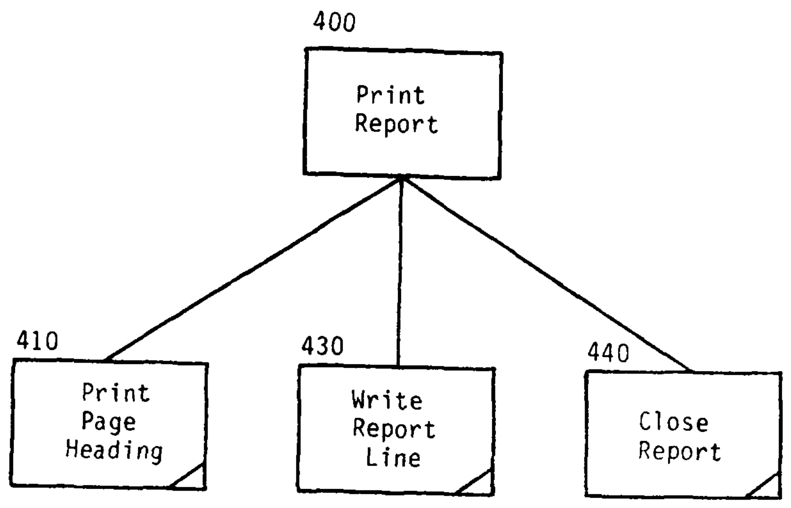
CASTS	System/Program: ERRRPT	Date: 9/80	Page: 4 of 5
Diagram ID: 220	Name: Terminal Session Error Report	Description: Process II Report	

Functional Hierarchy Diagram



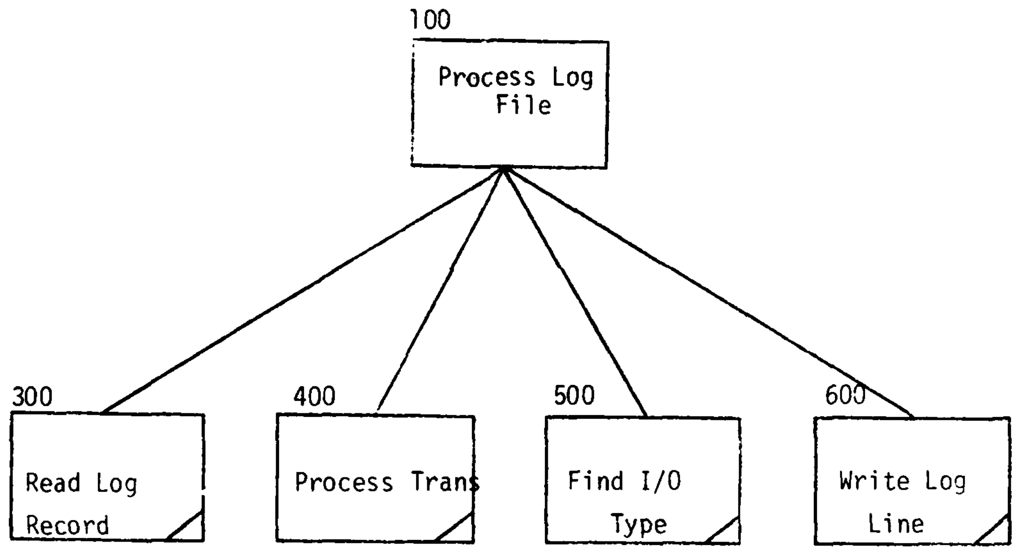
CASTS	System/Program: ERRRPT	Date: 9/80	Page: 5 of 5
Diagram ID: 400	Name: Terminal Session Error Report	Description: Process II Report	

Functional Hierarchy Diagram



CASTS	System/Program: LOGRPT	Date: 12/80 Page: 2 of 2
Diagram ID: 100	Name: Interaction Log Report	Description: Process II Report

