SCIENCE RESEARCH COUNCIL

RUTHERFORD LABORATORY
ATLAS COMPUTING DIVISION

F R 8 0 T E C H N I C A L P A P E R 2 5

Dimensional Flowchart Generator - an FR80 Software Tool

issued by
F Louazani
& R W Witty

4 September 1976

DISTRIBUTION:

F R A Hopgood
R W Witty
F Louazani
L O Ford
D A Duce
R E Thomas
J R Gallop
D C Sutcliffe
A H Francis

1. INTRODUCTION

The program described here is the implementation of the TREE-META dimensional flowchart definition (see Appendix 1) to generate FR80 produced dimensional flowcharts. (Ref: FR80 Technical Paper 21 -DRIVER SOFTWARE CONSTRUCTION).

1.1 DFG: What Is It?

The dimensional flowchart generator is a program to which the input is a machine readable specification of the flowchart, and the output is a dimensional flowchart drawn on FR80.

The program is written in ALGOL, and it uses the GROATS graphics package.

The top-down structured design approach was followed in developing the program.

1.2 Implementation Method

A dimensional flowchart definition consists of:

```
DIM FLOWCHART: = .NAME '\' NODE;
```

drawing the flowchart consists of:

(1) Printing the name of the flowchart.

(2) Drawing the R-dimension (ie next level).

(3) Drawing a NODE which can be several SERIALS.

(4) Each SERIAL is a flowchart in its own right, and the process of drawing is, thus, achieved recursively.

The 3-D aspect is implemented by recognizing 3 levels:

(1) The Serial dimension: drawn as a vertical line.
(2) The Parallel dimension: drawn as a horizontal line.
(3) The Refinement dimension: drawn as a diagonal line.

A statement can be either an ACTION statement or a CONDITIONAL statement. To differentiate between these 2 categories a conditional expression is denoted by drawing a half square around the beginning of its text.

The idea of 'scoped comments' is implemented by using labels to:

(1) Denote the beginning of a block.

(2) Describe briefly the action achieved in that block.

