510.84
Il63c
no.36

Engin.

CAC Document No.36

214-13

NARIS

Data Insertion Manual

Center for Advanced Computation
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

August 15, 1972

This work was supported in part by a grant from the Ford Foundation.

ACKNOWLEDGEMENT

NARIS is being developed by the Center for Advanced Computation of the University of Illinois at Urbana-Champaign in cooperation with the Northeast Illinois Natural Resource Service Center, Lisle, Illinois, and with the Northeast Illinois Planning Commission, Chicago, Illinois, and is sponsored in part by a grant from the Ford Foundation.

The NARIS computer system was designed and implemented by:

Peter A. Alsberg
Thomas S. Asbury
Calvin C. Corbin
John P. Day
Thomas P. L. Dowell
James A. Gast
William D. McTeer
Jean-Michel Michl
Stewart A. Schuster

Principal author: Calvin C. Corbin

Contributors: William D. McTeer
Peter A. Alsberg

ABSTRACT

This manual is addressed to anyone attempting to implement NARIS - other than at the University of Illinois.

This manual is concerned with two functions: transmitting the data from coding form to the NARIS data base and altering or adjusting programs which process the data during input.

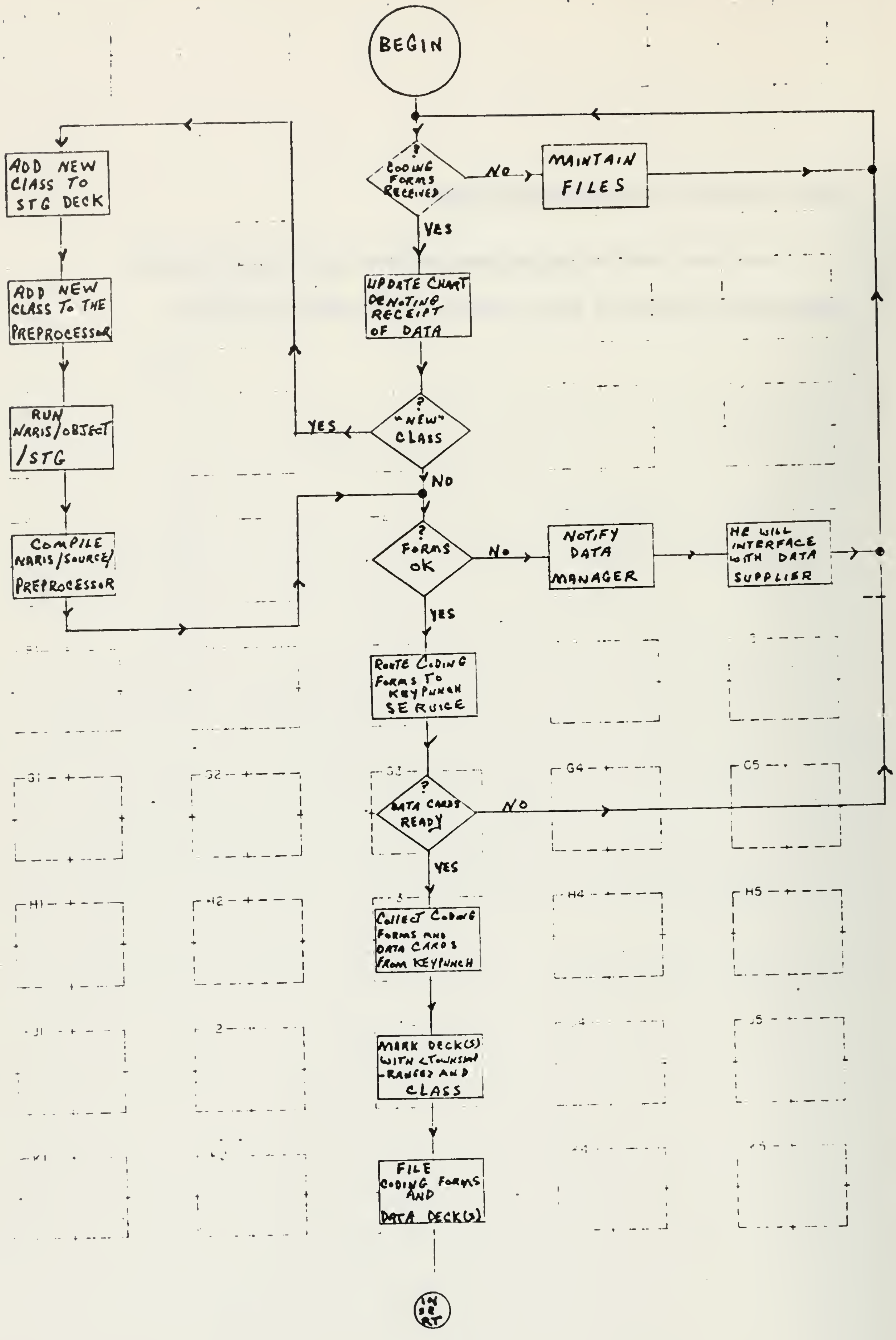
Part I of the manual is concerned with the mechanics of inserting data into NARIS when the Data Class is resident to the system and the filing and cataloging of the data. Part II comprises: adding a new Class of data to NARIS, and changing the Data Elements of a Class. Part III deals with adding geographic areas to the Data base. Part IV deals with data structure revisions.

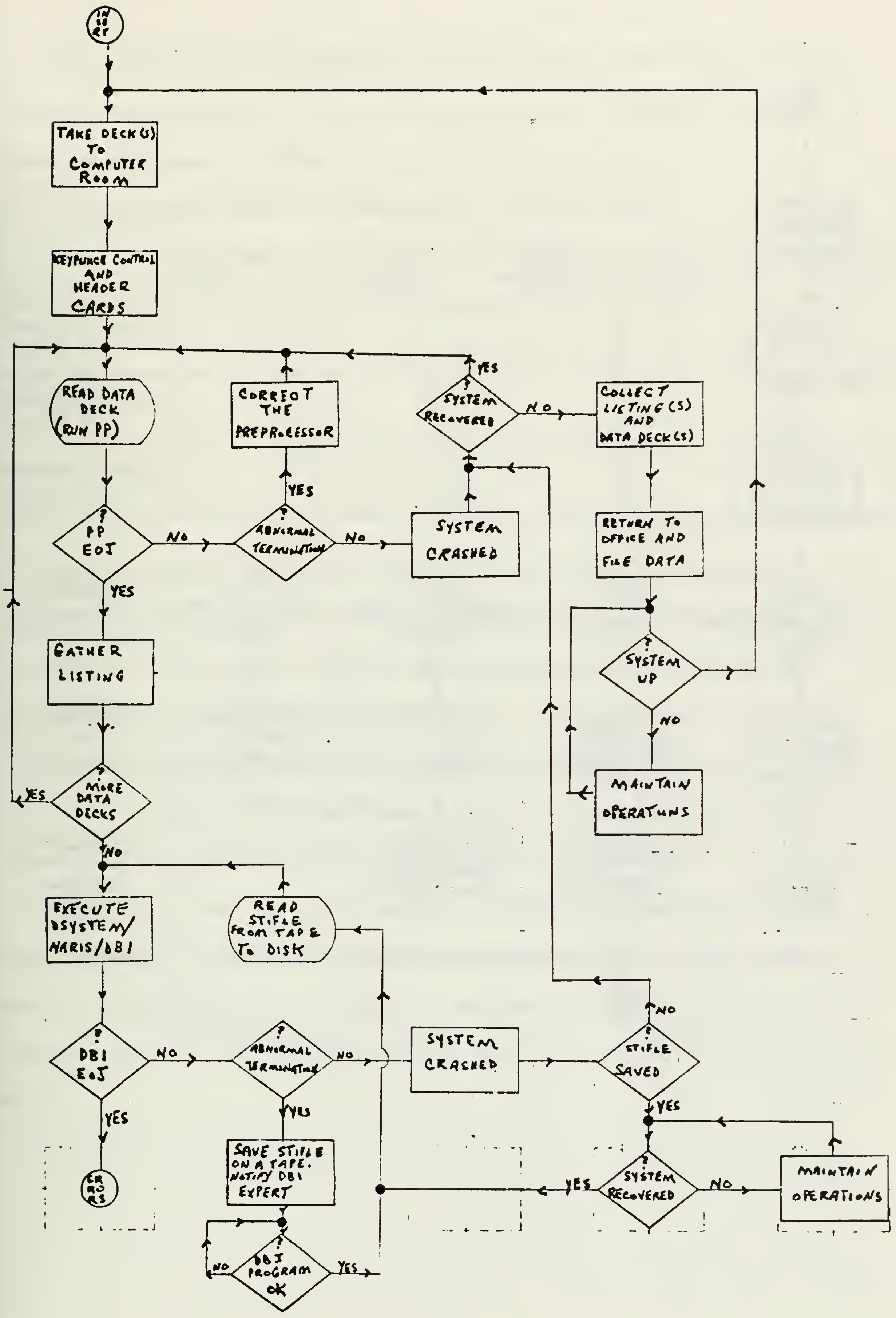
For purposes of putting data in NARIS, it is recommended that the NARIS Data Coding Manual be read.

