CHAPTER 2

THE BASIC STRUCTURES

In this chapter, the author is not trying to explain these basic techniques in the deep detail. The reader can find many valuable books that concerned about them.

2.1 THE STRUCTURE OF COBOL LANGUAGE:

It does not, however attempt to explain how to program the cobol language. It is chiefly concerned with the description of its structure.

Cobol is based on English language, it divided into 4 parts or divisions in the following order.

a) IDENTIFICATION DIVISION:

Its purpose is to identify the program and to include an overall description of the program such as the date of program was written, the date of compilation was accomplished and other information that will serve to document the program.

b) ENVIRONMENT DIVISION:

It specifies the characteristics of the computer used, the location of each file, the technique of file retrieval and input-output control referred in the program.

c) DATA DIVISION:

It describes a detail description of all the data to be used in the program whether to be accepted as input, to mainpulate, to be used in intermediate processing or to be produce as output.

d) **PROCEDURE DIVISION :**

It defines the necessary steps which compose of the specific instructions for solving the desire task. Cobol instructions are written in statements which may be combined to form sentences. Groups of sentences may form paragraph and many paragraphs may be combined to form the highest level that is section.

The basic unit of the **PROCEDURE DIVISION** is the statement. It composed of an action verb follow by group of cobol words. The cobol word may be literal, constant, dataname defined in DATA DIVISION or any reserve words. There are three categories of statements used in cobol language.

d.1) Conditional statement

It is statement containing a conditional that tested, it specifies the truth value of a condition to determine which of alternate path of the program flow is to be taken. The statement are as follows:

IF ... THEN ... ELSE	
ADD	(ON SIZE ERROR)
COMPUTE	("---------")
SUBTRACT	("---------")
MULTIPLY	("---------")
DIVIDE	("---------")
GO TO ... DEPENDING ON ...	
READ	(AT END)
SEARCH	("--")
RETURN	("--")
WRITE	(AT END-OR PAGE)
READ	(INVALID KEY)

WRITE	(INVALID KEY)
START	(" ----- ")
REWRITE	(" ----- ")
PERFORM	(UNTIL)
SEARCH	(WHEN)
STRING	(ON OVERFLOW)
UNSTRING	(" ----- ")

d.2) Imparative statement

It specified the sequence of action to be taken, these statements are:

Arithmetic

ADD

COMPUTE

DIVIDE

MULTIPLY

SUBTRACT

Procedure branching

GO TO

ALTER

PERFORM

STOP

EXIT

Data manipulation

MOVE

EXAMINE

TRANSFORM

INSPECT

STRING

UNSTRING

Input-Output

OPEN

START

SEEK

READ

WRITE

REWRITE

ACCEPT

DISPLAY

CLOSE

Report writer

GENERATE

INITIATE

TERMINATE

Table handling

SEARCH

SET

Sort feature

SORT

RETURN

RELEASE

d.3) Subprogram linkage

These statements are provided for communication between the program and other subprograms.

CALL

ENTRY

GO BACK

EXIT (program)

COPY

ENTER

2.2 THE STRUCTURE OF OUTLINE FLOWCHARTING INPUT:

The outline flowcharting input is a specific type of procedural algorithm, similar to English, that is accepted as input of the flowchart generator program and to be translated into the outline diagrammatic flowchart, Since it is an algorithm, it is a finite set of instructions for carrying out some logic process step by step and each step must be precisely defined.

2.2.1 Characteristic of the outline input

The outline flowcharting input is similar to cobol source program, but it has only one division that is PROCEDURE DIVISION. The outline instructions are written in outline statements which may be combined to form sentences. A sentence is also terminated by period, like cobol sentence, group of sentences may form procedures and each procedure must begin in area A. The basic unit, begin in area B, is the outline statement, it composed of a general data processing or cobol verb and verb identifier, the verb identifier is combination of any English words and symbols.

Unlike cobol, the outline flowcharting input is not a computer language, then there are no fixed format or syntax rules of outline statements, the clearness meaning of action to be performed is only required.

Here, there is an example showed the difference between cobol statement and outline statement.

In cobol:

```
IF TRANSACTION-NUMBER = MASTER-NUMBER
    THEN MOVE NEW-RECORD TO MASTER RECORD
         WRITE MASTER-RECORD.
```

The outline statements may be written like:

```
IF KEY MATCHING
    THEN UPDATE MASTER RECORD.
```

The KEY MATCHING and MASTER RECORD are verb-identifier of the IF and UPDATE verbs respectively.