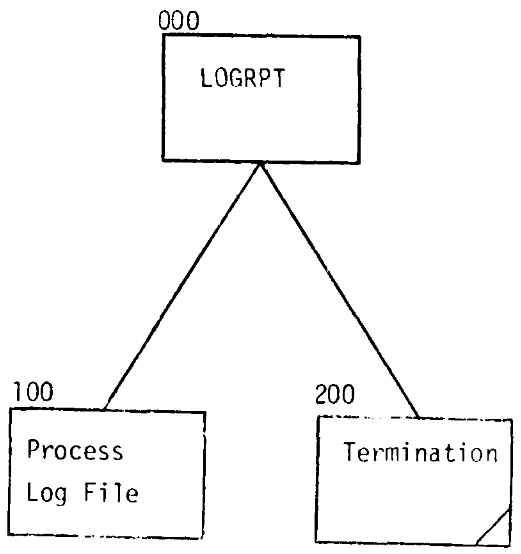
Functional Hierarchy Diagram



System/Program: LOGRPT Date: 12/80 Page 1 of 2

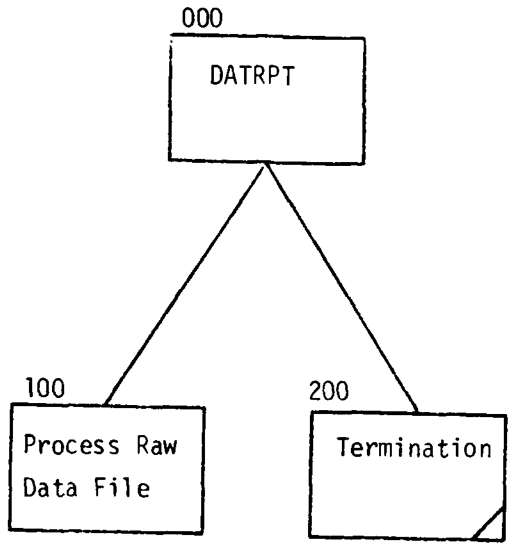
Name: Interaction Log Report Description: Process II Report

Functional Hierarchy Diagram



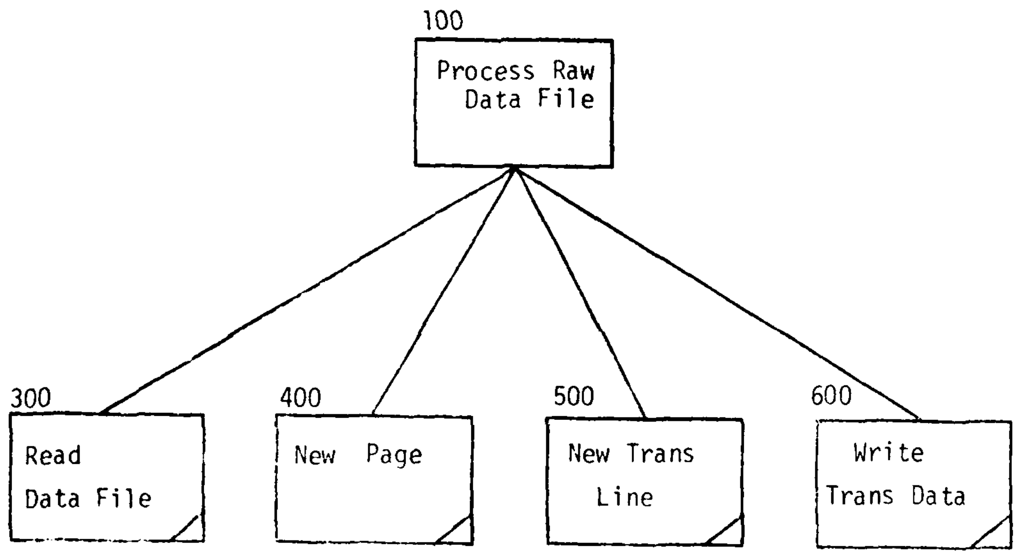
CASTS	System/Program: DATRPT	Date: 12/80	Page: 1 of 2
Diagram ID: 000	Name: Application Raw Data Report	Description: Process II Report	