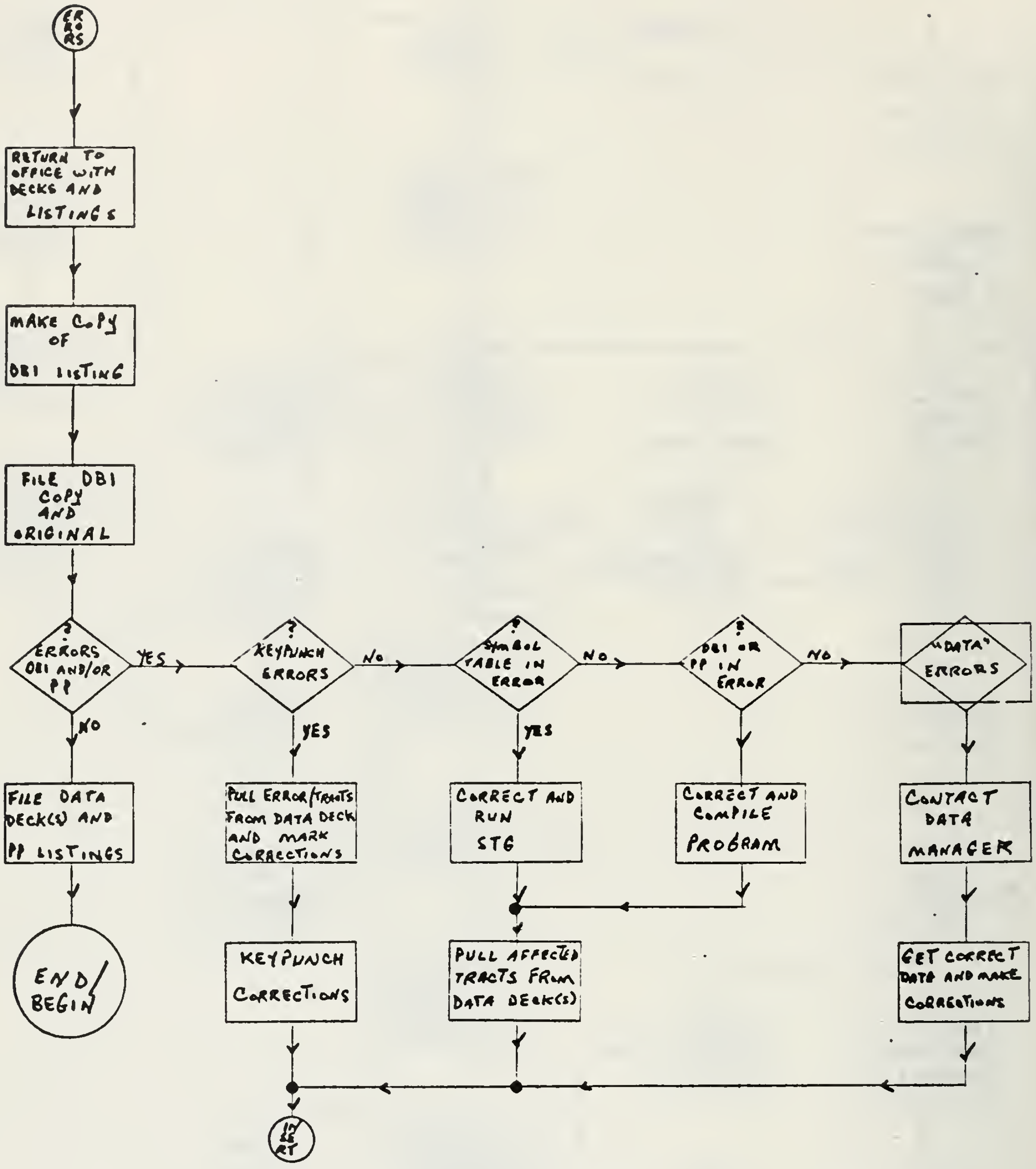
TABLE OF CONTENTS

I. Data Insertion of an Existing Class

The flow chart which follows depicts quite explicitly the procedures which have been used to insert data into NARIS.


BEGIN
? CODING FORMS RECEIVED
NO
MAINTAIN FILES
YES
UPDATE CHART DENOTING RECEIPT OF DATA
? "NEW" CLASS
YES
ADD NEW CLASS TO STG DECK
ADD NEW CLASS TO THE PREPROCESSOR
RUN NARIS/OBJECT/STG
COMPILE NARIS/SOURCE/PREPROCESSOR
NO
? FORMS OK
NO
NOTIFY DATA MANAGER
HE WILL INTERFACE WITH DATA SUPPLIER
YES
ROUTE CODING FORMS TO KEYPUNCH SERVICE
? DATA CARDS READY
NO
YES
COLLECT CODING FORMS AND DATA CARDS FROM KEYPUNCH
MARK DECK(S) WITH <TOWNSHIP-RANGE> AND CLASS
FILE CODING FORMS AND DATA DECK(S)
INSERT


IN SE RT
TAKE DECK(S) TO COMPUTER ROOM
KEYPUNCH CONTROL AND HEADER CARDS
READ DATA DECK (RUN PP)
CORRECT THE PREPROCESSOR
YES
SYSTEM RECOVERED
NO
COLLECT LISTING(S) AND DATA DECK(S)
YES
PP EOJ
NO
ABNORMAL TERMINATION
NO
SYSTEM CRASHED
RETURN TO OFFICE AND FILE DATA
YES
GATHER LISTING
SYSTEM UP
NO
MAINTAIN OPERATIONS
YES
MORE DATA DECKS
NO
READ STIFLE FROM TAPE TO DISK
EXECUTE DSYSTEM/NARIS/DBI
NO
DBI EOJ
NO
ABNORMAL TERMINATION
NO
SYSTEM CRASHED
STIFLE SAVED
YES
YES
YES
SAVE STIFLE ON A TAPE. NOTIFY DBI EXPERT
YES
SYSTEM RECOVERED
NO
MAINTAIN OPERATIONS
ER RO RS
NO
DBI PROGRAM OK
YES


ERRORS
RETURN TO OFFICE WITH DECKS AND LISTINGS
MAKE COPY OF DBI LISTING
FILE DBI COPY AND ORIGINAL
? ERRORS DBI AND/OR PP
YES
NO
FILE DATA DECK(S) AND PP LISTINGS
END/BEGIN
? KEYPUNCH ERRORS
NO
YES
PULL ERROR/TRACTS FROM DATA DECK AND MARK CORRECTIONS
KEYPUNCH CORRECTIONS
? SYMBOL TABLE IN ERROR
NO
YES
CORRECT AND RUN STG
PULL AFFECTED TRACTS FROM DATA DECK(S)
? DBI OR PP IN ERROR
NO
CORRECT AND COMPILE PROGRAM
"DATA" ERRORS
CONTACT DATA MANAGER
GET CORRECT DATA AND MAKE CORRECTIONS
INSERT

The "data insertion" process is initiated upon the receipt of data from the data supplier. Data is entered one Data Class per surveyed township and range at a time.

A "Data Status Chart" is maintained. It looks like:

	Data Class SOIL	Data Class FORESTRY	Data Class WATERSHED	Data Class PLANTATION	Data Class GEOWASTE
Township-Range T44N R05E					
Township-Range T44N R06E					
Township-Range T44N R07E					

The Data Status Chart is a matrix (rows by township-range and columns by data class(es)), the boxes of which reflect the status of the data for a Data Class within a township range. The following information is recorded in these units:

. whether or not the data has been received for input;

. whether or not the data has been inserted into NARIS; and

. whether or not the inserted data is "error-free".

The appropriate box in the matrix is updated to show that data has been received. If the data came in on coding forms, the coding forms are sent to a keypunch service. When the data deck(s) are received from the keypunch service, the following control cards are prepared for each data deck:

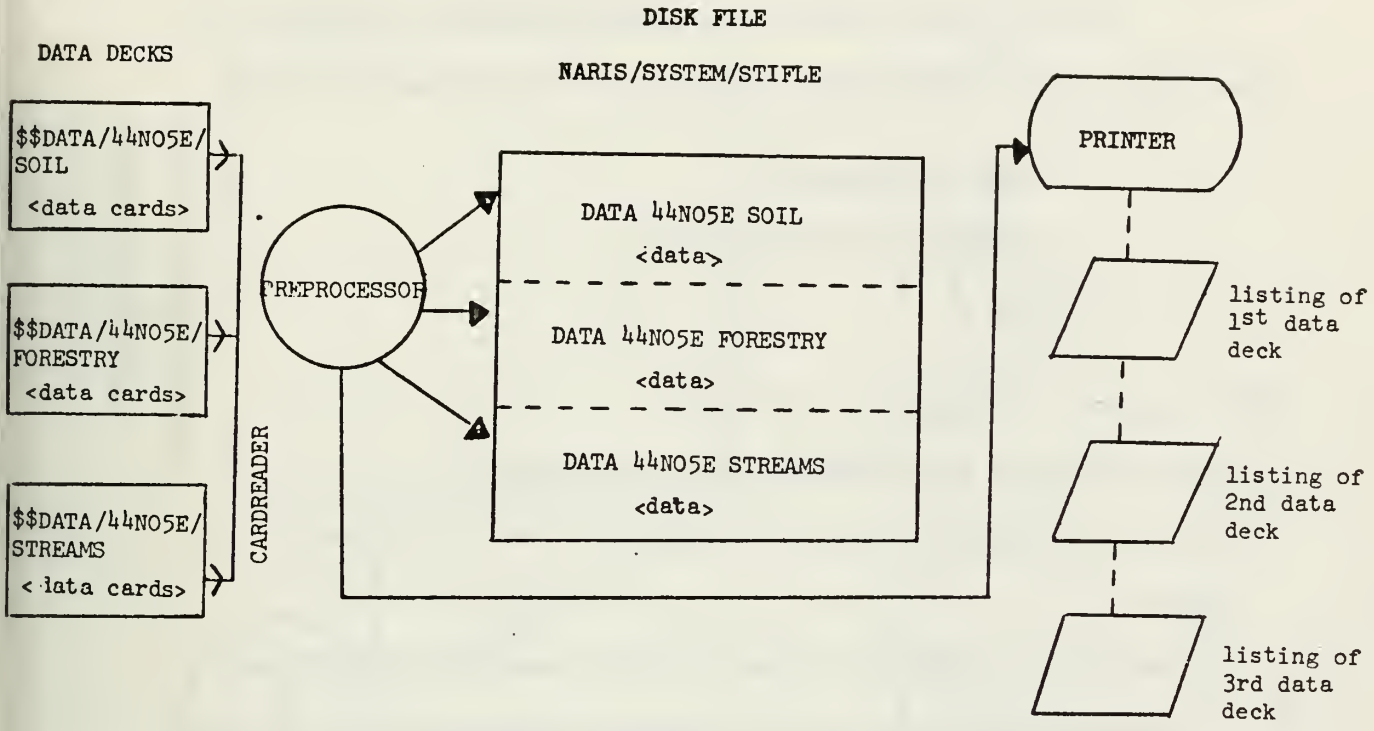
```
?USER = CACNNARIS

?EXECUTE DSYSTEM/NARIS/PREPROCESSOR

?VALUE = 1 (optional control card)

?DATA FØ/CDS

$$DATA/<township-range>/<class name>  }  Only one of these
                                      }  program control or
$$UPDT/<township-range>/<class name>  }  <header> cards is use
                                      }  for each data deck.
$$DLTE/<township-range>/<class name>  }

             <data deck>

?END
```

The optional "?VALUE = 1" card is used to suspend checking of the township and range values which appear on each card in the data deck. When the "?VALUE = 1" card is used the Preprocessor uses the <township-range> from the program control card (cards beginning with a "$$") and disregards the <township-range> column on the data cards.

Having prepared the data decks with the control cards, a data deck is submitted to be "read" through the cardreader - initiating the program, DSYSTEM/NARIS/PREPROCESSOR. If the Symbol Table is not on disk, the Preprocessor will ask that a tape be mounted on a tape unit and proceed to copy the Symbol Table to disk.

The Preprocessor checks the data on the data cards: if the data is correct, the Preprocessor puts it in a disk file (NARIS/SYSTEM/STIFLE) on the computer; if not correct, an error message is generated in an attempt to denote the type of error.

Following the Preprocessor's completion of a data deck, another data deck is "read"; this process terminates when there are no more data decks to be read. Figure 1 illustrates the creation of the disk file, STIFLE (Standard Intermediate File), by the Preprocessor.



Creation of a STIFLE by the Preprocessor

Figure 1

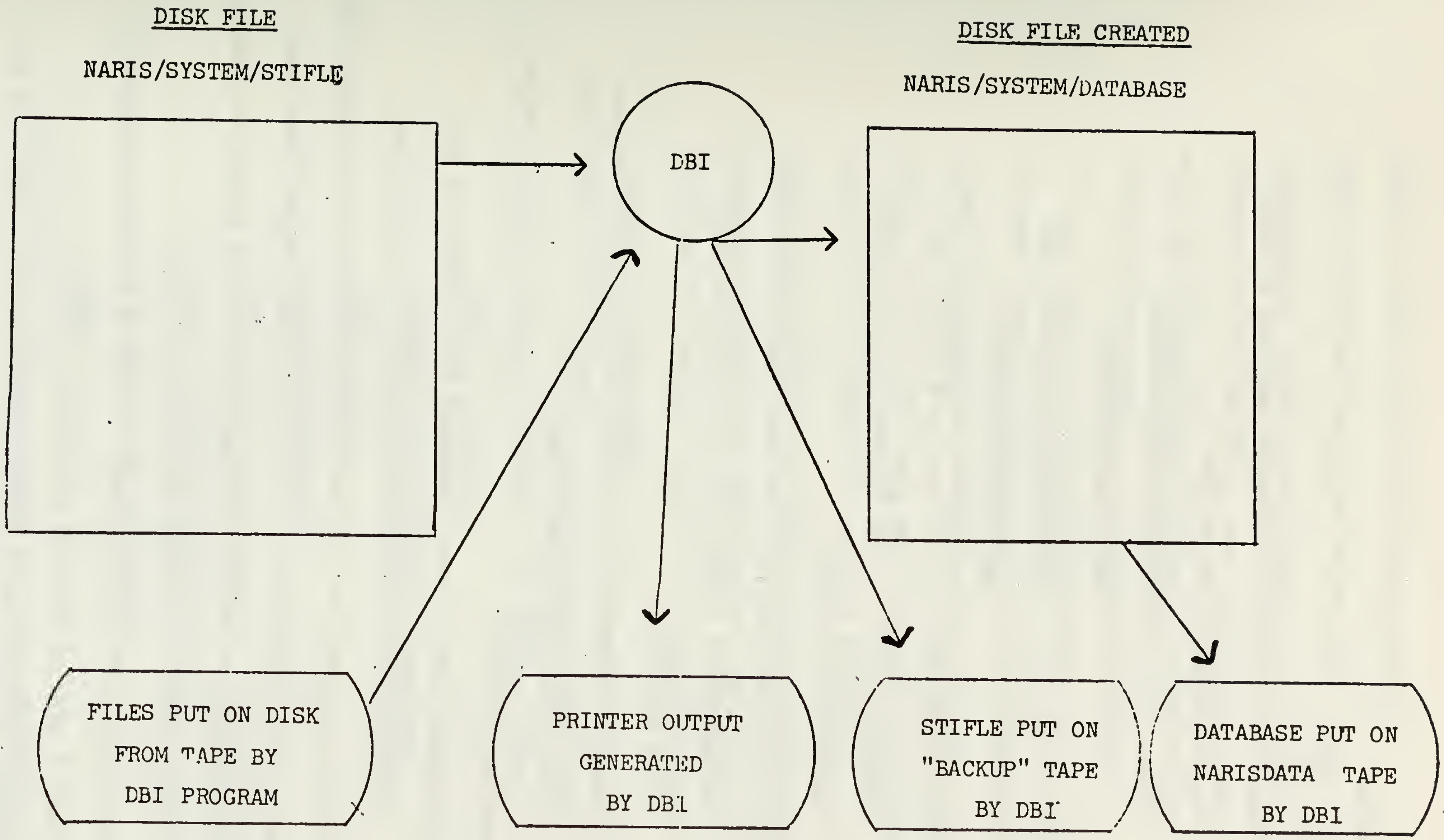
The Preprocessor produces a listing for each data deck. Error messages are printed on the listing immediately after the line on which the data card containing the error was printed unless the error was a tract error (not pertaining to one data card) in which case the error message is printed after the last line which pertained to the 1/4 1/4 section.

Once the data deck(s) have been "executed", the Data Base Insertion Program (DBI) is run by having the following cards read:

```
?USER = CACNNARIS
?EXECUTE DSYSTEM/NARIS/DBI
?END
```

This program will perform three tasks:

1) it will insert data into the NARIS data base; and
2) it will cause a listing to be printed on the Printer which will
   - denote the individual class/townships dealt with by DBI and
   - attempt to list all 1/4 1/4 sections which were not put in the data base along with the reason why they were not; and
3) it will back up the disk STIFLE on a tape STIFLE which could be used to regenerate the NARIS data base should it ever become necessary.


DISK FILE
NARIS/SYSTEM/STIFLE
DBI
DISK FILE CREATED
NARIS/SYSTEM/DATABASE
FILES PUT ON DISK FROM TAPE BY DBI PROGRAM
PRINTER OUTPUT GENERATED BY DBI
STIFLE PUT ON "BACKUP" TAPE BY DBI
DATABASE PUT ON NARISDATA TAPE BY DBI

Creation of DATABASE by DBI

Figure 2

DBI Tape Handling

When initiated, DBI:

1) asks for a tape to be mounted and puts a "fresh" copy of system files on disk
   (these files are NARIS/SYSTEM/DATABASE,
   NARIS/SYSTEM/SYMTAB,
   NARIS/SYSTEM/TRACTDICT, and
   NARIS/SYSTEM/MAPBASE)
2) will then ask that the last used STIFLE tape be mounted without a ring (prevents it from being accidentally erased);
3) it will then ask that the penultimate STIFLE tape be mounted on a unit with a ring (allowing the file to be written);
4) backup information for STIFLE will be copied from the last to the penultimate tape;

After processing, DBI will

5) request that a tape is mounted with a ring and put system files (including the DATABASE) on this tape;

Note that DBI will remove the file NARIS/SYSTEM/STIFLE when finished with it.