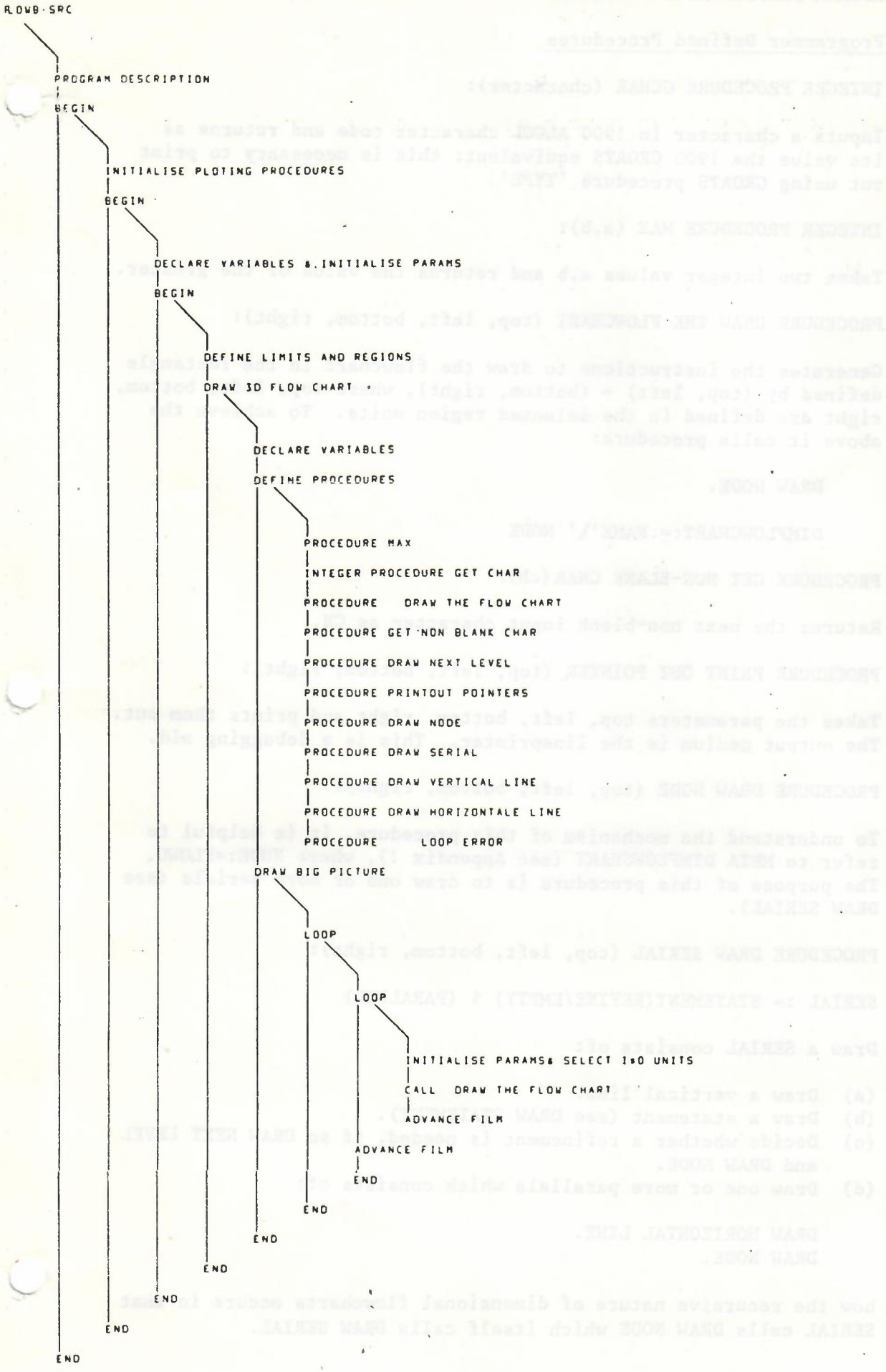
2. PHYSICAL AND LOGICAL CONSTRUCTION

2.1 Physical Layout

The following dimensional flowchart represents the physical layout (construction), of the program. It was generated using the data file listed in Appendix 4.

2.2 Logical Construction

The dimensional flowchart representing the logical flow of the program is listed at the end of this paper.


FLOWB-SRC
PROGRAM DESCRIPTION
BEGIN
INITIALISE PLOTING PROCEDURES
BEGIN
DECLARE VARIABLES & INITIALISE PARAMS
BEGIN
DEFINE LIMITS AND REGIONS
DRAW 3D FLOW CHART
DECLARE VARIABLES
DEFINE PROCEDURES
PROCEDURE MAX
INTEGER PROCEDURE GET CHAR
PROCEDURE DRAW THE FLOW CHART
PROCEDURE GET NON BLANK CHAR
PROCEDURE DRAW NEXT LEVEL
PROCEDURE PRINTOUT POINTERS
PROCEDURE DRAW NODE
PROCEDURE DRAW SERIAL
PROCEDURE DRAW VERTICAL LINE
PROCEDURE DRAW HORIZONTALE LINE
PROCEDURE LOOP ERROR
DRAW BIG PICTURE
LOOP
LOOP
INITIALISE PARAMS& SELECT I&O UNITS
CALL DRAW THE FLOW CHART
ADVANCE FILM
ADVANCE FILM
END
END
END
END
END
END
END

3. PROGRAM PROCEDURES AND FUNCTIONS

3.1 Programmer Defined Procedures

(1) INTEGER PROCEDURE GCHAR (character):

Inputs a character in 1900 ALGOL character code and returns as its value the 1900 GROATS equivalent; this is necessary to print out using GROATS procedure 'TYPE'.

(2) INTEGER PROCEDURE MAX (a,b):

Takes two integer values a,b and returns the value of the greater.

(3) PROCEDURE DRAW THE FLOWCHART (top, left, bottom, right):

Generates the instructions to draw the flowchart in the rectangle defined by (top, left) → (bottom, right), where top, left, bottom, right are defined in the selected region units. To achieve the above it calls procedure:

DRAW NODE.

DIMFLOWCHART:=.NAME'\' NODE

(4) PROCEDURE GET NON-BLANK CHAR(ch):

Returns the next non-blank input character as CH.

(5) PROCEDURE PRINT OUT POINTER (top, left, bottom, right):

Takes the parameters top, left, bottom, right and prints them out. The output medium is the lineprinter. This is a debugging aid.