Functional Hierarchy Diagram



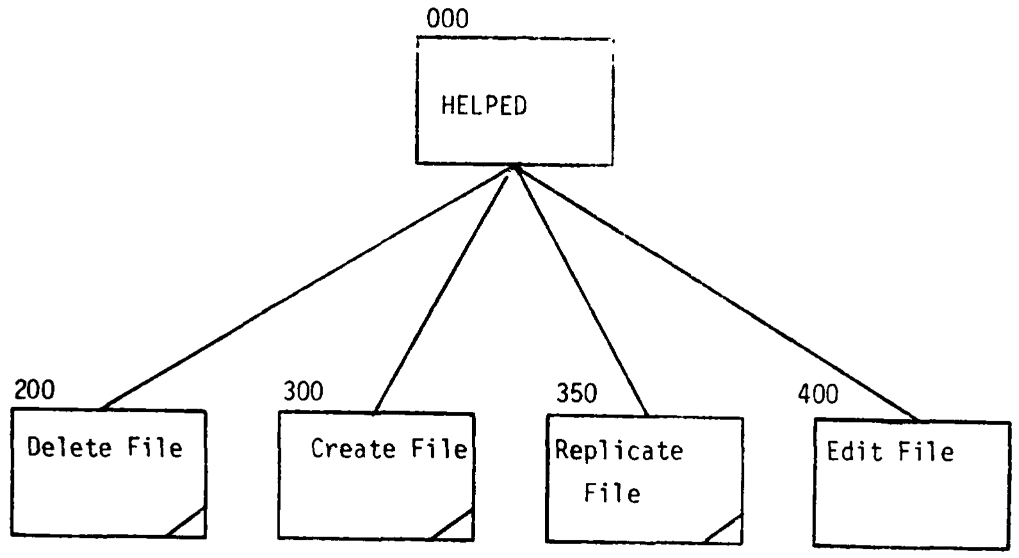
CASTS	System/Program: DATRPT	Date: 12/80	Page: 2 of 2
Diagram ID: 100	Name: Application Raw Data Report	Description: Process II Report	

Functional Hierarchy Diagram



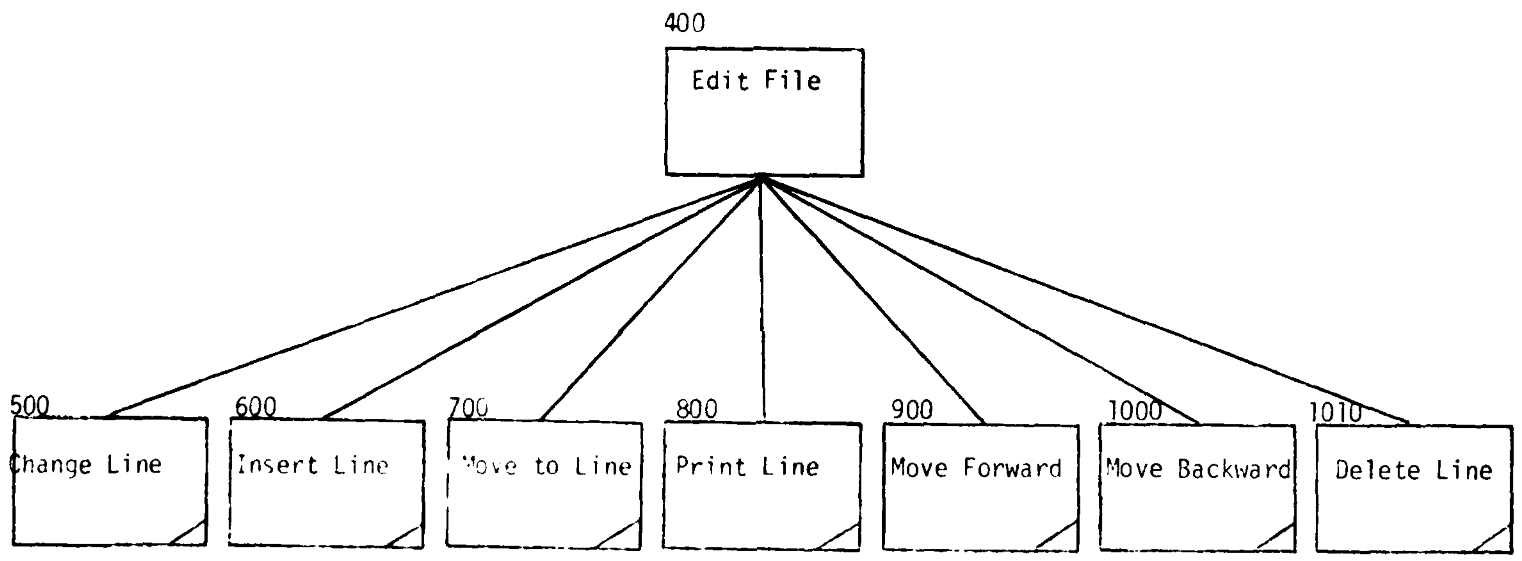
CASTS	System/Program: HELPED	Date: 10/80	Page: 1 of 2
Diagram ID: 000	Name: Help Messages Text Editor	Description: CASTS System Help	