Following DBI's completion, the individual performing these tasks gathers the Preprocessor and DBI listings and the data decks, and takes them back to his office for filing and error checking.

The procedure which has been used for filing and error checking is:

1. Xerox 1 copy of each DBI stifleblock listing;
2. Update the data status chart to show that an attempt has been

made to insert data for the class/township.

3. File the Xeroxed copy, the Preprocessor listing, and the coding forms for the class/township in the same folder - the folder itself being filed under the township-range "file";
4. File the "original" DBI listings in a "DBI Binder".
5. Return the data decks to the data file cabinet;
6. If errors have been noted by the Preprocessor or DBI, the data card(s) pertaining to the 1/4 1/4 section should be corrected (keypunched) and inserted in the data base. If data already exists in the data base for the 1/4 1/4 section to be reinserted, all cards for the tract should be run with a header card of $$UPDT/<township-range>/<class name>. This <header> card instructs DBI to remove data currently in the data base for the specified 1/4 1/4 sections and to insert the data given in the STIFLE.

The <header> card, $$DLTE/<township-range>/<class name> is used to delete the data from the data base. An example might be:

```
?USER = CACNNARIS
?EXECUTE DSYSTEM/NARIS/PREPRØCESSØR
?DATA  FØ/CDS
$$DLTE/44N05E/SØIL
{Those data cards (from the SOIL data deck which was
used as "$$DATA" input) which refer to tracts on which
SOIL data is to be removed}
?END
```

7. When the data in the data base for the class/township has been

corrected, the data status chart is updated to show that the data for this class/township is "clean".

To conclude the discussion of the normal Data Insertion Process, the following timing estimates are given pertaining to the tasks. It should be noted that some items are not taken into account here, e.g. time to route coding forms through the keypunch service and time involved with filing and cataloging the data cards and forms. Also, of course, the entire procedure is machine (computer) dependent - if the computer is "busy" or not operating, the estimated time may become considerably higher than what is given here.

NORMAL INPUT PROCESS	TIME Min:Sec
1. Initialize	
a. gather data decks to correct and to put in on first pass and go to machine room	10:00
b. type headers - 13 (15 sec/<header>)	3:15
. 5 new classes	
. 4 classes to be corrected = 8 <headers>	
c. make corrections on data cards	26:40
. 4 classes x 20 errors (20 sec/error)	
2. Run PP	35:00
. 5 classes at 5 min per class	
. 4 classes at 2:30 per class	
3. Run DBI	
a. Tapemounts and load files	7:30 (±2:30)
b. Process a stifleblock	
. 4 classes (30 sec/class)	2:00
. 5 classes (1:00-3:00/class)	10:00 (±5:00)
c. Dumping files to tape	6:30 (±1:30)
4. Gather listings and cards and return to office	10:00
5. File and Error check	
a. Xerox DBI stifleblock and file original	11:30
. 30 sec/stifleblock	
. 7 min overhead	
b. Note errors from DBI/PP for each class/twnp when 4 of the 9 classes have errors	36:00 (±4:00)
. 8 - 10 min/class	
c. Pull cards and write corrections on cards	26:40
. 20 sec/card	
. estimate a 3 1/2% error rate (tracts per <township-range>/class) - the number of tracts to be corrected will be 20	
. cards affected per class (30)	
d. File PP listings and Xeroxed DBI/stifleblock	5:00
e. Update Data Chart	5:00
6. Prepare decks to be run next time	5:00
	3:20:05 (±14:00)

It should be realized that the average time for processing each class for the entire procedure may be reduced by adding more classes to be processed per data base version. For example, the average time to insert a class of data in the procedure shown above is about 33 1/2 minutes; however, additional classes added to this procedure would add approximately 24 minutes per class to the total time involved.

II. Adding a New Data Class

A. Global Considerations

The first step to take when implementing a new Data Class into NARIS is to determine the type of Data Elements which comprise the Class. One must ascertain whether or not the Data Elements and the kind of values they represent are compatible with the existing structure of the Symbol Table. The system currently deals with only five types of Data Elements which have data values of the following kinds:

. Straight integer range;

. Bit patterns;

. Single-bit boolean;

. Integer with decimal point shift; and

. Expandable table translation.

A complete description of how the types of Data Elements mentioned above are "converted" and stored in the system may be found in the _NARIS Software Manual_, Section 3, The Symbol Table.

If a new type of Data Element (kinds of values other than noted above) is present in the new Class, the SYMBOL TABLE GENERATOR and EDICONVERT (same reference as above) will need to be modified to recognize and handle this kind of data. In the section of code, EDICONVERT, the following procedures will need to be modified:

INCONVERT

. will need a new CASE statement which compresses the external values into internal form in the manner specified by the Symbol Table.

OUTCONVERT

- will need a new CASE statement which "decompresses" the converted values to "real" values.

MAKINCONVERRTEXT

- will need CASE statements which produce error messages relating to errors noted in attempting to convert a data value in INCONVERT;

NORMALIZE

- if the new kind of values are numeric, a CASE statement dealing with NORMALIZE processing would be added here.

In the SYMBOL TABLE GENERATOR program, the following modifications would be required. All code modifications occur within procedure GETANEDI.

GETANEDI

- change the end test for highest legal translation type to a new high value;
- add a case statement to TTYPE processing;
- for complex translation it may be necessary to write a separate procedure (such as DOTTYPE5).

B. The Symbol Table

The processing done by the Preprocessor (and throughout NARIS) utilizes the NARIS Symbol Table in order to check actual data values which are present on the data card against the range of values that the Symbol Table stipulates for each Data Element. Thus, upon receipt of a new Data Class, the Symbol Table must be modified to include the new Class, its Data Elements, and their range of values. To change the Symbol Table, one must first alter or amend the data from which the Symbol Table is constructed - this data comprises the Symbol Table Generator Deck which is the data deck for the program (the Symbol Table Generator) which builds the Symbol Table.

Following is a BNF-like description for the format of adding a Data Class to the STG Deck:

```
 <DECK>:: = a list of <WHOLE CLASS>ES followed by an <ENDCARD>
  <WHOLE CLASS>:: = <CLASS HEADER> followed by a list of <WHOLE ELEMENT>S
   <CLASS HEADER>:: = "CLASS" <CLASS NAME><AGENCY><TEXT>
    <CLASS NAME>:: = <NAME>
     <NAME>:: = any combination of up to 35 alphanumerics which does
                not start with a numeric, and is not the same as a
                reserved word in the retriever, nor is it the same as
                a previous <NAME> in the symbol table.
   <AGENCY>:: = 1 or more cards starting with "**"

   <TEXT> ::  = a semicolon or 1 or more cards with a "*" in Col. 1 and
                text on the remainder of the card

 <WHOLE ELEMENT>:: = <ELEMENT HEADER><ATTRIBUTES>
  <ELEMENT HEADER>:: = "EDI" <ELEMENT NAME> or "ELEMENT' <ELEMENT NAME>
   <ELEMENT NAME>:: = <NAME>

  <ATTRIBUTES>:: = <TRNSTYPEOINFO> or
                   <TRNSTYPE2INFO> or <TRNSTYPE3INFO> or
                   <TRNSTYPE4INFO> or <TRNSTYPE5INFO>

   <TRNSTYPEOINFO>:: = "TTYPE 0" "LOW" <LOVALUE> "HIGH" <HIVALUE>
            followed by <DEFAULTSTUFF> followed by <TEXT>

     <LOVALUE>:: =  <POSITIVE INTEGER VALUE>
      <POSITIVE INTEGER VALUE>:: = any string of digits
     <HIVALUE>:: = <POSITIVE INTEGER VALUE>
      DEFAULTSTUFF :: = <NULL>  or "DEFAULT" followed by the value for this
                EDI to be given for default occurrences

   <TRNSTYPE2INFO>:: = "TTYPE 2" "BITS" <NUMBER OF BITS>
             <DEFAULTSTUFF><TEXT>
   <NUMBER OF BITS>:: = <POSITIVE INTEGER VALUE>

   <TRNSTYPE3INFO>:: = "TTYPE 3" <DEFAULTSTUFF><TEXT>

   <TRNSTYPE4INFO>:: = "TTYPE 4" "LOW" <LOWVAL> "HIGH" <HIGHVAL>
            "PRECISION' <PRECVAL><DEFAULTSTUFF><TEXT>
    <LOWVAL> :: = <DECIMAL VALUE>
    <HIGHVAL>:: = <DECIMAL VALUE>
    <PRECVAL>:: = <DECIMAL VALUE> Note: must be a power of 10 (e.g., 10,
                                  10000, .001)
    <DECIMAL VALUE>:: = any string of digits and not more than one decimal point.

    <TRNSTYPE5INFO> :: = "TTYPE 5" "STRINGS" <MAX NUMBER OF STRINGS>
             <DEFAULTSTUFF><TEXT> followed by a list of  <ELEMENT VALUE>S

    <ELEMENT VALUE>:: = "VALUE" <VALID STRING><TEXT>
    <VALID STRING> :: = any combination of alphanumerics which has not
                        already been used as a valid string for this
                        data element.

    <ENDCARD>:: = a card which reads "END;"
```

Some comments pertaining to the above syntax:

If DEFAULT is specified for any data element in a class, DEFAULT must be specified for all of the data elements in that class.