(6) PROCEDURE DRAW NODE (top, left, bottom, right):

To understand the mechanism of this procedure, it is helpful to refer to META DIMFLOWCHART (see Appendix 1), where NODE:=FLOWC. The purpose of this procedure is to draw one or more serials (see DRAW SERIAL).

(7) PROCEDURE DRAW SERIAL (top, left, bottom, right):

SERIAL := STATEMENT(REFINE/EMPTY) $ (PARALLEL)

Draw a SERIAL consists of:

(a) Draw a vertical line.
(b) Draw a statement (see DRAW STATEMENT).
(c) Decide whether a refinement is needed, if so DRAW NEXT LEVEL and DRAW NODE.
(d) Draw one or more parallels which consists of:

DRAW HORIZONTAL LINE.
DRAW NODE.

Note how the recursive nature of dimensional flowcharts occurs in that DRAW SERIAL calls DRAW NODE which itself calls DRAW SERIAL.

(8) PROCEDURE DRAW STATEMENT (top, left, bottom, right):

STATEMENT := ACTION/CONDITIONAL

Draw a statement consists of:

(a) Decide whether a statement is an ACTION statement or a CONDITIONAL.
(b) If it is an ACTION then write the text and update the pointers to skip over it.
(c) If it is a CONDITIONAL then write the text and draw a half square around it.

(9) PROCEDURE DRAW VERTICAL LINE (top, left, bottom, right):

Draws a vertical line joining the points (left, top) and (left, bottom).

(10) PROCEDURE DRAW HORIZONTAL LINE (top, left, bottom, right):

Draws a horizontal line joining the points (left, top) and right, top).

(11) PROCEDURE DRAW NEXT LEVEL (top, left, bottom, right):

Draws the diagonal line for the refinement dimension.

(12) PROCEDURE LOOP ERROR (top, left, bottom, right):

Indicates that a loop error has occurred, and prints out the pointers top, left, bottom, right. After a loop error has occurred, the program terminates.

3.2 GROATS Procedures

(1) SELECT CAMERA(7)·

Selects the camera used to produce hardcopy output - single frames.

(2) IDENTIFICATION·

Initialises the GROATS system and puts identification frames onto the output.

(3) LIMITS (Xmin, Ymin, Xmax, Ymax):

Defines minimum and maximum values of x and y which coincide with corners of the currently selected region.

(4) REGION (Xmin, Ymin, Xmax, Ymax, I):

Defines sub-region I as the part of the currently selected region which has max coordinates (Xmax, Ymax), and minimum coordinates (Xmin, Ymin).

(5) SELECT REGION (I):

Selects region I; all the GROATS procedures will have their positional parameters defined in terms of the currently selected region.

(6) CHARACTER SIZE (height, width):

Defines size of characters in units of the currently selected region.

(7) MOVE TYPE TO (x,y):

Moves typing positions to the point (x,y).

(8) CHARACTER SPACE (CH SP):

Inserts a space of CH SP units between character in the horizontal direction.

(9) TYPE (CH):

Types the GROATS character CH at the current typing position and moves the typing position one position to the right.

(10) PROCEDURE DRAW NEXT LEVEL (top, left, bottom, right):

Draws the diagonal line for the refinement dimension.

(12) PROCEDURE LOOP ERROR (top, left, bottom, right):

Indicates that a loop error has occurred, and prints out the pointers top, left, bottom, right. After a loop error has occurred the program terminates.

3.3 Oxford Utility Library

ALGOBEY(S):

