CHAPTER 5

CONCLUSION AND RECOMMENDATION

It is felt that the objective of this thesis has been accomplished and provided a program for generating the cobol source program and outline text. These can assist the data processing man to solve the problem in program documentation. In the program written, the author tried to apply the structure programming technique in order to make it well-structured and more easy to modify in the near future. The program outline flowchart in the appendix B can guide the one who is interesting to inderstand its outline and logic. There are some recommendation that have to be mentioned:

5.1 THE LIMITATION

5:1:1 It is specified that the cobol source program and the outline input wanted to be generated must free from errors, therefore, the cobol source program must be compiled for its clearance before, and the outline input must be checked by the users for its meaning and the correctness of text. If there have some errors occured either in cobol source program or in outline text input, the flowcharting may be generated incorrectly or may not be generated, with this importance point, the structure of outline flow-charting input in the chapter 2 and the cobol reference manual can help the users to clean the syntax error from the outline input and the cobol source input.

5.1.2 With the input to be generated :

- the maximum of 20 nested "IF" are allowed in one sentence
- the maximum of 10 "ELSE IF" are allowed in one paragraph
- the length of one sentence must not more than 10,000 characters
- an statement must not exceed 506 characters

- it must have not move than 500 procedures in the source

    program that wanted to be generated.

    however, these limitation can be reduced by increasing the size
of the respective datanames defined in the program, then for the one who
want to modified them, the description of dataname tables and entries in
appendix F can aid the modifier in adjustment and reduction of program limit-
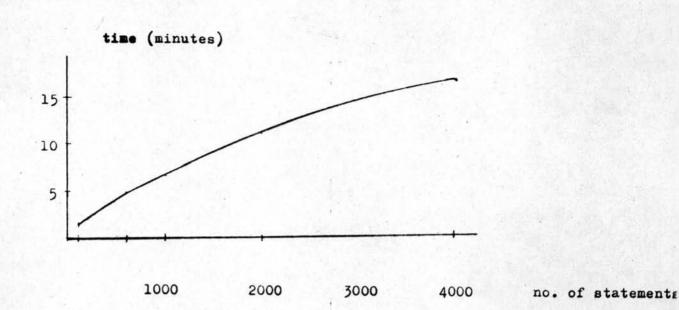tation and program **size.**

5.2 ThE PROGRAM PERFORMANCE

5.2.1 Since the program was written with some basic cobol verbs and
uncomplicated cobol statement, the simple logic set of instructions are also
applied for the purpose of program operatability on several computer, therefore,
it is not quite efficient as it should be in the term of execution time.
the author ever submit the sample about 12 cobol source program to be
generated on the **IBM** S/4331 with the memory 1MB the generating time for the
largest 4000 statements cobol source input is nearly 16 minutes. Anyway it
still much more efficient and quickier than the human-drawn.

5.2.2 The generating time consumption is based on 3 factors :-

- the number of source input statements
- the number of complicated statements in source input,

    especially, "IF", nested "IF" and "ELSE IF" statements
- the four additional requirements

First, in the number of source input statement affection, it is the
fact that more statements consume more generating time for approximate
100, 600, 1000, 2000, 4000 statements are 1.4,4.3,6.3,10.6 and 16.4 minutes,
respectively. The reader may see that the generating time proportions are
increasing less than the number of statement proportions. The graph of this
sample is as follow:-

time (minutes)

```
15 ┤
10 ┤
 5 ┤
```

        1000   2000   3000   4000   no. of statements

        Second, in the case of complicated statement affection,
is quite important factor. The most complicated statements are "IF", nested
"IF" and "ELSE IF" statements, and "IF" statement take time to generate appro-
ximate 13 times more than a simple statement like "MOVE", the 2 nested "IF"
consumes 4.5 times more than a single "IF" and the "ELSE IF" takes 3 times over
"IF" statement. For example, if we have a statement to be generated as follow:

        IF ... IF ... ELSE ... ELSE ... IF ...

        It will consume = 4.5 x 13 + 3 x 13

                         = 91 times more than a simple statement.

        Third, for the additional requirement affection, it can also
consume more generating time, but it is very difficult to measure. However,
by the program logic we know that the most time consumption in all of the
additional requirement is the "new more meaning name" requirement and the
least time consumption that we con neglect are the "title documentation"
requirement.

5.3 THE GUIDANCE

5.3.1 The reader can see the graphics shape in the flowchart printed are not beauty, since they are plotted by digital printer. If there have continuous plotters attached with the computer system, the diagram can be plotted on it as beautiful as human drawn, but the reporting section of this flowchart generator program must be modified or rewritten and some plotting instruction like, the plotting direction control, the plotter pen selection must be used for continuation plotting.

5.3.2 The flowchart generated from this tool is well diagram, if the structure of outline input or the cobol source input are divided into main level and successive levels, the main or top level shows the main components of the system and its successive levels decompose these components.

5.3.3 For outline flowchart generating, the author intend that is is used for programming aid in the program development stage, but the reader can used it for generating any procedural outline like as the author used it for generating the procedure for reading this thesis (see appendix D).

At last, the recommendation cannot be accomplished, without the following sentence:

"Computer science as our life, is dynamic, we don't know the tomorrow ahead of it. For the author's really hope, tomorrow, may be anyone could find a standard, economical and easy used tool to help the data processing man to handle all of data processing documentation".