. A Semicolon is allowed (but not necessary) between any command and its text

. Also, another variety of CARD is acceptable:

```
'$INCLUDE "<filename>."'
```

The program runs exactly as though the cards in file "<filename>" had appeared in the original deck in place of the INCLUDE card. Note that "$INCLUDE" cards are not acceptable within "included" decks.

. Each new command must be started on a new card.

An example of an actual Symbol Table Generator deck is:

```
CLASS GEOSANDGRAVEL ;                                                      00032000
**ILLINOIS STATE GEOLOGICAL SURVEY, URBANA, ILLINOIS.                      00033000
*PROVIDED BY THE ILLINOIS STATE GEOLOGICAL SURVEY AND CODED FROM THEIR     00033100
*MAP "SAND AND GRAVEL RESOURCES OF MCHENRY COUNTY"                         00033200
EDI TYPE TTYPE 5 STRINGS 9  ;                                              00033300
VALUE BR ;                                                                 00037400
*BEDROCK                                                                   00037500
VALUE G3  ;                                                                00034700
*SAND AND GRAVEL PARTICLE SIZE EXTREMELY VARIABLE, WITH COBBLES GREATER    00034800
*THAN 6 INCHES IN DIAMETER;SOME DEPOSITS CONTAIN LESS                      00034900
*THAN 40 PERCENT COARSER THAN 4-MESH.THICKNESS VARIABLE.MAY BE UP TO       00035000
*100 FEET IN EASTERN PART OF COUNTRY BUT AVERAGE DECREASES TO WEST.        00035100
*AREAL EXTENT OF DEPOSITS VARIABLE IN THE EASTERN THIRD OF THE COUNTY      00035200
*WHERE GRAVEL BED MAY BE THIN OR ABSENT LOCALLY,THE AREAS OF G3 IN THE     00035300
*SOUTH CENTRAL PART OF THE COUNTY MAY BE LESS VARIABLE,BUT LACK OF DATA    00035400
*PRECLUDES GIVING THEM A HIGHER RATING.LOCALLY,GRAVEL MAY BE COVERED       00035500
*BY THIN TILL.                                                             00035600
VALUE Y3 ;                                                                 00036700
*SAND;NO IMMEDIATE SIGNIFICANT COMMERCIAL VALUE BECAUSE ALL THE SAND       00036800
*NEEDED AT PRESENT AND IN THE NEAR FUTURE IS AVAILABLE FROM GRAVEL         00036900
*PITS.                                                                     00037000
VALUE G2 ;                                                                 00034100
*SAND AND MEDIUM-GRAINED GRAVEL;MORE THAN 40 PERCENT OF MATERIAL IS        00034200
*GENERALLY COARSER THAN 4-MESH,BUT SOME SAMPLES CONTAIN LESS THAN 10       00034300
*PERCENT COARSER THAN 1 INCH.THICKNESS GENERALLY 10 TO 20 FEET,THINNER     00034400
*AT WESTERN MARGIN.GRAVEL BED ESSENTIALLY CONTINUOUS,BUT THERE MAY BE      00034500
*MINOR INTERRUPTIONS.                                                      00034600
VALUE Y2 ;                                                                 00036300
*MOSTLY SAND WITH SOME PEA GRAVEL;PARTICLE SIZE NOT SUITABLE FOR COARSE-   00036400
*GRAINED CONCRETE AGGREGATE,BUT PEA GRAVEL MAY FIND CERTAIN                00036500
*SPECIALIZED USES.                                                         00036600
VALUE G1 ;                                                                 00033400
*SAND AND COARSE-GRAINED GRAVEL:OF MATERIAL PASSING 2-INCH SIEVE,MORE      00033500
*THAN 10 PERCENT IS COARSER THAN 1 INCH AND MORE THAN 40 PERCENT IS        00033600
*COARSER THAN 4-MESH SIEVE (.185 INCHES): FEW COBBLES. MORE THAN 4 INCHES  00033700
*IN DIAMETER.THICKNESS MORE THAN 60 FEET NEAR EASTERN MARGIN BUT THINGS    00033800
*RAPIDLY TO WEST.GRAVEL BED ESSENTIALLY CONTINUOUS,BUT BEDS OF SILT MAY    00033900
*OCCUR WITHIN THE GRAVEL SEQUENCE IN PLACES.                               00034000
VALUE Y1 ;                                                                 00035700
*SAND AND GRAVEL DISCONTINUOUS;GENERALLY SUITABLE FOR CONCRETE             00035800
*AGGREGATE PRODUCTION.BUT PARTICLE SIZE,THICKNESS,AND AREAL EXTENT OF      00035900
*DEPOSITS TOO VARIABLE FOR MINING BY LARGE-SCALE OPERATION.SOME DEPOSITS   00036000
*CONTAIN INTERBEDDED SILT LAYERS.BELIEVED TO BE SUITABLE FOR SMALL-SCALE   00036100
*OPERATION TO SUPPLY LOCAL USE.                                            00036200
VALUE R1 ;                                                                 00037100
*NO SAND OR GRAVEL DEPOSITS OF LARGE ENOUGH EXTENT TO HAVE COMMERCIAL      00037200
*VALUE.                                                                    00037300
VALUE WA ;                                                                 00037600
*WATER                                                                     00037700
EDI ACRES TTYPE 4 LOW 10 HIGH 40 PRECISION 10;                             00037800
EDI DATE TTYPE 0 LOW 60 HIGH 75 ;                                          00037900
EDI SUMACRES TTYPE 4 LOW 40 HIGH 40 PRECISION 10 ;                         00038000
CLASS FORESTRY ;                                                           00000100
**  ILLINOIS DEPARTMENT OF CONSERVATION, DIVISION OF FORESTRY, LISLE, ILL  00000200
*     "FORESTRY" IS GROUPINGS OF NATIVE OR PLANTED WOODY VEGETATION        00000300
*OF AT LEAST ONE ACRE IN SIZE.                                             00000400
EDI STANDNB TTYPE 0 LOW 0 HIGH 15 DEFAULT 0 ;                              00000500
*THIS IS AN IDENTIFICATION NUMBER ASSIGNED TO EACH STAND SO THAT IT MAY    00000600
```