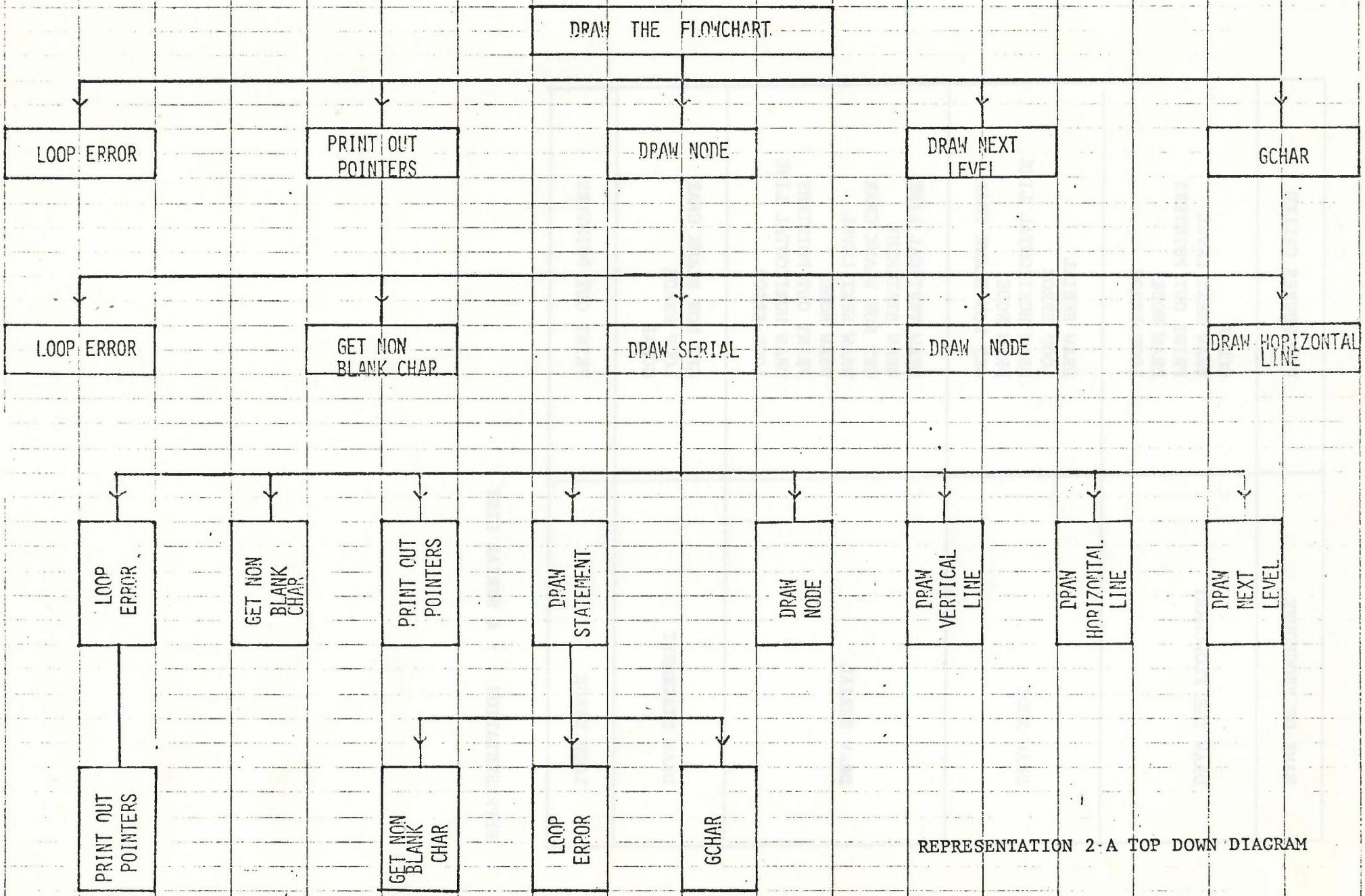
This is a semi-compiled procedure which issues GEORGE commands from within the ALGOL program. It is a Boolean procedure; it returns "*true*" if the command is issued successfully, and "*false*" otherwise (see Appendix 3).