2.2.2 The additional description of outline flowcharting input

a) The following words must be excluded from outline verb identifier:-

END

ON

AT

SIZE

NEXT

INVALID

WHEN

OVERFLOW

END-OF-PAGE

EOP

Single quotation mark or aphostrophe (but pair of it allowed)

When we need its meaning, the hyphen must be preceded or followed it, like -END or END-.

b) Every cobol verb must be used as outline verb in these manners:-

IF ... THEN ... ELSE	for condition branching
PERFORM	for repetition
GO TO proc-no.	for uncondition branching
GO TO proc1, proc2, DEPENDING ON ...	for case branching

The others cobol verbs for sequence operation.

c) The verbs extended from cobol verbs, when used, it must be submitted as the extended verbs input parameter. The sample of these action verbs list are introduced here:-

Control action verbs

CHARGE

CHECK

CONSERVE

CORRECT

COUNT

EDIT

ENFORCE

ENSURE

FOLLOW UP

LOG

MEASURE

PROVE

REJECT

REPORT

RESTRICT

Give action verbs

DELIVER

DISTRIBUTE

FORWARD

ISSUE

MAIL

MAKE

PAY

PROVIDE

ROUTE

SELL

SEND

SHIP

SUBMIT

SUPPLY

TRANSFER

Help action verbs

AID

ASSIST

PARTICIPATE

PROTECT

SERVE

Push along verbs

DEVELOP

ENCOURAGE

FURTHER

MAINTAIN

Work action verbs

ATTACH

CALCULATE

CLEAR

CHANGE

CONDUCT

VERIFY

Create action verbs

DESIGN

DEVELOP

DEVISE

FORMULATE

INSTALL

ORIGINATE

PLAN

SCHEDULE

Explain action verb

DEFINE

DESCRIBE

INDICATE

SHOW

STATE

Get action verbs

ACCUMULATE

BUY

COMPILE

FIND

GATHER

KEEP

OBTAIN

SECURE

PICK UP

PROCEDURE

PULL

PURCHASE

RECALL

RECEIVE

TAKE

WITHDRAWN

Render decision verbs

DECIDE

DETERMINE

REVIEW

WEIGH

Stop action verbs

DELETE

PREVENT

Study or apprase verbs

ANALYZE

CHECK

COMPARE

EVALUATE

FORCAST

IDENTIFY

RESTORE

INTERVIEW

INVESTIGATE

MEASURE

PLAN

STORE

SURVEY

Tell other action verbs

ADVISE

ASSIGN

NOTIFY

ORDER

PRESCRIBE

RECOMMEND

SUBMIT

Tie together verbs

CONFER

CONTRACT

COORDINATE

RECONCILE

REPRESENT

CONNECT

CONCATENATE

DESTROY

ENTER

FILE

HANDLE

HIRE

INSERT

LIST

LOCATE

MAKE

PLACE

PREPARE

PRINT

PROCESS

RETAIN

RUN

SEPARATE

TABULATE

TRANSCRIBE

USE

d) Since there are no fixed syntax rule in every outline statement, then we can use:

AT END

INVALID KEY

ON SIZE ERROR

ON OVERFLOW independently with every statement.

For example, when we use the ON OVERFLOW in cobol source program, it must only follow the STRING or UNSTRING verb.

STRING ... ON OVERFLOW ...

But in outline statements, it can be used with any statements, like:

PUT THE REMAIN PART OF TEXT INTO THE STACK ON OVERFLOW

PERFORM POP-STACK UNTIL THE STACK IS EMPTY.

The another example is AT END, in cobol, the AT END must be only used with READ statement:

READ ... AT END ...

But in outline statements:

DO THIS STEP AT END GO TO NEXT-STEP.

e) The 2 additional verbs are provided, the FLOWBEGIN for flowchart starting box and FLOWEND for flowchart ending box. For example:

```
PROCEDURE DIVISION
INITIAL.        FLOWBEGIN
                INITIALIZE SUM TO ZERO.
ACCUM.          ADD 1 TO SUM.
                IF SUM EQUAL TO 100
                   PRINT THE SUM TOTAL
                   FLOWEND
                ELSE GO TO ACCUM.
```

f) The EXIT verb for a common end for a series of procedure can be used in anywhere in the procedure. For example:

```
ACCUM.          ADD 1 TO SUM.
                IF SUM EQUAL TO 100 THEN PRINT SUM
                   TOTAL EXIT ELSE GO TO ACCUM.
```

g) The comment of outline text may be inserted on any line by placing the percent character (%) in the continuation area of the line and the comment text must begin in area B of that line.

2.2.3 The guidance of outline flowcharting input preparing

The method of outline input preparing advocated here used an outline format to describe the action to be performed, it is an basic technique for preparing both data processing and user outline input. After the outline or algorithm is prepared, the following questions should be asked:

- Does the outline really cover the subject and does it represent the most straight forward way of describing the logic to the programmers or the readers?

- Does the outline have too few or many procedures?

- Does it include any unnecessary redundance text?

- Does it convey an impression of clearness and continuity?