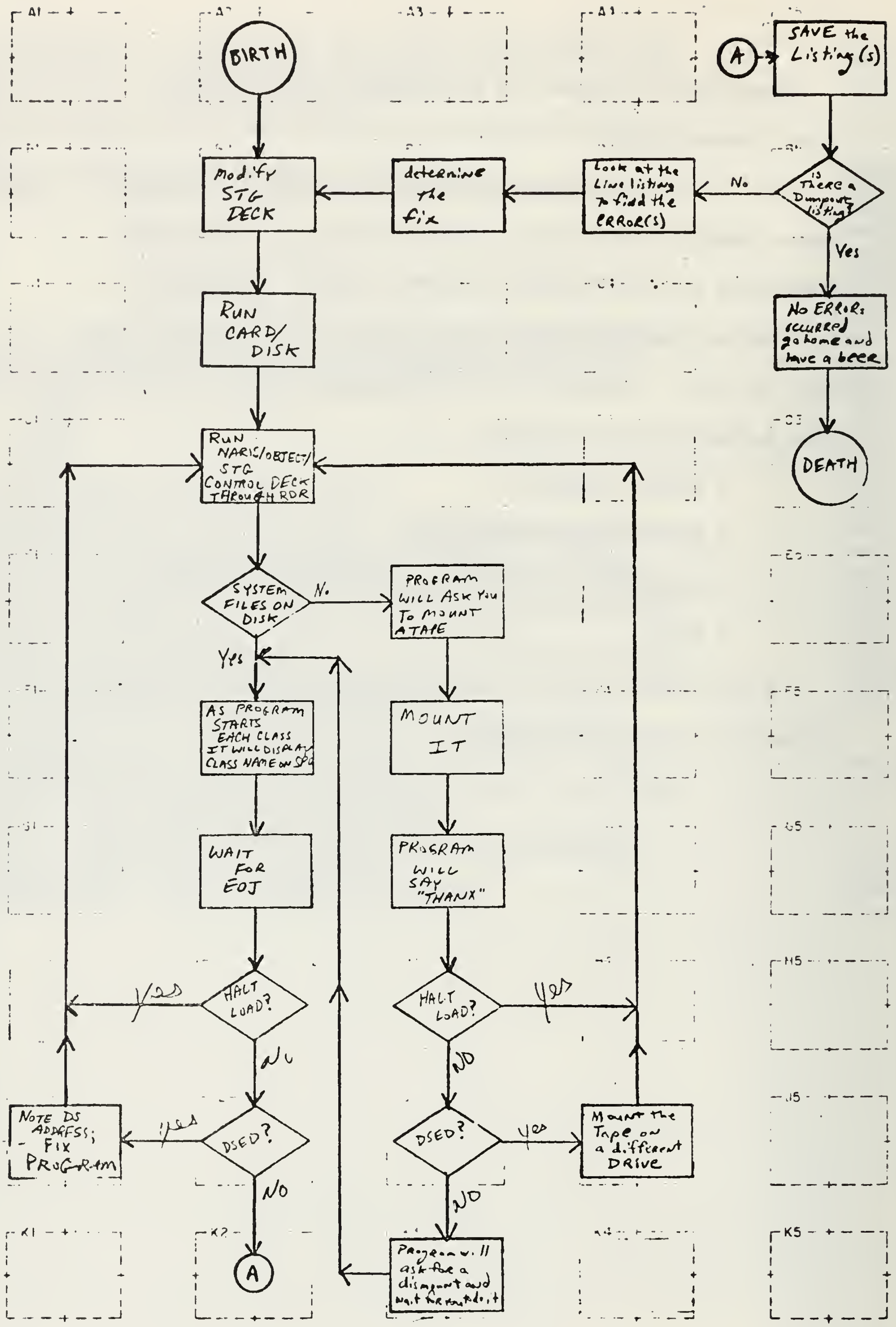
```
*BE IDENTIFIED WITHIN THE TRACT. IF A TRACT CONTAINED THREE STANDS, THEY
*WOULD BE NUMBERED 1, 2 AND 3. STANDS ARE NUMBERED SEPARATELY REGARDLESS
*OF SPECIES OR COVER VARIATION WITHIN THE STANDS
EDI ACRES TTYPE 0 LOW 0 HIGH 45 DEFAULT 0;
*THIS IS THE ACREAGE FOR THE STAND DESCRIBED ON THE LINE ON WHICH IT
*APPEARS.
EDI LOCATION TTYPE 2 BITS 16 DEFAULT '''''''''''''''' ;
*THE LOCATION OF THE STAND ON A 16 POINT GRID. THE GRID IS NUMBERED
*
*  N                  /  4   3   2   1   /
*W   E                /  5   6   7   8   /
*  S                  / 12  11  10   9   /
*                     / 13  14  15  16   /
*
*FOR EXAMPLE A STAND THAT FILLED THE SOUTH-EAST QUARTER OF A FORTY-ACRE
*TRACT WOULD BE CODED ''''''''XX''''XX, AND A STAND THAT WAS CODED
*XXX'XX'''XXX'''' WOULD BE SITUATED AS SHOWN BELOW
*        / XXX/
*        /XX  /
*        /XXX /
*        /    /
*THIS GIVES ACCURATE PLACEMENT TO 2.5 ACRES
EDI OVERLAP TTYPE 2 BITS 4 DEFAULT '''' ;
*THE DIRECTION IN WHICH THE STAND OVERLAPS. IF THE STAND CROSSES THE
*NORTHERN BOUNDARY OF THE TRACT, IT OVERLAPS NORTH, ETC.
*SPACES ARE IN THE ORDER NORTH,EAST,SOUTH,WEST. IF AN AREA OF NATIVE
*WOODLAND IS NOT TOTALLY ENCLOSED WITHIN A 40-ACRE TRACT THE DIRECTION
*OF OVERLAP(S) OVER THE 40-ACRE BOUNDARY LINES IS NOTED BY AN "X" IN THE
*APPROPRIATE COLUMN OR COLUMNS.
*FOR EXAMPLE, A STAND WHICH OVERLAPED BOTH EAST AND SOUTH WOULD BE CODED
*'XX', AND A STAND CODED '''X ONLY OVERLAPS WEST
EDI COVEROVER40 TTYPE 0 LOW 0 HIGH 45 DEFAULT 0 ;
*  CROWN COVER OVER 40  IS THE NUMBER OF ACRES OF FORESTRY IN THE
*  STAND THAT HAVE OVER 40 PER CENT CROWN COVER.
EDI COVERUNDER40 TTYPE 0 LOW 0 HIGH 45 DEFAULT 0 ;
*  CROWN COVER UNDER 40 IS THE NUMBER OF ACRES OF FORESTRY IN THE STAND
*  THAT HAVE A CROWN COVER LESS THAN 40 PER CENT.
EDI DATE        TTYPE 0 LOW 60 HIGH 75 DEFAULT 60 ;
*  THIS IS THE DATE THE DATA WAS RECORDED FROM AN ON-SITE INSPECTION.
EDI SUMACRES TTYPE 0 LOW 0 HIGH 45 DEFAULT 0 ;
* THE TOTAL NUMBER OF ACRES OF FORESTRY IN THE TRACT
END ;
```

Operating procedures for generating a "new" NARIS Symbol Table follow.

Due to the size of data involved as input to the Symbol Table Generator (STG), the procedure has been to keep the components of the STG deck on magnetic tape. Thus prior to re-creating the Symbol Table, the deck(s) are transferred from tape to disk. Subsequently, the following control cards may be read through the card reader:

```
? USER = CACNNARIS

? EXECUTE NARIS/OBJECT/STG

? FILE CARD = <file name for input deck> SERIAL

? END
```

A flow chart depicting the process of modifying the STG is presented on the following page.


BIRTH
Modify STG DECK
determine the fix
Look at the Line listing to find the ERROR(S)
No
Is There a Dumpout listing?
A
SAVE the Listing(s)
Yes
No ERRORs occurred go home and have a beer
DEATH
RUN CARD/DISK
RUN NARIS/OBJECT/STG CONTROL DECK THROUGH RDR
SYSTEM FILES ON DISK
No
Yes
PROGRAM WILL ASK YOU TO MOUNT A TAPE
AS PROGRAM STARTS EACH CLASS IT WILL DISPLAY CLASS NAME on SPC
MOUNT IT
WAIT FOR EOJ
PROGRAM WILL SAY "THANX"
HALT LOAD?
Yes
No
HALT LOAD?
Yes
NO
NOTE DS ADDRESS; FIX PROGRAM
Yes
DSED?
NO
A
DSED?
Yes
Mount the Tape on a different DRIVE
NO
Program will ask for a dismount and wait for you to do it

Soon, after the program (STG) begins, it will request that a tape be mounted by the operator if the files it wants are not present on disk. When mounted, STG will read files from the tape to disk. Among these files is the current version of the Symbol Table. If STG has difficulty reading the tape simply ask the computer operator to move the tape to another tape drive. As the STG processes each class, it will display the name of the class on the Supervisory Console (SPO).

If a HALT/LOAD occurs prior to EOJ or if the job is DS-ED for any reason other than failure to read the tape, re-submit the 4 card control deck - this time STG will not request the tape mount since the files it wanted are still on disk.

Following EOJ, a listing will appear on the line printer. On the last page of the LINE listing (there may also be a DUMPOUT listing) will appear the line

"{number} ERRORS OCCURRED"

If the number is not zero, the new Symbol Table was not constructed; thus, programs which depend on the new Symbol Table should not be run until it is fixed. If the number was zero, the new Symbol Table was created and may be accessed in later jobs. The listing(s) produced are filed in a Data Input Binder - the binder may be kept in the computer room where it is accessible to others who need this information when writing programs or debugging jobs.

C. The Preprocessor (NARIS/SOURCE/PREPROCESSOR)

Following the completion of modifications to the STG deck for the new Class, the Preprocessor may be amended so that it is capable of processing data for the Class.

Using the example which was given as a sample input deck to the STG, one notes that the Preprocessor must be able to recognize the Class, GEOSANDGRAVEL, and be able to process the values in each Class occurrence.

Figure 3 shows a NARIS coding form on which GEOSANDGRAVEL data is coded. The Preprocessor must be able to read and process data cards which have this format; therefore, the following additions would be made in the Preprocessor.

Class	-	GEOSANDGRAVEL
Data Element	-	DATE
Data Element	-	SUMACRES
Data Element	-	TYPE
Data Element	-	ACRES

In the Preprocessor where "class initialization" processing is performed, one would enter the following code:

```
If D1 = "GEØSANDGRAVEL" THEN
BEGIN
DEFINE GEØS = #;
REPLACE PØINTER (FØTEST[0]) BY D1 FØR LENGTH," "FØR 60-LENGTH;
MØDE: = {the last MØDE + 1};
FØCLASSNUM;
I: = -1;
ØCCURRENCE ("DATE");
ØCCURRENCE ("SUMACRES");
ØCCURRENCE ("TYPE");
ØCCURRENCE ("ACRES");
END
ELSE
```

		TRACT NUMBER															
CARD TYPE	COUNTY CODE	1/4 1/4 SECTION	1/4 SECTION	SECTION	TOWNSHIP	RANGE	DATE	SUM OF ACREAGE	SAND & GRAVEL RATING	ACREAGE	SAND & GRAVEL RATING	ACREAGE	SAND & GRAVEL RATING	ACREAGE	SAND & GRAVEL RATING	ACREAGE	
1 2	3 4 5	6 7 8	9 10 11	12 13	14 15 16	17 18 19	20 21	22 23	24 25	26 27	28 29	30 31	32 33	34 35	36 37	38 39	40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

The code above allows the Preprocessor to scan off the Class name, GEOSANDGRAVEL, when it appears on a <header> card and puts the Preprocessor in a mode which will process only GEOSANDGRAVEL data.