3.4 Procedures Hierarchy

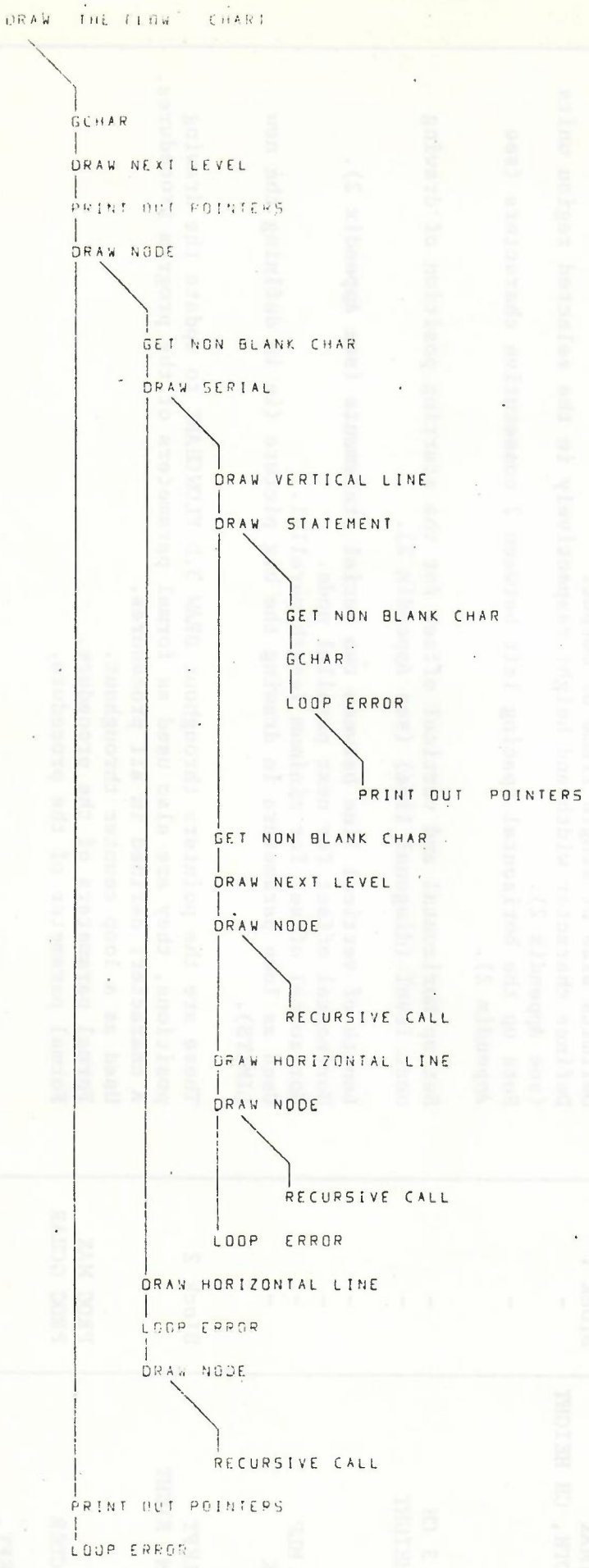
Because of the recursive nature of the procedures used in the program, it may be helpful to list the interaction between different procedures. Below are 3 different representations of the procedures hierarchy and interaction.

NAME OF PROCEDURE	PROCEDURES CALLED
DRAW THE FLOWCHART	GCHAR DRAW NEXT LEVEL PRINT OUT POINTERS DRAW NODE LOOP ERROR
DRAW NODE	DRAW SERIAL LOOP ERROR DRAW HORIZONTAL LINE DRAW NODE GET NON BLANK CHAR
DRAW SERIAL	DRAW VERTICAL LINE DRAW STATEMENT GET NON BLANK CHAR DRAW NEXT LEVEL DRAW NODE PRINT OUT POINTERS DRAW HORIZONTAL LINE LOOP ERROR
DRAW STATEMENT	GET NON BLANK CHAR LOOP ERROR GCHAR
LOOP ERROR	PRINT OUT POINTERS