After this preparation, the step of outline text coding should take place with these manner:

a) Select the title and the purpose that state in one short phrase what the outline cover.

b) Select the names assign to procedures and also comment or describe the function of them.

c) Sentences in a procedure should not be written too broad, otherwise, they may lack of effectiveness.

d) Try to use simple, short sentence and statement to communicate ideas, avoid wordy.

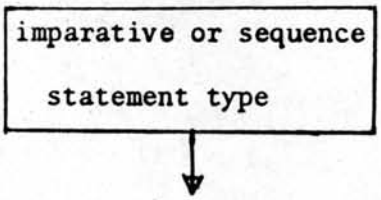
e) Every sentence and statement must begin with cobol verb or action verb in the present tense.

2.3 THE STRUCTURE OF FLOWCHART:

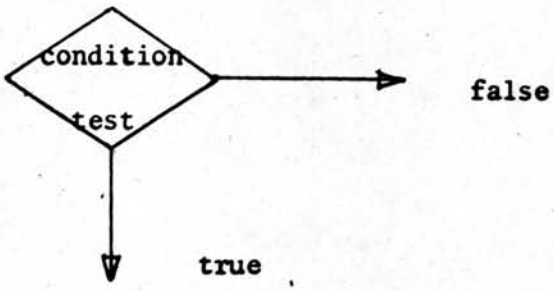
A flowchart is a diagrammatic representation of an algorithm, it composes of simple graphics with boxes and arrows, the boxes represent the appropriate texts and the arrows are the interfaces. A programming flowchart is a diagrammatic representation of a logic flow of program operation and is composed of linked symbol and the appropriate statement. Flowchart are widely used in programming procedure because of the order they imposed on the thinking of the programmer. The ease with which information may be assimilated from them.

Every program flow was built in these structure

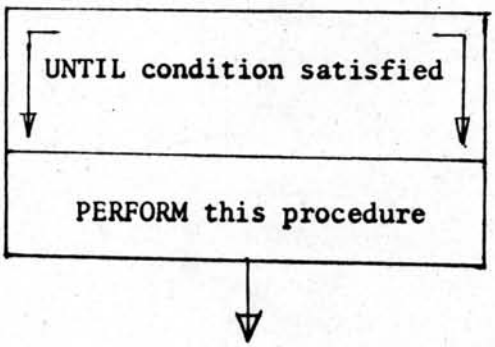
1) the sequence flow



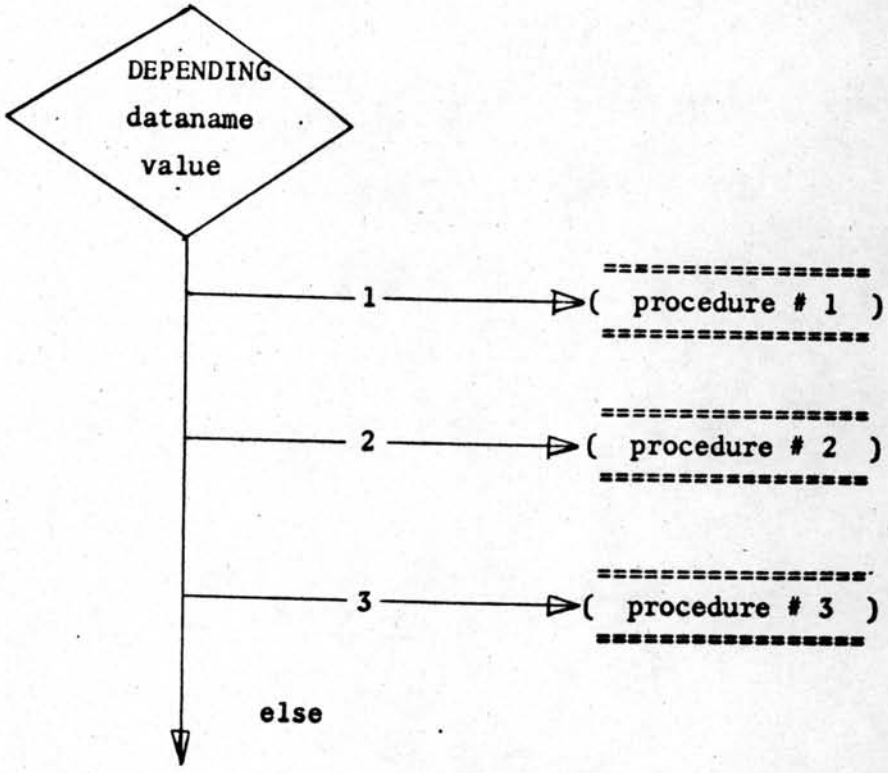
2) the condition branch flow



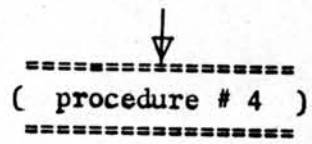
3) the repeated operation flow



4) the case flow



5) the uncondition branch flow



6) <u>begin and terminate flow</u>