. The REPLACE statement is used to fill an array for the DEFINE, FØCLASSNUM, with the Class name (GEOSANDGRAVEL) from the <header> card;

. MØDE is an integer used to denote the data Class which is being processed;

. FØCLASSNUM is a DEFINE used to initialize a value for the integer, CLASSNUM. This is done by accessing the Symbol Table.

. OCCURRENCE is a DEFINE used to access the Symbol Table and store the Data Element number in an array. Thus, further processing of Data Elements is performed by using the unit of the array. Note that the integer, I, must be initialized to minus one prior to storing Data Element numbers in the array.

Next one must check any special effects that this data processing has for each 1/4 1/4 section. For example, currently the data for all geology Classes has been coded such that a description of GEOSANDGRAVEL in a 1/4 1/4 section is always entirely contained on one data card. Therefore, if more than one data card were "read" for a 1/4 1/4 section, an error message would be generated (see TRACT/DATA FORM PROCESSING in the Preprocessor source code). Having taken care of Class processing which is dependent on the number of data cards per 1/4 1/4, one will note that the Data Elements, DATE and SUMACRES, are processed the same way for a number of Classes (see SPECIAL GENERAL OCCURRENCE PROCESSING in the source code listing).

GEOSANDGRAVEL is one of these Classes and the following additions to the processing taking place in SPECIAL GENERAL OCCURRENCE PROCESSING mentioned above in the source code would be:

1) add between

"MODE = 11" and "THEN" the following:

```
"OR MODE = {MODE number established in initializing
            the Class}"
```

2) add between

"MODE = 9" and "THEN" the following:

```
"OR MODE = {MODE number established in initializing
            GEOSANDGRAVEL}"
```

One should now construct a Class label - it should be declared at ADD NEW CLASS LABELS HERE in the Preprocessor source code. The label will be used following END OF SPECIAL GENERAL OCCURRENCE PROCESSING in the following manner:

```
IF MODE = {MODE number for GEOSANDGRAVEL} THEN GO TO
<Class label>;
```

The label and the occurrence processing for the Class would be inserted in the source code at CLASS OCCURRENCE PROCESSING FOR 'NEW' CLASSES as:

```
<class label>:
     BEGIN  INTEGER  MAKEØWNBLØCK;
     IENTER("GEØSAN",);
     IF NØTFIRSTPASS THEN REPLACE C1:C1 BY DS FØR 29;
     FØR I:=2 STEP 1 UNTIL 3 DO
           BEGIN
```

```
            DIRTYVALUE;

            CASE I ØF BEGIN I:=0; I:=0; CØNVERSIØN ("TYPE");

              CØNVERSIØN ("ACRES");

                       END;
        END;
    NØTFIRSTPASS: = TRUE;

    I: = 3;

      ACREAGESUM;

      ILEAVE("GEØSAN",);

      IF (STNAC=TAC AND D1= " ") ØR ØCCURRERR THEN

        BEGIN

           STNAC: = TAC: = 0;

             GØ  FØWORK;

           END ELSE

             GØ <class label>;

      END;
```

Explanation of Code used:

. IENTER and ILEAVE are used to provide timing information regarding the code between them.

. The REPLACE statement is used for each occurrence (other than the first) so that the stored values (saved in SPECIAL GENERAL OCCURRENCE PROCESSING) for certain Data Elements are placed in the buffer to constitute a "whole" <class occurrence> .

. CONVERSION is a DEFINE used to compress and store information in the buffer: the value of the Data Element (the Data Element number is accessed from the array created by DEFINE ØCCURRENCE) which was scanned by DEFINE DIRTYVALUE is checked through the use of procedure INCONVERT against the range of allowed values in the Symbol Table. Error messages are generated to the line printer file when "bad" values are noted.

. ACREAGESUM is a DEFINE used to accumulate occurrence acreage and compare it with total acres (SUMACRES).

Some special characteristics were noted about the coding of GEOSANDGRAVEL in order to process occurrences in the manner presented above:

. an occurrence of GEOSANDGRAVEL data in a 1/4 1/4 section consists of 4 Data Element values;

. the data is coded in such a way that the values for the first 2 Data Elements must be stored - in case the Data Elements TYPE and ACRES, occur more than once; and

. the sum over the tract of the data values for ACRES should equal the data value of SUMACRES.

Following the modifications to the Preprocessor, the program should be compiled:

```
? USER = CACNNARIS
? COMPILE DSYSTEM/NARIS/PREPROCESSOR ALGØL LIBRARY
? DATA
  <Preprocessor deck>
? END
```

Now that the new Data Class has been put in the Symbol Table and the Preprocessor, the Class may be considered "resident" (see part of this manual - Data Insertion of an Existing Class) for purposes of inputting the data to the NARIS Data base.

III. Adding Geographic Areas to the Data Base

THE TRACT DICTIONARY GENERATOR

The Tract Dictionary Generator (TDG) program adds to the geographi range of the data base by

- adding keys to the tract dictionary.
- adding records to the data base. These records will contain only SURVEY information after a TDG run, and will be ready to accept input from a normal data base insertion process (see Part I of this manual).

To run TDG, load a copy of NARIS/OBJECT/TDG on disk. Run the cards:

```
?USER=CACNNARIS
?RUN NARIS/OBJECT/TDG
?DATA CARD
     <data>
?END
```

<data> is a list of the tracts to be inserted into the system. The format of the list is <tract specifications> (see format in NARIS User Manual) separated by semicolons, with an "END# " signifying end of list.

For example,

```
?USER=CACNNARIS
?RUN NARIS/OBJECT/TDG
?DATA CARD
```

```
                    T44NR12E; SH SEC 1-6, SEC 7-36 T43NR11E;
                              END #
                ?END
```

will insert all of township T44N R12E, the south half of sections 1 through 6 and all of sections 7 through 36 of township T43N R11E. <data> is free format and may extend over several cards.

Three important restrictions exist in the running of TDG:

1. All the tracts for a township must be inserted in the same tract specification. Note that if tracts are not specified correctly, they cannot be corrected except by destroying all the Data base and TRACTDICT information describing the township. It is also not possible to input

```
SH SEC 1-6 T43NR11E;  SEC 7-36 T43NR11E;
```

   instead one must write: `SH SEC 1-6, SEC 7-36 T43NR11E;`

   or the second tract specification will be flagged with an error, nor can two tract specifications for the same township appear in different runs of TDG. All the tracts for a given township must be inserted with one tract specification.

2. <u>TDG CANNOT BE RUN WHEN DBI IS BEING RUN</u>. If this were to occur, the first program to finish processing would have its files overwritten following termination of the second program. Note that only the files created by the second program would be "backed-up" on tape. Re-running the first program would fix this problem.

3. Townships which are outside the area specified in the first level of TRACTDICT (specified at INITIALIZETDG time) cannot be added.

TDG will generate a listing with the time and the date of the run, the run number, the input data, and a dump of the Tract Dictionary. TDG will also generate error messages - if any error conditions are sensed. This includes townships already in the system, errors in the tract specification syntax, or problems in allocating new DATABASE records.

Deleting a Township from the NARIS System

To delete a township from NARIS, run TDG and precede the tract specification of the township to be removed by the word "DELETE". For example, the TDG input

```
DELETE T44NR5E; T44NR5E; END#
```

will remove the township T44N R5E from TRACTDICT and DATABASE. The township will then be re-inserted containing no data (except for Data Class SURVEY).

Because the records in DATABASE which are freed by the DELETE are r immediately used (they are allocated to the overflow record queue), the frequent repetition of the DELETE operation will have the unpleasant sic effect of making DATABASE grow in size; it is thus an operation which sh be used only as a last resort (if ever).

Data Structure Revisions - Classes, Data Elements, and Data Values

This section of the Data Input Manual is designed to provide procedures/methods to use when confronted with implementing any change in the status of the data - as defined by the data supplier; the following is a list of possible changes of "data status" - any one of which the data supplier might envision at some time:

1. Adding a new Data Class to the Data Base

   . See Part II of this manual.

2. Adding table-valued values to a Data Element

   . If the total number of VALUES does not exceed the STRINGS number, simply put the new values at the end of the Data Element's VALUES in the STG deck, run STG - creating a new Symbol Table, and put the data in the data base through the Preprocessor and DBI. However, if there are now to be more VALUES than the STRINGS number, increase the STRINGS number after doing 3a) and then do 3c) and 3d).