REPRESENTATION 1 : A SERIAL LIST



REPRESENTATION 2-A TOP DOWN DIAGRAM



Representation 3 : A 3-D Flow Chart

4. LIST OF VARIABLES NAMES

VAR NAMES (1)	SCOPE (2)	DESCRIPTION
TR tr 5	Block 0	Set to '*true*' if the GEORGE command is issued successfully, and false otherwise.
XMAX,YMAX	Block 1	Delimits size of single frame of output.
CH WIDTH, CH HEIGHT	-	Defines character width and height respectively in the selected region units (see Appendix 2).
CH SP	-	Sets up the horizontal spacing left between 2 consecutive characters (see Appendix 2).
WIDTH 3 CH CH 2 HEIGHT	- -	Set up horizontal and vertical offset for the starting position of drawing next level (diagonal line) (see Appendix 2).
DTB	-	Length of vertical line between two serial statements (see Appendix 2).
DRP	-	Horizontal offset for next parallel node.
DELTA MLP	-	Horizontal offset for minimum length parallel.
RJ, RK	-	Used as loop parameters in drawing the big picture (ie in defining the new LIMITS).
TOP, LEFT BOTTOM, RIGHT	Block 2	These are the pointers throughout DRAW 3.D FLOWCHART to update the drawing positions, they are also used as formal parameters of the program procedures.
CH		A character; defined in all procedures.
J		Used as a loop counter throughout.
A, B	PROC MAX	Formal parameters of the procedure.
CHARACTER	PROC GCHAR	Formal parameter of the procedure.
NEW LEFT, NEW RIGHT, NEW TO P NEW BOTTOM	PROC DRAW THE FLOWCHART	Used to reset pointers to new positions to draw next level. They are redefined in DRAW NODE to control drawing positions; they are updated after a drawing instruction has been issued.
DIAGONAL	PROC DRAW NEXT LEVEL	Length of X and Y components of diagonal line.

VAR NAMES (1)	SCOPE (2)	DESCRIPTION
MAX NEW RITHT MAX NEW LEFT	DRAW NODE	Pointers to control drawing positions, updated after a drawing instruction has been issued.
TOP MLP, LEFT MLP, BOTTOM MLP, RIGHT MLP MAX BOTTOM MLP, MAX RIGHT MLP	DRAW NODE	Pointers to denote coordinates of minimum length parallel; used in calling DRAW HORIZONTAL LINE for m.l.p.
TOP SERIAL, LEFT SERIAL, BOTTOM SERIAL RIGHT SERIAL	DRAW SERIAL	Pointers to denote serial drawing positions.
NEXT TOP, NEXT LEFT, NEXT RIGHT, NEXT BOTTOM	DRAW SERIAL	Pointers to denote coordinates for drawing next node.
TOP PARA, LEFT PARA, BOTTOM PARA, RIGHT PARA	DRAW SERIAL	Denote coordinates for drawing positions of minimum width parallel.
TOP TEXT, RIGHT TEXT	DRAW STATEMENT	Denote X coordinates of starting and finishing of text.
ACTION	DRAW STATEMENT	Set to '*true*' if starting character is (') (ie SERIAL) and '*false*' if starting character is (?) (ie CONDITIONAL).

NOTES: (1) Variable names are of integer type except the following:
RJ,RK : REAL
TR5,ACTION : BOOLEAN

(2) BLOCK 0 : the most outer block
BLOCK 1 : the second most outer block
BLOCK 2 : the draw 3-D flow block

5. THE INPUT DATA

Data is presented in a form which uses the following symbols as '*reserve symbols*' to implement different features of a 3-D flowchart.

SYMBOL	MEANING	DRAWING ACTION
!	Denotes R-dimension (ie next level)	Draw diagnonal line
#	Denotes the end of a node	Reset pointers to previous level
? *text* ?	Denotes a conditional statement	Draw a half square around text
' *text* '	Denotes an action statement	Text without half square
-	Denotes a serial-parallel	Draw horizontal line of length DRP followed by DRAW NODE
=	Denotes a minimum length parallel	Draw horizontal line of length DELTA MLP followed by DRAW NODE

The first data record is of the form:

N .NAME *text*;

where:

N is the scaling factor for the whole flowchart,
text describes the title of the flowchart.

The second record will be always a R dimension symbol (!).

the last data record is of the form:

. *text* ;

Note: An example of the input data used to generate the physical construction of the program is included in Appendix 4.

6. ERROR CHECKING AND DEBUGGING AIDS

6.1 Error Checking

(1) Loops: all loops in the program are proved to terminate, and any iteration outside the maximum loop range will result in an appropriate error message stating the whereabouts of the error, eg:

ERROR IN COPY TITLE.