Functional Hierarchy Diagram



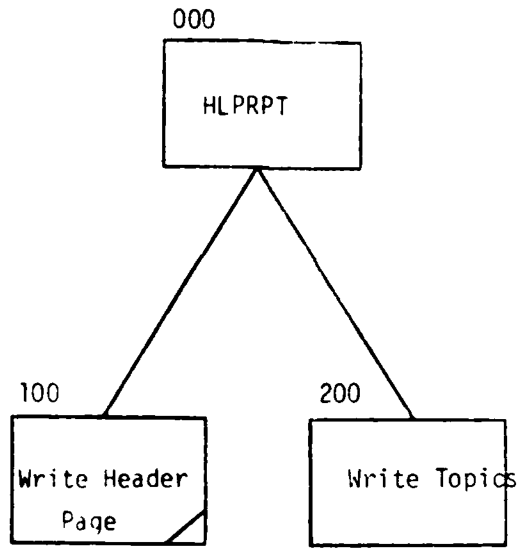
CASTS	System/Program: HELPED	Date: 10/80	Page: 2 of 2
Diagram ID: 400	Name: Help Messages Text Editor	Description: CASTS System Help	

Functional Hierarchy Diagram



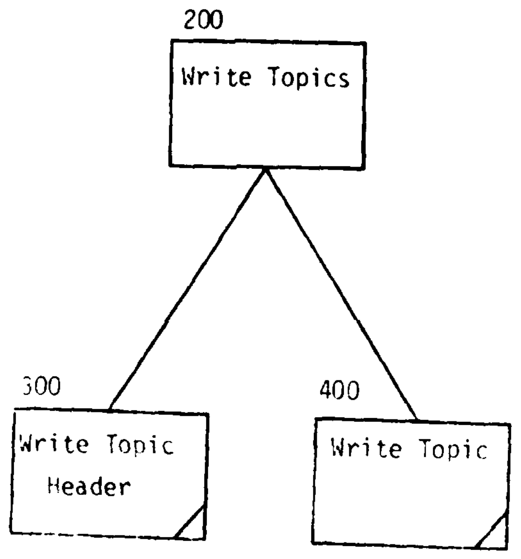
CASTS	System/Program: HLPRPT	Date: 10/80	Page: 1 of 2
Diagram ID: 000	Name: Help Messages Report	Description: CASTS System Help	

Functional Hierarchy Diagram



CASTS	System/Program: HLPRPT	Date: 10/80	Page: 2 of 2
Diagram ID: 200	Name: Help Messages Report	Description: CASTS System Help	

Functional Hierarchy Diagram



FILMED

3-8