3. Changing table-valued values of a Data Element

   Note that removal of table-valued Data Element VALUE(s) may cause other table values for the same Data Element to be OUTCONVERTed incorrectly because the list collapses to fill the hole.

   When changing table-valued values, e.g., Soil Number 24 has been correlated to Soil Number 27 (if Soil Number 27 did not exist in the Symbol Table, simply changing the "4" to "7" on the card in the STG DECK will be satisfactory. Of course, the Symbol Table will need to be re-created).

(a) Remove all SOIL data from the data base

- see the use of "$$DLTE" PP and DBI <header> card described in Part I of this Manual.

(b) Remove "VALUE 24;" and associated "*" text from the STG DECK;

(c) Add text which was for the former VALUE 24 to the new VALUE 27;

(d) All data value "24"'s must be changed on the data cards to "27";

(e) Re-create new Symbol Table;

(f) Re-insert all SOIL data.

4. Changing the size of an existing Data Element's values (the Data Element and all later Data Elements in the Class change location, which invalidates existing data base data).

. e.g. let us assume that the data supplier is now providing ACRES resolution (Class GEOSANDGRAVEL) to .1 acres instead of to the nearest 10 acres.

a) Remove all GEOSANDGRAVEL data from the data base

b) Change specifications of Data Elements ACRES and SUMACRES in the STG Deck to be

```
"EDI SUMACRES TRNSTYPE 4 LOW .1 High 40 PRECISION .
 EDI ACRES    TRNSTYPE 4 LOW .1 High 40 PRECISION .
```

c) Re-create new Symbol Table;

d) Re-insert all of the revised GEOSANDGRAVEL data.

5. Adding a new Data Element to an existing Class

(The problem is that the occurrence length for the class may change or data elements may change positions within the occurrence.)

a) Remove all of this Class's data from the data base;

b) Add appropriate definition of the new Data Element to the Class within the STG Deck;

c) Re-create the new Symbol Table;

d) Amend the Preprocessor so that values for the new Data Element can be processed;

e) Compile the Preprocessor;

f) Re-insert all of the revised data for the Class into the data base.

6. Changing a Class name or Data Element name

. Due to the affect of this change on the data base regeneration procedure (note that STIFLEs which "back up" the data base would still have the old name), the following action would be taken:

a) Remove all data for the Class from the data base;

b) Change the <NAME> in the STG DECK to the new <NAME>;

c) Re-create the Symbol Table;

d) Change the <NAME> of the Data Element in the Preprocessor;

e) Compile the Preprocessor;

f) Note that regeneration of the data base from the STIFLE backup tapes (Data Base Insertion program) will no longer read and process the affected values whose <NAME> was changed - the Symbol Table will have no reference to the old <NAME>.

7. Deleting a Data Class from NARIS

. the removal of a Class of data (no data to be reinserted) affects each subsequent Data Class in the STG deck - as does the re-ordering of the Classes in the deck. In particular, the correspondence between classes and header fields in the database may be disrupted.

a) To delete a Data Class entirely, one must first remove all data for the Class from the Data base;

b) Remove all data cards from the STG deck which defined the Data Class;

c) Re-create the Symbol Table;

d) Regenerate the data base from the STIFLE backup tapes (Data Base Insertion program) - the "pointers" in the data base will now be accessing the right data;

e) Note that all processing related to the former Data Class may be removed from the Preprocessor;

f) The Preprocessor may then be compiled to replace the forme

Preprocessor.

Following the implementing of any of the above modifications, data input may proceed as noted in Part I of this manual - Data Insertion of an Existing Data Class.

APPENDIX

Regeneration of the Data Base from STIFLE Backup

Optional DBI run modes

If DBI is run with the file include card

```
?FILE STIFLE = BACKUP
```

DBI will insert information from the tape version of STIFLE. This will force DBI to attempt re-insertion of all the data ever input into the data base, and should only be used when the system has so drastically changed as to make impossible further retrieval or insertion of data, for example, if the Symbol Table were drastically changed. DBI in BACKUP mode should only be run after INITIALIZETDG and TDG have been run. This will re-create the entire TRACTDICT and a blank DATABASE.

INITIALIZEDBI

INITIALIZEDBI is to be run once in the entire life of a NARIS System to bootstrap the STIFLE tapes. Since this routine destroys the information on the first two STIFLE tapes, it should not normally be run more than once on a given set of STIFLE tapes. In particular, it should not be run before DBI is run in BACKUP mode.

After the program has been loaded to disk, it is run using the control cards:

```
?USER = CACXNARIS
?RUN NARIS/OBJECT/INITIALIZEDBI
?DATA CARD
```

```
<tape number> <tape number>

?END
```

Where the two <tape number>s are the numbers of the NARIS tapes to be used for the STIFLE. Input is in the format 2I4.

INITIALIZEDTDG

INITIALIZETDG is run at the beginning of each cycle of the NARIS system (TRACTDICT and DATABASE become deleted). It generates the first record of the DATABASE and reserves storage for the first level of the TRACTDICT.

After the program is loaded to disk, it can be run using the control cards

```
?USER = CACXNARIS

?RUN   NARIS/OBJECT/INITIALIZETDG

?END
```

DATA INSERTION MANUAL

UPDATE #1

Subject: Data Input to UCSD, FILE NAME Changes

Many of the file names mentioned in your Data Insertion Manual have changed as a consequence of moving NARIS to UCSD.

All file names which began with:	Now begin
NARIS/OBJECT/	NARISOBJECT/
NARIS/SYSTEM/	NARISDATA/
NARIS/STGDECK/STG	NARISSTG/
The DSYSTEM file names:	are now:
DSYSTEM/NARIS/DBI	NARISOBJECT/DBI
DSYSTEM/NARIS/PREPROCESSOR	NARISOBJECT/PREPROCESSOR
DSYSTEM/NARIS/LOG	NARISPERMNT/LOG
DSYSTEM/NARIS/TIMING	NARISPERMNT/TIMING
No change has been made to file names which begin with:	they are still:
NARIS/SOURCE/	NARIS/SOURCE/

Subject: INITIALIZING THE TRACT DICTIONARY OR DATA BASE

INITIALIZETDG (see Data Insertion Manual) has been incorporated into the Tract Dictionary Generator (TDG) and INITIALIZEDBI (see Data Insertion Manual) has been incorporated into the Data Base Insertion program (DBI). These inclusions represent a substantial savings in programmer time required to include Indian boundaries in NARIS and to create IRIS I.

DBI and TDG have far-reaching and potentially destructive effects if used for initializing. If there is any data in the data base, TDG, when run in initialize mode, will destroy the data. DBI, when run in intialize mode, will destroy STIFLE: all old STIFLE tapes for that data base will be lost.

To run TDG in initialize mode:

```
?EXECUTE NARISOBJECT/TDG (or ?EXECUTE IRISOBJECT/TDG)
?DATA CARD
INITIALIZE <min east  range>, <number of east ranges>, <min west
range>, <number of west ranges>, <min north township>, <number
of north townships>, <min south township>, <number of south
townships>, <number of indian boundary townships>;
<survey description of area to be covered>
END#
?END
```

To run DBI in initialize mode:

```
?EXECUTE NARISOBJECT/DBI (or ?EXECUTE IRISOBJECT/DBI)
```

```
        ?FILE STIFLE=INITIALIZE

        ?END
```

If a disk file by the name NARISDATA/STIFLE (or IRISDATA/STIFLE) exists, the data in it will be inserted into the data base after DBI completes initialization.

NARIS DATA INSERTION MANUAL

UPDATE #3

SUBJECT: IRIS I Data Input

The IRIS I Data Base, Tract Dictionary, and Symbol Table are separate files from their NARIS counterparts. This means that a completely different geographic resolution and data class set will be available to IRIS users. It also means that it will be impossible to relate NARIS data to IRIS I data through the system. See section IV, IRIS I System, of the NARIS User Manual.

The IRIS I System stores data at a resolution of 160 acres (a quarter-section); thus, there will be one-fourth as many data cards to input as in NARIS. In IRIS, a "normal" survey township contains 144 tracts.

File name mapping

NARIS file names	IRIS file names
NARISDATA/DATABASE	IRISDATA/DATABASE
NARISDATA/TRACTDICT	IRISDATA/TRACTDICT
NARISDATA/SYMTAB	IRISDATA/SYMTAB
NARISDATA/MAPBASE	IRISDATA/MAPBASE
NARISOBJECT/PREPROCESSOR	IRISOBJECT/PREPROCESSOR
NARIS/SOURCE/PREPROCESSOR	IRISSOURCE/PREPROCESSOR
NARISDATA/STIFLE	IRISDATA/STIFLE
NARISOBJECT/DBI	IRISOBJECT/DBI
NARISOBJECT/TDG	IRISOBJECT/TDG
NARISOBJECT/STG	IRISOBJECT/STG

and the classification of files	corresponds to
NARISSTG	IRISSTG

If you have comments about this manual (corrections, suggestions, etc.), please write them below and mail to

NARIS
Center for Advanced Computation
University of Illinois at Urbana-Champaign
61801