(2) The input data checks: errors in the input data format (eg, errors in statements delimiters) will result in one of the following error messages:

ERROR MESSAGE	CAUSE
ERROR IN COPY TITLE	Either: (a) text exceeding 82-characters (b) (;) missing
.ERROR	(.) end of data marker not encountered
ERROR IN NON BLANK CHAR	Loop to get non blank characters exhausted
ONE OR MORE SERIALS ERROR	More than 5000 serials
MIN LEN PAR LOOP ERROR	More than 50 consecutive min len Parallel nodes
# ERROR	(#) end of node marker not encountered (often when a statement is started by an *illegal* character)
PARALLEL LOOP ERROR	More than 82 consecutive serial - parallels encountered
? OR ' ERROR	Wrong statement type (ie not an action nor a conditional statement)
WRITE OUT TEXT ERROR	More than 999 chs in statement text
ACTION OR ' OR ? ERROR	Delimeters of a statement not compatible (eg starting delimeter ', and terminating delimeter ?)
A LOOP ERROR HAS OCCURRED	This will normally follow the error messages: (a) one or more serials error (b) parallel loop error (c) write out text error (d) .ERROR (e) min len par loop error

6.2 Debugging Aids

Throughout the program, a number of trace statements recording the progress of the program execution are built in as '*comments*' statements of the form:

```
'COMMENT'     WRITE TEXT ('text');
'COMMENT'     PRINT OUT POINTERS (top, left, bottom, right);
```

Leaving out the ALGOL words 'COMMENT' will result in a full trace of the program execution; this is to be used in extreme necessity as it results in a large amount of printer paper.

APPENDIX 1

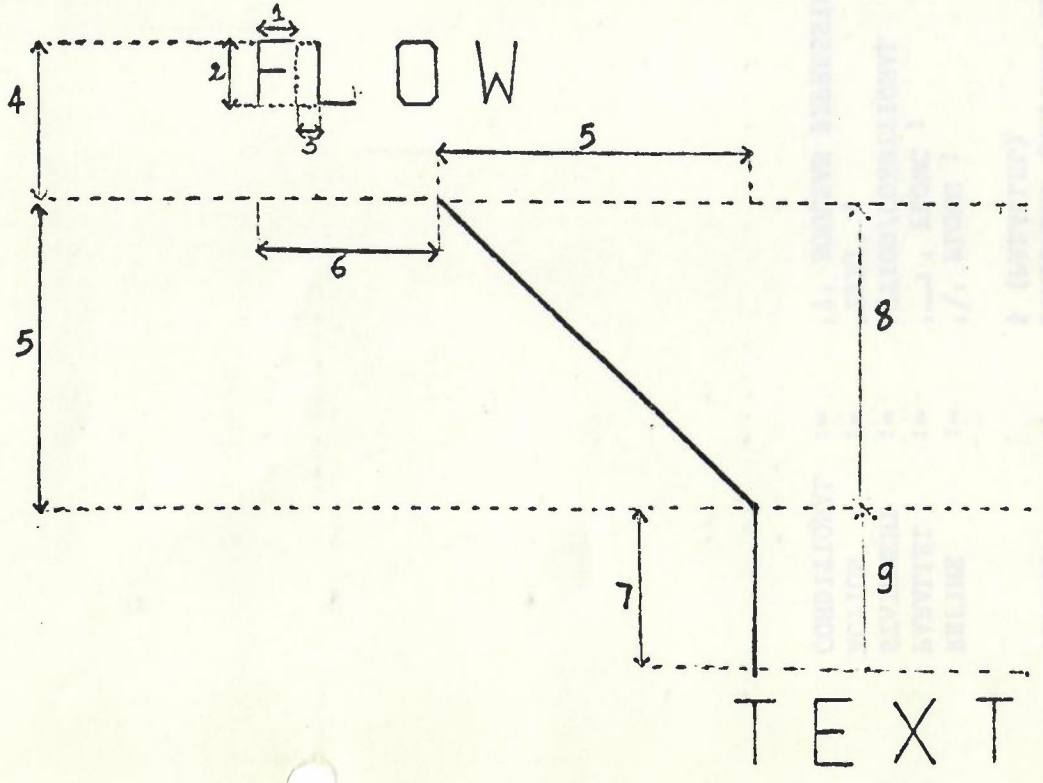
```
META DIMFLOWCHART

DIMFLOWCHART :=     .NAME '\' FLOWC ;
FLOWC        :=     NODE
NODE         :=     SERIAL $ (SERIAL) '#' ;
SERIAL       :=     STATEMENT (REFINE/.EMPTY)
                    $ (PARALLEL)

REFINE       :=     '\' FLOWC ;
PARALLEL     :=     '¬' FLOWC ;
STATEMENT    :=     ACTION/CONDITIONAL
ACTION       :=     'TEXT' ;
CONDITIONAL  :=     '?' BOOLEAN EXPRESSION OR TEXT '?' ;
```

APPENDIX 2

1	CH WIDTH	= 8
2	CH HEIGHT	= 8
3	CH SP	= 2
4	CH 2 HEIGHT	= 16
5	DIAGONAL	= 40
6	WIDTH 3 CH	= 20
7	DTB	= 20
8	DRAW NEXT LEVEL	
9	DRAW VERTICAL LINE	



APPENDIX 3

1. Boolean procedure ALGOBEY(S);

   string S;

2. This procedure issues GEORGE commands from within an ALGOL program. Most commands which do not cause deletion or suspension of the program may be issued by this procedure. The routine returns TRUE if the GEORGE command was issued successfully and FALSE otherwise.

   Since GEORGE parameter substitution involving use of % characters cannot be used directly in an ALGOL string the routine will convert any £ characters in the string to % characters before issuing the command. Parameter substitution is therefore accomplished using £ characters in the original string in positions where GEORGE would require % characters. % characters in the original string will of course be converted at compile time into space characters.

3. Language PLAN

4. Parameters

   S - a character string which consists of the command to be obeyed enclosed in string quotes.

5. Error Indicators

   The routine returns FALSE if the GEORGE command was not issued successfully.

6. Examples

   (a) In the following example NEWFILE will be assigned to *CRO

```
'IF' 'NOT' ALGOBEY ('('AS%*CRO,NEWFILE')') 'THEN'
'BEGIN' 'COMMENT' ERROR ROUTINE;
                .
                .
                .
'END' 'ELSE'
'BEGIN' 'COMMENT' COMMAND OBEYED OK;
```

   (b) If a call to the macro that runs a program includes the parameter:

```
PARAM (F1 FILE1, F2 FILE2)
```

   then to issue the GEORGE commands:

```
AS *CRO,%(F1)
AS *CPO,%(F2)
```

   from within the ALGOL program, the calls to ALGOBEY should be:

```
'IF' 'NOT' (ALGOBEY('('AS%*CRO,£(F1')'))
'AND' ALGOBEY('('AS%*CPO,£(F2)')'))
'THEN' 'GOTO' L99;
```

APPENDIX 4 - DATA USED TO PRODUCE FLOW CHART IN CHAPTER 2

```
 0 1500  .NAME FLOWB-SRC;
 1 !
 2 'PROGRAM DESCRIPTION'
 3 'BEGIN'
 4 !
 5 'INITIALISE PLOTING PROCEDURES'
 6 'BEGIN'
 7 !
 8 'DECLARE VARIABLES &,INITIALISE PARAMS'
 9 'BEGIN'
10 !
11 'DEFINE LIMITS AND REGIONS'
12 'DRAW 3D FLOW CHART'
13 !
14 'DECLARE VARIABLES'
15 'DEFINE PROCEDURES'
16 !
17 'PROCEDURE MAX'
18 'INTEGER PROCEDURE GET CHAR'
19 'PROCEDURE   DRAW THE FLOW CHART'
20 'PROCEDURE GET NON BLANK CHAR'
21 'PROCEDURE DRAW NEXT LEVEL'
22 'PROCEDURE PRINTOUT POINTERS'
23 'PROCEDURE DRAW NODE'
24 'PROCEDURE DRAW SERIAL'
25 'PROCEDURE DRAW VERTICAL LINE'
26 'PROCEDURE DRAW HORIZONTALE LINE'
27 'PROCEDURE     LOOP ERROR'
28 #
29 'DRAW BIG PICTURE'
30 !
31 'LOOP'
32 !
33 'LOOP'
34 !
35 'INITIALISE PARAMS& SELECT I$O UNITS'
36 'CALL  DRAW THE FLOW CHART'
37 'ADVANCE FILM'
38 #
39 'ADVANCE FILM'
40 'END'
41 #
42 'END'
43 #
44 'END'
45 #
46 'END'
47 #
48 'END'
49 #
50 'END'
51 #
52 'END'
53 #
54 .END OF FLOW;
55 ****
56
57
```